د تخنيکي او مسلکي زده کړو اداره

د تخنيکي او مسلکي زده کړو جنوب ختيځ (غزني) زون رياست

پکتيکا د تخنيکي او مسلکي زده کړو آمريت

مرکز کثرالرشتوی انستيتوت

ډيټابيس څانګه

د پروګرام ليکنې اساسات (C++)

تولکي: ديارلسم

سمستر:دوهم

لارښود او ايډيټ: استاد فريد احمد «ياد»

ژباړونکي: ذبيح الله «تتيوال»

کال.....1405

لومړی فصل

دا فصل د پروګرام لیکنې د بنسټیزو او ابتدایي مفاهیمو لکه د پروګرام لیکنې اهمیت، تعریف او تاریخچه، او د پروګرام لیکنې د چاپېریال په اړه دی، چې موضوعګانې یې په ترتیب سره په تفصیل سره بحث شوې دي.

کمپیوټر یو الکترونیکي وسیله ده چې د مختلفو مسایلو د حل لپاره کارول کېږي .د» مسألو « (Problems)څخه هدف هغه ستونزې یا ننګونې دي چې مور یې د کمپیوټر په مرسته د حل لارې پیدا کوو .د مسألې د مفهوم د روښانتیا لپاره لاندې مثالونه په پام کې ونیسئ:

- مسأله (Problem) په انجینرۍ برخه کې کېدای شي د یو موټر ډیزاین کول یا د یوې ودانۍ د نقشې جوړول وي.
- مسأله په طبي برخه کې کېدای شي د ناروغیو دقیق تشخیص، د روغتون د مدیریت سیستم کمپیوټري کول، یا د یو روغتون د درملو د ګودام ساتنه وي.
- مسأله په سوداګرۍ برخه کې کېدای شي د بانکي معاملو کمپیوټري کول، د شتمنیو او مالونو لېست، د اسعارو بازار کنټرول، د مالیاتو محاسبه، یا د بانکي حساب ساتنه وي.

نو ویلی شو چې د کمپیوټر په مرسته د مسایلو حل کول یو ستونزمن خو په عین حال کې نوښتګر بهیر دی، ځکه چې دا بهیر تحلیل، تجزیه، پلان جوړونه او منطقي تصمیم نیونه پکې شامل وي .د یادونې وړ ده چې د مختلفو مسایلو د حل لپاره باید اړین او منطقي ګامونه پورته شي تر څو سم او مطلوب حل ته ورسېږو.

۱.۱ پروګرام لیکنه

پروګرام لیکنه په حقیقت کې د یوې مسألې د حل لارې موندلو بهیر دی .د پروګرام لیکنې ژبه د کمپیوټر لپاره د کوډونو د لیکلو لپاره کارول کېږي.

د پروګرام لیکنې ژبه د هغو لارښوونو مجموعه ده چې د هغې له لارې کمپیوټر ته امر کېږي څو

ځانګړي عملیات ترسره کړي .دا پروګرام لیکونکی دی چې د بېلابېلو پروګرامي ژبو په مرسته کمپیوټر ته لارښوونه کوي تر څو مطلوب عملیات ترسره کړي.

۱.۲ د پروګرامي ژبو تاریخچه

په ۱۹۶۰ کال کې یو شمېر پروګرامي ژبې رامنځته شوې چې هرې یوې ځانګړې کارونې لرلې .خلکو فکر وکړ چې ولې د څو ژبو د زده کړې پر ځای یوه داسې ژبه ونه زده کړي چې په ټولو برخو کې وکارول شي .له همدې امله نړیوالي کمېټي د Algol 60 ژبه رامنځته کړه، خو دا ژبه ستړې کوونکې او ډېره عمومي وه، د خلکو له خوا یې ښه استقبال ونه شو او له همدې امله ناکامه شوه.

له دې وروسته یوه نوې ژبه د کمپیوټر د ترکیبي ژبې په نوم CPL (Combine Programming Language)د کمبریج پوهنتون له خوا رامنځته شوه .وروسته معلومه شوه چې د دې ژبې زده کړه او عملي کول ډېر ستونزمن دي .له دې وروسته بله ژبه د BCPL (Basic Combined Programming Language)په نوم رامنځته شوه، په دې هیله چې پخوانۍ ستونزې ونه لري، خو څرګنده شوه چې دا ژبه کمزورې ده او یوازې په ځانګړو برخو کې کارېده.

په همدې وخت کې بله ژبه د B په نوم رامنځته شوه چې د یونکس (Unix) عملیاتي سیستم لپاره د کین تامسن (Ken Thompson) له خوا په AT&T Bell لابراتوار کې لیکل شوې وه.

په ۱۹۷۲ کال کې یو بل شخص چې ډینس ریچي (Dennis Ritchie) نومېده، د B او BCPL د ځینو ځانګړتیاوو او خپلو نظریاتو په یوځای کولو سره یوه نوې ژبه جوړه کړه چې د C ژبې په نوم یادیږي C .ژبه یوه عمومي پروګرامي ژبه ده .دا ژبه د دې لپاره جوړه شوې وه چې د یونکس عملیاتي سیستم کوډونه بیا ولیکل شي.(Jones & Aitken, 2014)

په ۱۹۷۹ کال کې پروګرام لیکونکي بیارن سټراسټروپ (Bjarne Stroustrup) د امریکا د نیوجرسي ایالت په Bell لابراتوار کې د **C++ ژبې** پر جوړولو کار پیل کړ .C++ د C ژبې پرمختللې بڼه ده او د شیګرا (Object Oriented Programming) ژبو له ډلې څخه شمېرل کیږي .دا ژبه ګرافیکي او دوستانه چاپېریال لري او زده کړه یې نسبتاً اسانه ده.

لکه څنګه چې مخکې یادونه وشوه، C++ د C پرمختللې بڼه ده او تقریباً ټول هغه پروګرامونه چې په Cکې چلېږي، په C++ کې هم اجرا کېدای شي .د C++ مهمه ځانګړتیا دا ده چې پکې د کلاسونو (Classes)، وراثت (Inheritance) او دندو (Functions) مفاهیم شامل دي .دا ځانګړتیاوې د

دې لامل شوي چې پيچلي ډيټا ډولونه (Abstract Data Types) جوړ شي، د کلاسونو ځانګړتياوي ميراث شي او د څو بڼو ملاتړ (Polymorphism) وکړي (Jones & Aitken, 2014).

په عمومي ډول، د پروګرام ليکنې ژبې په پنځو ډلو وېشل کېږي:

1. د ماشين ژبه
2. د ټيټې کچې ژبې
3. د لوړې کچې ژبې
4. د څلورم نسل ژبې
5. د پنځم نسل ژبې

اوس به هره يوه له دغو ډلو ژبو په بحث کې وڅېړو:

د ماشين ژبه

ماشين ژبه هغه ژبه ده چې له 0 او 1 څخه جوړه شوې وي او د کمپيوټر لپاره د پوهېدو وړ وي .يعنې اړتيا نه وي چې د 0 او 1 کوډونه د کوم ژباړن (Compiler) يا (Interpreter)له لارې وژباړل شي تر څو کمپيوټر يې درک کړي.

د ټيټې کچې ژبې(Low Level Languages)

دا هغه ژبې دي چې له ماشين ژبې سره ډېر ورته والی لري .د دې ژبو لاربښوونې د ځانګړو سمبولونو په بڼه وي چې د هغو په وسيله بېلابېل پروګرامونه جوړېږي .د اسمبلی ژبه (Assembly) هم د ټيټې کچې ژبو له ډلې څخه ده .يادونه کېږي چې د اسمبلر (Assembler) ژباړن د اسمبلی ژبې د جملو د ماشين ژبې ته د ژباړې لپاره کارول کېږي.

د ټيټې کچې ژبو جملې د کمپيوټر لپاره د انسان په پرتله اسانه وي، ځکه چې دا ژبې له ماشين ژبې سره زيات ورته والی لري.

د لوړې کچې ژبې(High Level Languages)

دا هغه ژبې دي چې جملې يې د انسان په ژبه جوړې او ترتيب شوې وي .له همدې امله په دې ژبو کې پروګرام ليکل د پروګرام ليکونکي لپاره اسانه وي .خو کمپيوټر دا ژبې مستقيم نه شي پېژندلی، نو

ژباړې ته اړتيا لري .له همدې امله دا ژبې د کمپايلر (Compiler) يا انټرپريټر (Interpreter) په وسيله ژباړل کيږي.

لکهJava:، BASIC ، COBOL او C++ د لوړې کچې ژبو له ډلې څخه دي.

د څلورم نسل ژبې(Fourth Generation Languages)

دا ژبې هم د لوړې کچې ژبو له ډلې څخه دي، خو په دې کې پروګرام ليکنه لا اسانه وي .دا ژبې زياتره د ډيټابېسونو د جوړولو لپاره کارول کيږي .لکهFoxPro :، MySQL او Oracle د څلورم نسل ژبو له ډلې څخه دي.

د پنځم نسل ژبې

دا نوې ژبې هم د لوړې کچې ژبو له ډلې څخه دي .د دې نويو ژبو ځانګړې توپير له نورو ژبو سره دا دی چې په دې ژبو کې پروګرام ليکنه اسانه وي، ځکه د پنځم نسل ژبې له ګرافيکي ژبو څخه دي . Java، Visual Basic او C++ د شيګرا ژبو له ډلې څخه ګڼل کيږي.

۱.۳.د ژبې ژباړونکي(Translators Language)

لکه څنګه چې ټولو ته معلومه ده، کمپيوټر يوازې 0 او 1 پېژني .په بل عبارت، د کمپيوټر ژبه 0 او 1 ده .کله چې يو کوډ يا بيان (Statement) په لوړو ژبو کې وليکل شي، کمپيوټر يې نه شي درک کولی او اړتيا وي چې دغه کوډ د 0 او 1 ژبې ته واړول شي تر څو کمپيوټر يې وپېژني او مطلوب عمل ترسره کړي .له همدې امله ټولې پروګرامي ژبې ژباړونکي لري چې د لوړې کچې ژبې کوډونه د ماشين ژبې(0او1) ته بدلوي.

په عمومي ډول دري ډوله ژباړونکي شته چې په لاندې ډول دي:

۱.۳.۱اسمبلر(Assembler)

دا ژباړن د اسمبلی ژبې کوډونه د ماشين ژبې (0او1) ته بدلوي .د اسمبلر ژباړن يوازې د اسمبلی ژبې لپاره کارول کيږي .اسمبلی ژبه له ټيټې کچې ژبو څخه ده او په کې پروګرام ليکل هم ستونزمن وي، له همدې امله اوس مهال د اسمبلی ژبې استعمال ډېر کم شوی دی.

۱.۳.۲ کمپایلر(Compiler)

دا ژبارن د لوړې کچې ژبو کوډونه د کمپیوټر ژبې (0او1) ته بدلوي .د C++ ژبه او یو شمېر نورې پروګرامي ژبې له کمپایلر څخه استفاده کوي .دا ژبارن د پروګرام ټول کوډونه په پای کې، وروسته له دې چې پروګرامر د Run تنۍ کېکاږي، په 0 او 1 بدلوي .که په پروګرام کې کومه تېروتنه موجوده وي، لکه د سیمي کولن (;) نه لیکل د یوې جملې په پای کې یا کومه غلطه لارښوونه، نو کمپایلر دغه تېروتنې پیدا کوي او د پیغام په شکل یې پر سکرین ښیي .پروګرامر وروسته له دې چې پیغام ولولي، تېروتنې سموي او پروګرام بیا اجرا (Run) کوي .که ټولې تېروتنې سمې شوې وي، پروګرام اجرا (Execute) کېږي او پایلې پر سکرین ښکاري.

۱.۳.۳ تفسیر کوونکی(Interpreter)

دا ژبارن پروګرام کرښه په کرښه ژباړي .په دې معنا چې کله پروګرام لیکونکی یوه جمله بیانیه په بشپړ ډول ولیکي او د کیبورډ د Enter تنۍ کېکاږي، نو Interpreter په اتومات ډول هغه کرښه ژباړي .که په بیانیه کې کومه ستونزه وي، تفسیر کوونکی سمدستي یو پیغام پر سکرین ښیي او موجوده تېروتنه په هماغه کرښه کې مشخصوي چې باید اصلاح شي .په دې حالت کې کرسر بلې کرښې ته نه ځي، یا هم هماغه کرښه په سره رنګ ښیي .د غلطۍ له اصلاح وروسته کرسر بلې کرښې ته ځي او د لیکنې رنګ هم سم تور کېږي.

په همدې ډول کله چې پروګرام لیکونکی په دوهمه کرښه کې بیانیه ولیکي او Enter تنۍ کېکاږي، که په روانه کرښه کې کومه تېروتنه وي، تفسیر کوونکی پیغام ښیي، او که کومه تېروتنه نه وي، نو کرسر بلې کرښې ته ځي او پروګرام لیکونکی کولی شي بل بیان ولیکي.

نو ویلی شو چې د Compiler سرعت د Interpreter په پرتله زیات وي، ځکه Compiler ټول پروګرام په پای کې یوځل اجرا کوي.

دوهم فصل

دا فصل د الګوريتم(Algorithm) ، فلوچارت او په پروګرام ليکنه کې د هغوی د رول په اړه دی، چې موضوعګاني يې په ترتيب سره په تفصيل سره څېړل شوي دي.

د مسايلو د حل لپاره تر ټولو زيات کارېدونکي دوه وسيلې عبارت دي له:

۱ .**الګوريتم**

۲ .**فلوچارت**

الګوريتم(Algorithm)

د الګوريتم کلمه د نهمې پېړۍ د مشهور عرب رياضيدان الخوارزمي له نوم څخه اخيستل شوې ده .دا نوم د وخت په تېرېدو سره له Alkhowarism څخه Algorism او بيا بالاخره د Algorithm په بڼه بدل شوی دی.

الګوريتم د ساده جملو مجموعه ده چې په انګليسي ژبه او حسابي جملو سره ليکل کېږي (Klousen, 2017).

الګوريتم يو مرحلهوار پروسيجر دی چې د ښلابښلو مسايلو د حل لپاره کارول کېږي (Backman, 2012).په بل عبارت، الګوريتم د يوې مسألې مرحله په مرحله حل دی چې په اسانۍ سره د پوهېدو وړ وي .په حقيقت کې الګوريتم د مشخصو مرحلو مجموعه ده چې د هغو له تېرېدو وروسته مطلوبې نتيجې ته رسېږو.(Lippman, 2002)

۱.۴ د الګوريتم ځانګړتياوې(Properties of Algorithm)

هر الګوريتم بايد لاندې ځانګړتياوې ولري:

1. ساده وي.
2. واضح او مشخص وي.
3. ورودي (Input) ولري.

4. د مسألې لپاره یوازینی (Unique) حل وړاندې کړي.

5. له محدودو مرحلو وروسته پای ته ورسېږي.

6. لږ تر لږه یوه پایله (Output) ولري.

د الګوریتم د جوړولو شرطونه

د دې لپاره چې د یوې مسألې لپاره الګوریتم جوړ کړو:

1.باید مسأله په سمه توګه درک کړو.

2 . باید معلومه وي چې څه ډول پایله (Output) ترلاسه کوو.

3 ورودي (Input) یې باید مشخص وي.

۴ .هغه پروسه چې قیمتونه (Input) داخلوي او په نتیجه کې خروجی تولیدوي، باید ډیزاین شوې وي.

۵ .د الګوریتم سموالی باید ارزیابي او امتحان شي، یعني معلومات د ازمایښتي ډول داخل شي او نتیجه یې وڅېړل شي.

۶ .که مطلوبه نتیجه ترلاسه نه شي، نو الګوریتم باید بیا له سره وکتل شي.

۱.۵د الګوریتم نښې(Notations of Algorithm)

د الګوریتم د لیکلو پر وخت لاندې نښې باید په پام کې ونیول شي:

I . نوم :(Name) مسأله مشخص کوي.

ii . د مرحلې شمېر :(Number Step) هره لاربښوونه د ځانګړي نمبر له لارې مشخص کېږي.

iii . تبصره :(Comments) د جملو او عملیاتو وضاحت کوي .تبصرې د مربع قوسونو کې لیکل کېږي.

iv . پیل / پای :(Beginning / Termination) د الګوریتم پیل او پای ښیي.

مثال 2-1الګوريتم وليکئ چې درجه حرارت له سانتي ګراد څخه فارنهايت ته واړوي .
الګوريتم :د سانتي ګراد څخه فارنهايت ته د حرارت بدلول مرحله 1: پيل مرحله 2 : C ولولئ مرحله 3 : F = 9/5 * c+ 32محاسبه کړئ مرحله 4 :F ښکاره کړئ مرحله 5 : پای
مثال2.2 الګوريتم وليکئ چې مفاد محاسبه کړي، په دې شرط چې ټول سرمايه، د مفاد فيصدي او د وخت موده ورکړل شوې وي.
الګوريتم :د ساده مفاد محاسبه مرحله 1 : پيل مرحله 2 : د ټولې سرمايې قيمت، د مفاد فيصدي او د وخت موده داخل کړئ مرحله 3 : مفاد محاسبه کړئ مرحله 4 :د مفاد اندازه ښکاره کړئ مرحله 5 : پای اندازه مفاد = ټوله سرمايه + د مفاد فيصدي + د وخت موده100 /

۱.۵.۱ د الګوريتم ګټې

۱. د ستونزو تعقيب آسانه دی ځکه ټولې مرحلې په ترتيب سره ليکل شوې دي.

۲ . د الګوريتم جوړول آسانه دي ځکه د پروګرامي ژبو پوهې ته اړتيا نه لري.

۳ . الګوريتم موږ سره مرسته کوي چې پروګرام په سمه توګه وساتو.

۱.۵.۲د الګوريتم نواقص

۱. د پيچلو او مغلقو مسايلو لپاره د الګوريتم جوړول ستونزمن او وخت‌اخيستونکی دی.

۲. د پيچلو مسايلو (Complex Logic) درک کول د الګوريتم په وسيله سخت دي.

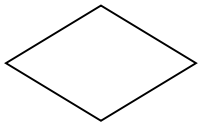
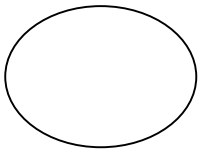
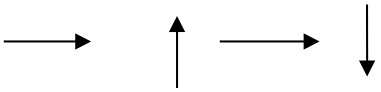
۱.۶فلوچارت(Flowchart)

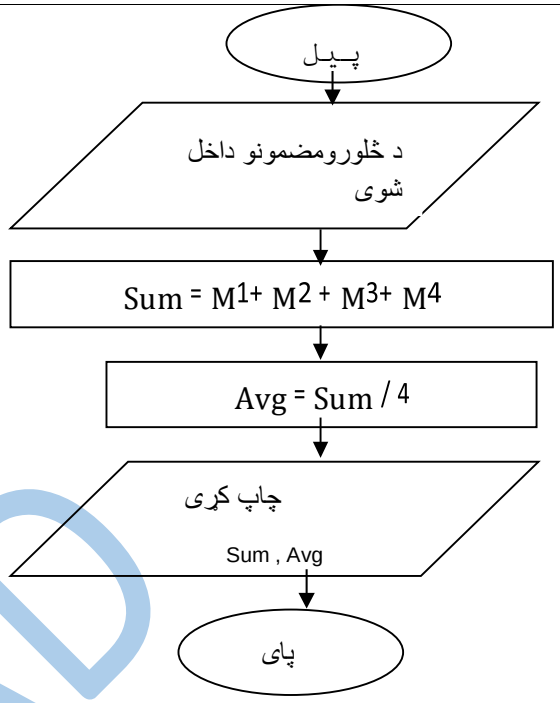
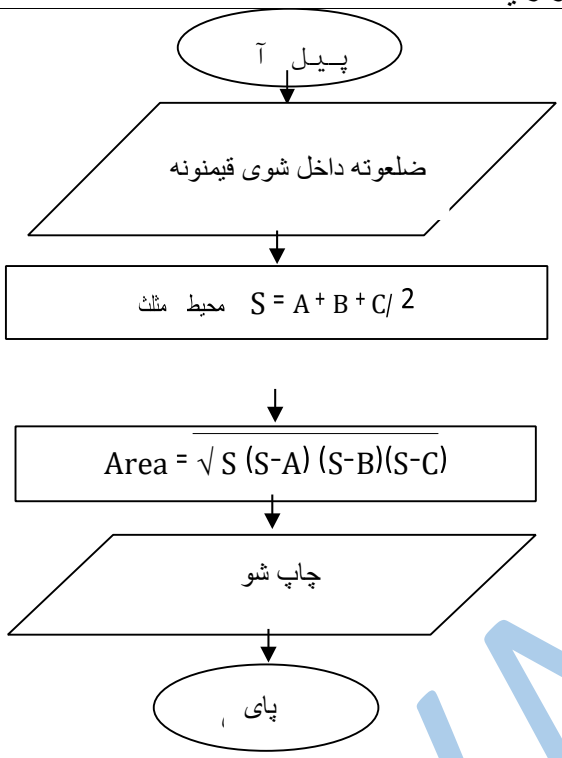
فلوچارت د الګوريتم د مرحلو تصويري بيان ته ويل کيږي .فلوچارت د مسألې د حل طريقه، اړوند حسابونه، د تصميم نيونې نقطې او نور مهم معلومات ښيي.

فلوچارتونه د هندسي شکلونو او سمبولونو په وسيله جوړيږي .هر هندسي شکل ځانګړې دنده ښيي .دا شکلونه د تيرونو (Arrows) په وسيله سره نښلول کيږي او د يوې مسألې بشپړ حل په تصويري ډول ښيي.(Halterman, 2018)

جدول (2-1) د بيلابيلو هندسي اشکالو او د هغوی د کارونې موارد بيانوي:

سمبول	معنی
	آغاز / انجام(Start / End) دا شکل د پروګرام د پيل او پای ښودنه کوي.
	وروډي / خروجي(Input / Output) دا شکل د وروډي قيمتونو د لوستلو او د خروجي ښودلو لپاره کارول کيږي.

	پروسس کوونکی(Processing) ټول حسابي عملیات او فورمولونه په دې شکل کې درج کېږي.
	تصمیم نیونه(Decision) په دې شکل کې د قیمتونو مقایسه او د تصمیم نیونې پروسه ترسره کېږي.
	د دستورو تکرار(Repetition) دا شکل ښیي چې په کې شامل دستورات څو ځله تکرارېږي.
	مخکې تعریف شوې پروسه (Pre-defined Process)دا شکل د هغو محاسبو نمایندګي کوي چې مخکې تعریف شوي وي، لکه د عملیاتو هغه ګروپ چې په بل ځای کې مشخص شوی وي.
	ارتباط ورکوونکی(Connector) دا شکل د فلوچارټ د یوې برخې د داخلېدو یا وتلو نقطه له بلې برخې سره نښلوي.
	د جهت ټاکونکی(Flow Direction) دا اشکال د پروسې د جریان جهت ښیي.

مثال :۲.۳ فلوچارت رسم کړئ چې د ۴ مضمونونو نمرې داخل کړي، مجموعه او اوسط يې ترلاسه کړي؟	مثال :۲.۴ فلوچارت رسم کړئ چې د مثلث مساحت ترلاسه کړي او د A ، B او C ضلعې ولري؟
	

۱.۶.۱د فلوچارت ګټې • (Advantages of Flowchart) د پیچلو او مغلقو مسایلو حل د هندسي اشکالو په وسیله اسانه کوي.

•دا د پروګرام د مستندسازۍ (Documentation) یو ډول دی.

•د فلوچارت په وسیله پروګرام په مؤثره او اسانه ډول کوډ کېږي.

•د پروګرام د تعقیب او ازموینې لپاره ښه طریقه برابروي.

•پروګرامر ته دا زمینه برابروي چې د پروګرام هره برخه بدله کړي.

۱.۶.۲د فلوچارت محدودیتونه • (Limitations of Flowchart) دا طریقه د لویو، مغلقو او پیچلو مسایلو د ښودلو لپاره مؤثره نه ده، ځکه دا ډول مسایل په واضح ډول نه شي بیانولی.

•که کومه لارښوونه د حلقوي (Loop) بڼه ولري، نو د هندسي اشکالو تکرار رسمول ضروري کېږي.

۱.۶.۳د فلوچارتونو د رسم کولو قواعد(Rule for writing flowcharts)

•فلوچارت له پاسه ښکته یا له ښي نه چپ رسم کېږي.

•فلوچارت تل د پیل (Start) له سمبول څخه شروع کېږي او د پای (End) په سمبول ختمېږي.

•د جهت لرونکي تیرونه (Arrows) د هندسي اشکالو د نښلولو لپاره کارول کېږي.

•په هر فلوچارت کې لږ تر لږه یو د پای سمبول (End) باید موجود وي.

•شرطي شکل (Decision) باید یو داخل او دوه خارج جهتونه ولري.

•جملې او بیانیې باید د هندسي اشکالو په دننه کې ولیکل شي.

د کمپیوټري حل لارې پرمختګ(Development of Computer Solution)

a .کوډ لیکنه(Coding)

کوډ لیکنه په حقیقت کې د الګوریتم یا فلوچارت د جملو ژباړه د پروګرام لیکنې ژبو ته ده، لکه++C ، Java، Fortran او.Pascal

b . ردیابي او ازموینه(Testing and Debugging)

د دې لپاره چې مطلوبه نتیجه ترلاسه شي، اړتیا ده چې لیکل شوی پروګرام وازمویل شي(Test) ، کمپایل (Compile) او اجرا (Execute) شي .د کمپایل پر وخت ځینې تېروتنې راڅرګندېږي چې د Compilerله خوا معلومېږي .دا ګرامري تېروتنې د Syntax errors په نوم یادېږي.

د پروګرام د سم کار کولو لپاره باید دا تېروتنې وپېژندل شي او اصلاح شي.

د اجرا پر وخت هم ځینې تېروتنې رامنځته کېدای شي چې د Runtime errors په نوم یادېږي .لکه

که یو عدد په صفر تقسیم شي، داسي پیغام ښودل کیږي" :هیڅ عدد په صفر نه شي تقسیم کېدای ."دا ډول تېروتنې د پروګرام په اصطلاح کې Bugs بلل کیږي.(Halterman, 2018)

۱.۷ معنوي تېروتنې(Semantic Errors)

که مور د ریاضیي عبارت A + B = Y واخلو، لیدل کیږي چې A او B سره جمع کیږي او نتیجه په Yکې ذخیره کیږي .خو که دا عبارت داسي ولیکل شيY = B + A ، نو که څه هم محاسبه سم ترسره کیږي، خو نتیجه په Y کې په سمه توګه نه ذخیره کیږي .دا ډول تېروتنه معنوي تېروتنه (Semantic Error)بلل کیږي.(Stroustrup, 2013)

۳ فصل :د تېروتنو ردیابي(Debugging)

دا پروسه د تېروتنو (Bugs) پېژندنه او پاکول دي .ځینې وختونه په پروګرام کې داسي تېروتنې موجودي وي چې د منطقي تېروتنو (Logical Errors) په نوم یادیږي.(Oualline, 1995)

دريم فصل

په دې فصل کې د C++ پروګرام ليکنې ژبه، د پروګرام جوړښت، کليدي دستورونه، ثابتونه او د هغوی ډولونه، متغيرونه او د مختلفو ډيټا ډولونه تر بحث لاندې نيول شوي دي.

د C++ پروګرام ليکنې ژبه د نورو پروګرام ليکنې ژبو په پرتله د لوړې کچې يو لړ برتريوي لري چې عبارت دي له:(Rama, 2011)

- دا يوه قوي او ځواکمنه ژبه ده.
- زياتره د سيستم سافټويرو (System Software) او تطبيقي سافټويرو (Application Software)د جوړولو لپاره کارول کيږي.
- د ډيټا ډولونو (Data Types) او عملګرونو (Operators) زيات ډولونه لري چې د سرعت او کارکردګۍ سبب ګرځي.
- د آزاد پلیټفارم (Platform) ژبه ده او انتقال وړتيا لري .زياتره هغه پروګرامونه چې په C++ کې ليکل کيږي، په ټولو عملياتي سيستمونو کې اجرا کېدای شي.
- له شيۍګرا (Object Oriented) ژبو څخه ده.

۱.۹ د C++ ژبې تطبيق(Application of C++)

C++ژبه زياتره د سيستمونو او تطبيقي پروګرامونو د پرمختګ لپاره کارول کيږي، لکه (Rama, 2011):

1. کمپايلرونه(Compilers)
2. تفسير کوونکي(Interpreters)
3. عملياتي سيستمونه(Operating Systems)

4. د ډیټابېس مدیریت سیستمونه(DBMS)

5. د MS Word پروګرام

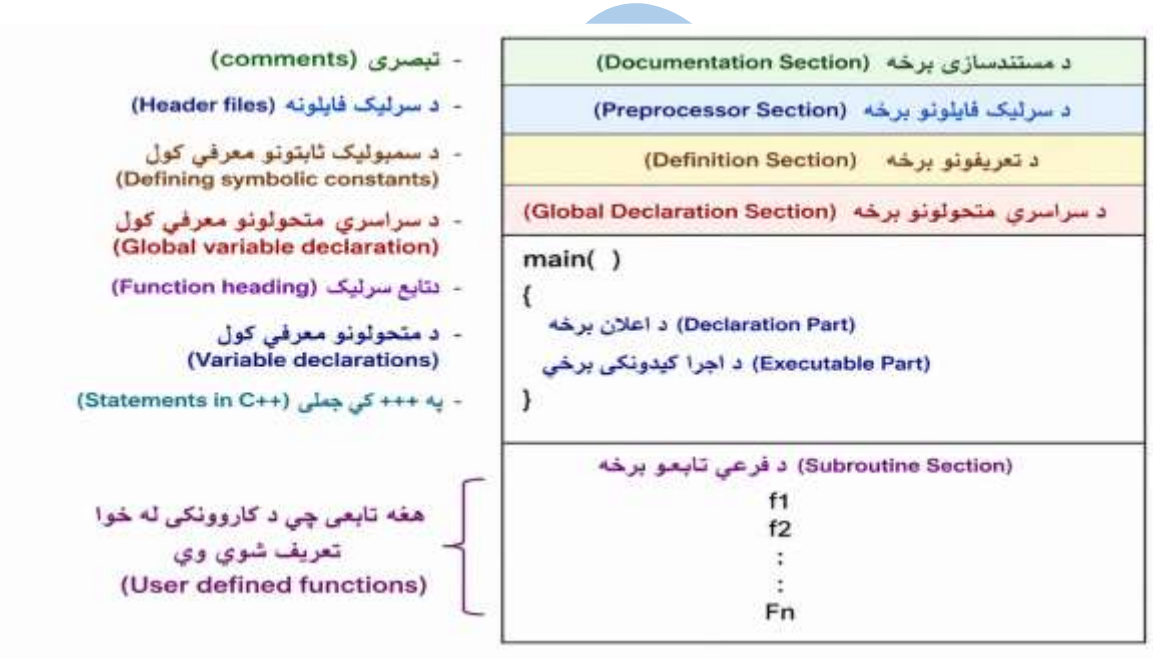
6. د MS Excel پروګرام

7. ګرافیکي پروګرامونه(Graphics Packages)

8. ساینسي او انجینري پروګرامونه

۱.۱۰د C++ د پروګرام ابتدايي جوړښت

يو C ++ پروګرام په څو برخو وېشل شوی دی چې په لاندې ډول تشریح کېږي (2011, Rama) :



I . د مستندسازۍ برخه (Documentation Section)

دا برخه د هغو ګټورو تبصرو مجموعه ده چې د کارووڼکي د لارښوونې لپاره کارول کېږي .که څه هم د تبصرو لیکل اجباري نه دي، خو د پروګرام د مفاهیمو، موخو او جوړښت په پوهېدو کې د کارووڼکو سره مرسته کوي.

II . د تعریف او پرې‌پروسېسر برخه(Preprocessor and Defination Section)

دا برخه هغه فایلونه او لارښوونې لري چې د # نښې سره شروع کېږي او په پای کې h. لري .دي ته د پرې‌پروسېسر لارښوونې (Preprocessor Directives) ویل کېږي چې دC++ د سر فایلونه پکې شامل وي.

د Preprocessor د بیانیو څو مثالونه:

```
#include <iostream.h>
#include <math.h>
```

مثال :ثابتونه (Symbolic Constants) د #define په وسیله تعریفېږي:

```
#define PI 3.14159
#define e 2.71826
```

III . د عمومي متغیرونو برخه(Global Declaration Section)

په دې برخه کې هغه متغیرونه (Variables) معرفي کېږي چې په له یو څخه زیاتو توابعو کې کارول کېږي .دي ډول متغیرونو ته عمومي متغیرونه (Global Variables) ویل کېږي او باید د Headerفایلونو لاندې معرفي شي .یو پروګرام کې کېدای شي څو تابعي موجودي وي .د عمومي متغیرونو د معرفي ګټه دا ده چې تاسو کولی شئ هغوی په هر تابع کې پرته له بیا معرفي کولو څخه وکاروئ.

IV . د main تابع(main Function)

ټول C++ پروګرامونه بايد د main تابع ولري .يادونه کېږي چې په هر پروګرام کې يوازې يو mainتابع موجود وي.

V . قوسونه(Braces)

ټول C++ پروګرامونه د لويو قوسونو څخه جوړ وي .د پروګرام اجرا د main تابع د پرانيستي قوس څخه شروع کېږي او د همدې تابع په ټرلي قوس پای ته رسېږي.

VI . د تعريف برخه(Declaration Part)

په دې برخه کې ټول متغيرونه، توابع، صفونه (Array) او نور معرفي کېږي .همدارنګه ابتدايي ارزښتونه (Initialization) هم په همدې برخه کې ټاکل کېږي.

VII. د اجرا برخه(Execution Part)

دا برخه د اجرا کېدونکو جملو مجموعه ده، لکه :د داخلولو او خارجولو جملې (Input/Output Statements)، حسابي جملې(Arithmetic Statements) ، شرطي او حلقوي جملې (Control Statements)او نور .په دې برخه کې تبصرې (Comments) هم شاملېدای شي . لکه څنګه چې مخکې وويل شول، کمپايلر تبصرو ته پام نه کوي او هغه د اجرا وړ نه ګڼي .د تعريف او اجرا برخې بايد د لويو قوسونو ترمنځ وي .هره اجرايي جمله د سيمي کولن (;) په وسيله ختمېږي (Balagtas, 2006).

VIII . د فرعي توابعو برخه(Subroutine Section)

دا برخه اختياري ده او هغه توابع پکې شامل وي چې د پروګرام ليکونکي له خوا جوړ شوي وي (User Defined Functions)او د main تابع څخه اجرا کېږي.

۲۳ مثال ۳.۱

```
/* Program to print a message */
#include <iostream.h>
main ( )
```

د لومړی سطر د مستندسازۍ (Documentation) برخې پورې اړه لري .دا برخه د تبصرو مجموعه ده چې د پروګرام د نوم او د کوډ د تشریح لپاره کارول کېږي .دوهم سطر د پرې‌پروسېسر بیانیه (Preprocessor Directive) ده چې د کتابتون سره اړیکه جوړوي او د cout دستور ته مرسته ورکوي تر څو د پروګرام نتیجه په سکرین ښکاره کړي .درېیم سطر د اصلي تابع (main) پورې اړه لري .ټول C++ پروګرامونه باید یو اصلي تابع ولري.

پاتې برخه د اجرایي برخې (Execution Part) پورې اړه لري چې پکې اجرایي بیانیې شاملې دي . د تعریف برخه (Declaration) او اجرایي برخه باید د قوسونو {} ترمنځ وي .هره بیانیه د سیمي کولن (;) په وسیله ختمېږي .پروګرام د پرانیستي قوس څخه شروع او د تړلي قوس په وسیله پای ته رسېږي.(Nawaz, 2006)

په C++ کې تبصرې(Comments in C++)

په C++ کې دوه ډوله تبصرې شتون لري:

يو سطري تبصره (Single-line Comment) : کله چې غواړئ په خپل پروګرام کې يو سطري تبصره وليکئ، نو د Single-line comment څخه استفاده کېږي .دا تبصره تل د دوه سلشونو (//) سره شروع کېږي.

مثال:

```
// This is my first program in C++
```

تبصره څو سطري (Multiline Comments)

کله چې غواړئ په خپل پروګرام کې څو سطري تبصره وليکئ، نو د multiline comments څخه استفاده کولی شئ .دا ډول تبصره تل د */ په وسيله شروع کېږي او د /* په وسيله ختمېږي.

مثال:

طور مثال:

```
/* This is
my first
program in C++ */
```

دا بايد يادونه وشي چې تبصرې د اجرا وړ نه دي (Non-executable) او د کمپايلر له خوا له پامه غورځول کېږي (Ignore) کېږي .(يو پروګرامر کولی شي په خپل پروګرام کې څو تبصرې وکاروي .د تبصرو هدف دا دی چې د کوډ معنا او تشريح نورو کسانو ته واضح کړي .هڅه بايد وشي چې تبصرې لنډې او واضح وي.

Output Operator (cout)د

لاندې بيانيه ته وګورئ:

```
cout << "Programming in C++ is fun";
```

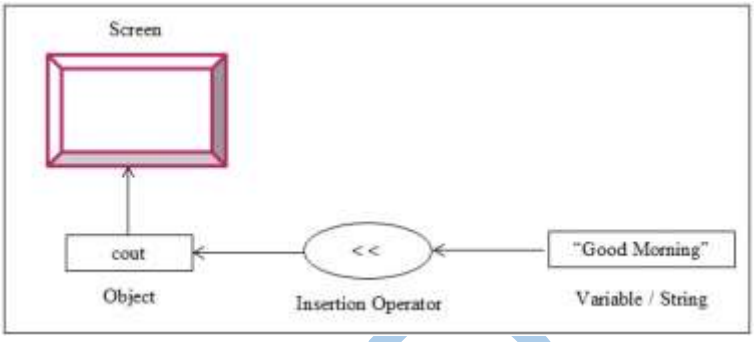
په دې کې cout د دې لپاره کارول کېږي چې د پروګرام نتيجه د ښودلو برخې (screen/output device)ته واستوي .cout د console output مخفف دی.

coutداسې تلفظ کېږي) :سي اوټ (او دا يو مخکې تعريف شوی (pre-defined) عملګر دی چې

د نتايجو د ښودلو لپاره کارول کيږي.(Nawaz, 2006)

په عمومي ډول، هغه معلومات چې cout ته ورکول کيږي د حروفو په سلسله بدليږي.

عملګر >> د insertion operator په نوم يادیږي، چې خپل ښي لوري معلومات خپل چپ لوري objectته استوي.



شکل 3.2 په کارولو سره خروجی cout

مثال ۳.۲لاندی پروګرام د cout په کارولو سره حروفي قیمت پر سکرین ښکاره کوی

```
#include <iostream.h>
#include <string> // used for string data type main ( )
{
  // Declaration    string First_Name = "Habibullah ";    string Last_Name  =
"Barakzai ";    cout << "The Complete Name is : " << First_Name << Last_Name;
return 0;
  }
```

Output

The Complete Name is : Habibullah Barakzai

داخلولو عملګر (Input Operator-cin)

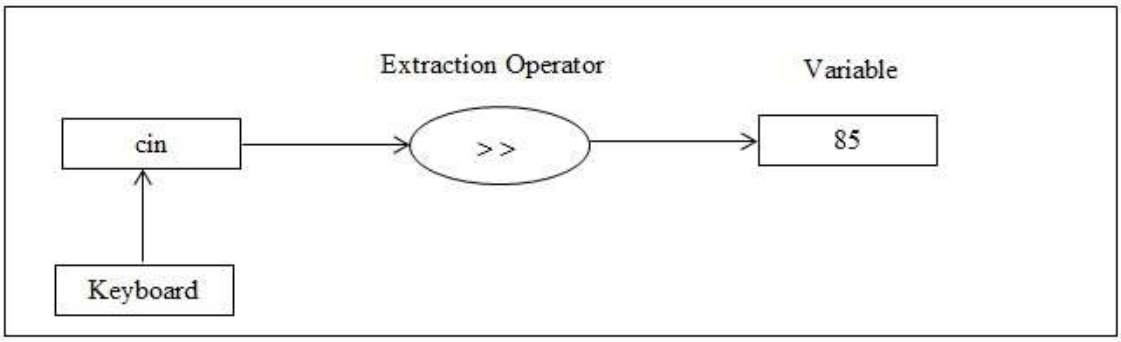
لاندې بیانیه ته وګورئ:

```
cin >> marks;
```

په C++ ژبه کې cin د دې لپاره کارول کیږي چې د کیبورډ له لارې معلومات داخل کړي cin .د Console Inputمخفف دی .دا عملګر پروګرام دې ته اړ باسي چې انتظار وکړي تر څو کارووونکی د کیبورډ څخه د marks ارزښت داخل کړي .وروسته داخل شوی ارزښت د marks په متغیر کې ذخیره کیږي.

cinداسې تلفظ کیږي سې ان او دا یو مخکې تعریف شوی (pre-defined) عملګر دی چې د کیبورډ څخه د معلوماتو د اخیستلو لپاره کارول کیږي.(Rama, 2012)

عملګر >> د extraction یاget from په نوم یادیږي، چې د کیبورډ څخه معلومات اخلي او په ښې لوري متغیر کې یې ذخیره کوي.



شکل ۳.۳ د cin په وسیله ورودي

مثال لاندې پروګرام د یو کس نوم، تخلص او عمر د کیبورډ څخه اخلي او په سکرین یې بنیي3. 3

```
#include <iostream.h>
#include <string> // used for string data type main ( )
{
  // Declaration
string FName;   string
LName;   int Age;
  cout <<" Enter your first and second name and your age please : ";   cin >>
FName >> LName >> Age;   cout << "Your full name is :" << FName << "  " <<
LName << "\n";   cout << "You are " << Age << " years old";   return 0;
  }
```

Output

Enter your first and second name and your age please : Asma Gul 8 Your full name is : Asma Gul

You are 8 years old

د داخلولو او خارجولو عملګرو تکرار(CASCADING OF I / O OPERATORS)

کله چې د >> او << عملګرونه په یوه بیانیه کې څو ځله وکارول شي، دې حالت ته د عملګرو تکرار (Cascading) ویل کېږي.

د خروجی عملګر >> مثال:

```
cout << " Simple Interest = " << SI << "\n";
```

په همدې ډول د داخلولو عملګر >> هم داسې تکرار کېدای شي:

```
cin >> Prin >> Rate >> Year;
```

په دې دستور کې معلومات له چپ څخه ښي لوري ته په متغیرونو کې ذخیره کېږي .یعني لومړی ارزښت پهPrin ، دوهم په Rate او دریم په Year کې ذخیره کېږي .باید د >> او متغیرونو ترمنځ مناسب فاصله(space)

ورکړل شي تر څو نتیجه واضح وي.

مثال(3.4) پروګرام د قائم الزاويه مثلث مساحت محاسبه کوي:

```
// Program finds area of triangle
 #include <iostream.h>
  main ( )
{
     double Base, Hight, Area;
     cout << " Enter value for Base and Hight of a triangle: " ;       cin
>>Base >> Hight;          Area = Base * Hight / 2;
     cout << " The area of Triangle = " << Area;
{
```

Output

Enter value for Base and Hight of a triangle : 10 5 The area of Triangle = 25

د C++ د حروفو مجموعه(C++ Character Set)

د حروفو مجموعه هغه ټول سمبولونه دي چې د معلوماتو د وړاندې کولو لپاره کارول کيږي، لکه الفبا، ارقام او ځانګړي سمبولونه.(Rama, 2011)

جدول ۳.۱ د C++ د حروفو مجموعه

الفبا(Alphabets)	پکې لوی حروف AترZ او کوچني حروف aترz شامل دي.ارقام(Digits)
ځانګړي سمبولونه (Special Symbols)	پکې 0 تر9 ارقام شامل دي

tilde ~ back quote ` exclation ! hash # percentage % caret ^ ampersand & double quote “ single quote (apostrophe) ‘ left parantheses (right parantheses) left braces { right braces } left bracket [right bracket] plus sign + minus sign - asterisk * slash / underscore _ equal sign = backslash \ semi colon ; colon : - comma , dot . -	سمبولهایخاص)Special symbols(

	less than < greater than >> question mark ? vertical bar | -
فاصله های خالی **(White spaces)**	(form شامل خالیگاها) spaces، tab، \v, \t) خط جدید (\n و feed

۱.۱۱د C++ اجزا(C++ Tokens)

د C++ یو پروګرام د هغو عناصرو مجموعه ده چې د ټوکن (Token) په نوم یادیږي .ټوکن د پروګرام تر ټولو کوچنی عنصر دی .په C++ کې ببلابیل ډول ټوکنونه شتون لري چې په لاندې جدول کې ښودل شوي دي .یو ټوکن کېدای شي د یو یا څو C++ حروفو څخه جوړ وي.

جدول ۳.۲ ټوکنونه

ټوکنونه	د کارونې موارد
دستوري کلیدي(Keywords)	لکه: do، int، for او نور
شناسه(Identifiers)	لکه د متغیرونو نومونه: sum، area، number او نور
ثابتونه(Constants)	لکه: 250، 5.25، 200- او نور
سلسله حروف(Strings)	لکه: "College"، "2+3" او نور
عملګرونه(Operators)	لکه: +، -، *، /، ++ او نور
ځانګړي سمبولونه (Special Symbols)	لکه: !، #، [،]، {، } او نور

۱.۱۱.۱دستوري کليدي(Keywords)

Keywordsهغه کلمي دي چې د ځانګړو موخو لپاره کارول کېږي .دا د C++ مخکې نه تعريف شوي (reserved) کلمات دي چې ځانګړې معنا لري او يوازې د ټاکلو موخو لپاره کارول کېږي .ټول Keywords بايد په کوچنيو حروفو وليکل شي .دا کلمات د متغيرونو د نوم په توګه نه شي کارول کېدای.(Buard, 2005)

د C++ د کليدي دستورو (Keywords) معياري لست په لاندې ډول دی:

Asm	Double	Not	Switch
Auto	else	operator	Termplate
break	enum	private	this
case	extern	protected	throw try
catch	float	public	typedef
char	for	register	union
class	friend	return	unsigned
const	goto	short	virtual
continue	if	signed	void
default	inline	sizeof	volatile
delete	int	static	while
do	long	struct	

۱.۱۱.۲ شناسه(Identifiers)

• شناسه هغه نوم ته ویل کېږي چې د متغیرونو(variables) ، توابعو (functions) او صفونو (array) لپاره په پروګرام کې کارول کېږي.

• شناسه د کرکټرونو یوه مجموعه ده چې د حروفو(A–Z) ،(a–z) ، ارقامو (0–9) او underscore (_) څخه جوړه وي.

• یو C++ Case Sensitive ژبه ده، نو ALFA ، Alfa او alfa له یو بل سره توپیر لري.

• د underscore (_) سمبول عموماً د نوم په منځ کې کارول کېږي، لکه Name_F :

• د Identifier د نوم اوږدوالی عموماً د 8 تر 10 کرکټرونو پورې وي، خو په اصل کې کوم محدودیت نه لري او اوږد هم کېدای شي.

• Identifier باید په حرف شروع شي، وروسته کولی شي حروف، ارقام یا دواړه ولري .دا له عدد څخه نه شي شروع کېدای.

• په C++ کې کلیدي کلمې (Keywords) ځانګړي مانا لري او د نورو موخو لپاره نه شي کارول کېدای، لکه int چې د متغیر نوم په توګه نه شي استعمالېدای.

مجاز شناسه‌ګان (Valid Identifiers) عبارت دي له:

Sum, avg _ marks, x1, y2, PINCODE, Total

غیر مجاز شناسه‌ګان (Invalid Identifiers) عبارت دي له:

10th, S.I, total amount, Std – no

۱.۱۱.۳ ثابتونه(Constants)

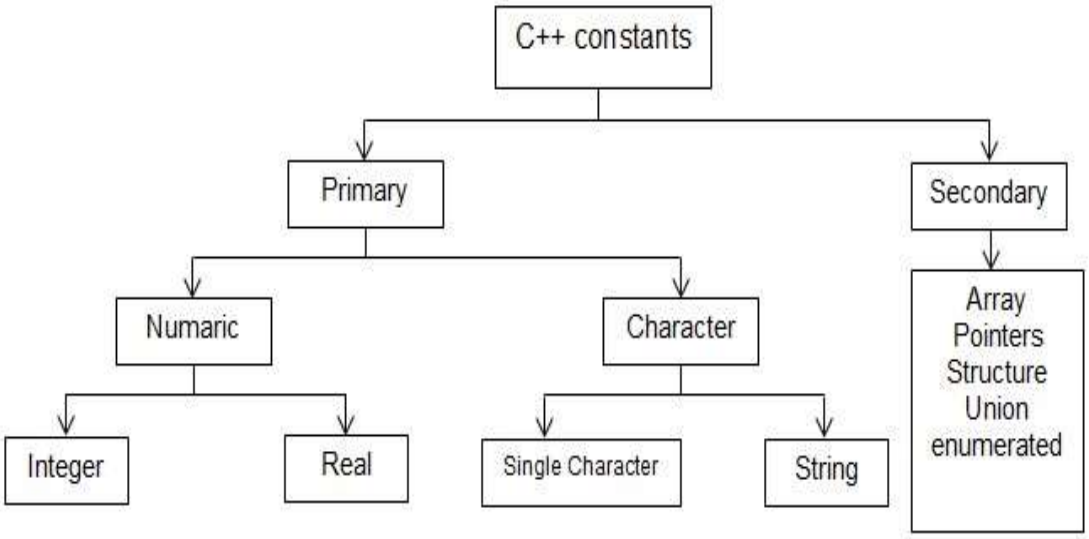
هغه ارزښت ته چې د پروګرام د اجرا پر مهال نه بدلېږي، ثابت (Constant) ویل کېږي .د C ++ ژبه څو ډوله ثابتونه لري.

د ثابتونو ډولونه(Types of Constants)

په عمومي ډول د C++ ثابتونه په دوو برخو وېشل کېږي:

1. ابتدايي ثابتونه
2. ثانوي ثابتونه

دا ثابتونه بيا په نورو ډولونو هم وېشل شوي دي لکه لاندې:



شکل ۳.۴ د ثابتونو ډولون

ټول عددی ثابتونه(Integer Constants)

عددي ثابتونه هغه ثابتونه دي چې له عددونو څخه جوړ وي .په عموم کې د integer ثابتونو درې ډولونه شتون لري:

- قاعده ۱۰ (Decimal system)

قاعده ۸ (Octal system)

قاعده ۱۶ (Hexa Decimal system)

جدول ۳.۳ د تامو ثابتونو ځانګړتياوې

د ۱۰ قاعدي تام	د ۸ قاعدي تام	د ۱۶ قاعدي تام
حاوی ارقام (0 تر 9)	حاوی ارقام (0 تر 7)	حاوی ارقام (0 تر 9)او حروف (A تر F)چې د 10 تر 15 ښيي
حداقل بايد يو رقم ولري	لومړی رقم بايد 0 وي	بايد د 0x يا 0X سره شروع شي
اعشاري نښه نه لري	اعشاري نښه نه لري	اعشاري نښه نه لري
کولی شي مثبت يا منفي وي	کولی شي مثبت يا منفي وي	کولی شي مثبت يا منفي وي
د ارقامو ترمنځ کامه او فاصله نشته	د ارقامو ترمنځ کامه او فاصله نشته	د ارقامو ترمنځ کامه او فاصله نشته
مثال 426:،+17 ، 5050 جائز 10.0، 45.12 ناجائز	مثال 051،+044 ، 025 جائز 540،45.068 ، 03 ناجائز	مثال 0xA1:،0x38 ، 0X1 جائز 0xEGG،0ABC ، 0x2.5 ناجائز

ثابتونه اعشاري(Real Constants)

هغه عددي ارزښت چې اعشاري برخه ولري، اعشاري ثابت (Real Constant) بلل کېږي .د دې ځانګړتياوې:

- لږ تر لږه بايد يو رقم ولري
- اعشاري نښه ولري
- مثبت يا منفي کېدای شي
- د عددونو ترمنځ کامه او فاصله نشته

مثال ۳.۶

6.3مثال
2.38 ,-6789.124 ,54.0 ,523.66+ از جمله ثابتهای مجاز اعشاری است **در حالیکه** 10 -5E ,035 ,11 از جمله ثابتهای حیر مجاز اعشاری میباشدم

کله چې اعشاري عدد ډېر لوی يا ډېر کوچنی وي، نو په دې حالت کې د علمي طريقې (Scientific Notation)څخه استفاده کېږي، چې د exponential طريقې په نوم هم يادېږي .په دې طريقه کې اعشاري عدد په دوو برخو ښودل کېږي:

هغه ارزښت چې د (e) څخه مخکې راځي د مانتيس (Mantissa) په نوم يادېږي، او هغه برخه چې د (e) څخه وروسته راځي د توان يا نما (Exponent) په نوم يادېږي.

مثال:

اعشاري عدد 516.55 ته په علمي طريقه کې داسې ليکل کېدای شي:

5.1655e2

دلته e د 10 په معنا ده، نو 2 د توان (Exponent) په توګه کارول شوی دی.

هغه قواعد چې د علمي طريقې لپاره بايد په پام کې ونيول شي:

- د مانتيس او توان برخه بايد د e په وسيله له يو بل څخه جلا شي.
- د مانتيس برخه کېدای شي مثبت يا منفي وي.
- د توان برخه بايد لږ تر لږه يو عدد ولري او دا عدد مثبت يا منفي کېدای شي.
- د علمي طريقې د اعشاري عددونو ساحه عموماً د 3.4e-38 تر 3.4e38 پورې وي.

7.3مثال
5 - 2.4E + 3,- 0.5e + 4.5e8,- 3.2e – 5+ از جمله ثابت های نمادار Exponentials constant مجاز در حالیکه 5.5 – 4 .2e ,3E06 - ثابت های نمادار حیر مجاز می باشدم

د حرفي ثابتونه(Character Constants)

- حرفي ثابت هغه يو حرف وي چې د يو واحد (single quote)ترمنځ ليکل کېږي.
- يو حرفي ثابت له يو څخه زيات حروف نه شي لرلی.
- پر حرفي ثابتونو حسابي عمليات کېدای شي، ځکه چې دوی د عددي ارزښت په بڼه ذخيره کېږي.
- په C++ کې escape sequences لکه\n، \t، \b هم پېژندل کېږي.

مثال 3.8
‘=’ , ‘|’ , ‘a’ , ‘\o’ از جمله ثابتهای مجاز حرفی میباشد در حالیکه ‘62’ , ‘a4’ , ‘xy’ از جمله ثابتهای حیر مجاز حرفی میباشدم

پاملرنه وشي چې حرفي ثابت ‘2’ د عدد 2 سره يو شان نه دی .هر حرف يا کرکټر يو ثابت کوډ لري چې د ASCII code په نوم يادېږي.

لاندې جدول د ځينو حرفي ثابتونو او د هغوی د ASCII کوډونه بنيي:

ثابتونه	د ASCII کوډ
‘A’	65
‘B’	66
‘Z’	90
‘a’	97
‘z’	122
‘1’	49
‘2’	50
‘#’	35

حرفي ثابتونه(String Constants)

حرفي ثابتونه هغه د حروفو مجموعه ده چې له يو څخه زيات حروف لري او د دوه ګونو quotation marks (" ")په منځ کې ليکل کيږي .دا حروف کېدای شي الفبا، اعداد، ځانګړي سمبولونه يا خالي ځای هم وي.

مثال 3.9

" " , "", "AFN .20", "Phone : 334 – 5670", "25Read" له اجازی ورکړل شوو ثابتونو څخه يو هم حرفي ثابتونه دی په داسی حال کې 'The Error', "Line1 \n \ Line2 \n Line3"له اجازه شوو ثابتونو څخه يو هم حرفي ثابت ":" د خالي حرفي ثابت په نوم null string يادیږی

پام وکړئ چې 'A' او "A" يو شان نه دي .حرفي ثابت چې د دوه ګونو quotation marks (" ")کې وي، په اصل کې عددي ASCII ارزښت نه لري.

۴ سمبولیک ثابتونه یا ماکرو(Symbolic Constants / Macro)

#define د preprocessor له لارښوونو څخه دی چې د ثابتونو د تعریف لپاره کارول کیږي (Soulié, 2007).

#define name replacement-string

په دې ځای کې (name) یو متحول دی چې کولی شي عددي یا حروفي ثابت قیمت په خپل ځان کې وساتي .هر کله چې پروګرام اجرا کیږي، هر هغه ځای کې چې د متحول (name) نوم موجود وي، هغه قیمت چې د پروګرام په پیل کې دې متحول ته ورکړل شوی وي، ورسره بدلیږي .د یادونې وړ ده چې دا متحول نشي کولی له هغه قیمت څخه بل مختلف قیمت وساتي کوم چې د پروګرام په پیل کې ورته ټاکل شوی وي .په اصل کې #define متحول name مجبوروي چې یوازې هغه قیمت وساتي چې د پروګرام په پیل کې د کارووونکي له خوا ورته تعریف شوی وي او بل هیڅ قیمت نه شي اخیستلی. ځینې مثالونه د سمبولیک ثابتها:

```
#define   Max_Marks   100
#define   PI          3.14159
#define   University  “Kabul Education University”
```

مثال10.3 لاندې پروګرام د یوې دایرې شعاع پیدا کوي په هغه صورت کې چې مساحت ورکړل شوی وي:

```
// Program to find redius of a circle given area
#include <iostream.h>
# define  PI   3.14159
      main ( )
    {
         float  r, area ;            cout << "\n " << " Enter
the area of circle : " ;            cin >> area ;            r =
sqrt ( area / PI ) ;
          cout << "\n" <<  "The radius of the given circle is : " << r ;
return 0 ;
   }
```

Output

```
Enter the area of the circle : 12 . 57
The radius of the given circle is : 2. 0029
Enter the area of the circle : 19 . 3
The radius of the given circle is : 2.47
```

هغه قواعد چې د سمبولیک ثابتونو د معرفي کولو پر مهال باید په نظر کې ونیول شي:

- سمبولیک ثابتونه معمولاً د پروګرام په شروع برخه کې تعریف کېږي، خو کاروونکی کولای شي چې د پروګرام په هره برخه کې د استعمال نه مخکې هغه معرفي کړي.
- #define د سیمي کولن سره نه ختمېږي .که چېرې په اخر کې سیمي کولن اضافه شي، مفسر (compiler)به هغه د ثابت د برخې په توګه حساب کړي.

• د # او define ترمنځ د سیمي کولن کارول اجازه نه لري.

• د #define او د سمبولیک نوم ترمنځ باید د یو یا دوو سپیسونو فاصله وي .همدارنګه د سمبولیک نوم او د هغه د قیمت ترمنځ هم فاصله باید په نظر کې ونیول شي.

• د سمبولیک نوم تعریف کول باید هماغه قواعد تعقیب کړي چې د یو متغیر (variable) د نوم لپاره کارول کیږي .معمولاً سمبولیک نومونه په لویو تورو لیکل کیږي تر څو له عادي متغیرونو څخه جلا وپیژندل شي.

• تاسو نشئ کولای چې د پروګرام په منځ کې د سمبولیک نوم لپاره د پیل نه ټاکل شوی قیمت بدل کړئ.

• کله چې پروګرام اجرا کیږي، هر ځای چې سمبولیک نوم وي، خپل قیمت په اوتومات ډول له هغه ځایه اخلي چې تعریف شوی وي.

د بېلګې په توګه:

```
#define PI 3.14159

Curricum = 2 * PI * Radius ;
.....................
Cout <<" \n PI value =  " << PI ;
```

کله چې پروګرام اجرا شي، سمبولیک نوم PI په خپل قیمت بدلیږي او جمله دا بڼه اخلي:

```
Curricum = 2 * 3.14159 * Radius ;
```

فکر وکړئ که ماکرو په لاندې ډول تعریف شوی وي:

```
#define PI 3.14159 ;
```

یعني که د ماکرو د نوم په اخر کې سیمي کولن اضافه شي، نو جمله به په لاندې ډول شي:

```
Curricum = 2 * 3.14159 ; * Radius;
```

دا کار سبب کيږي چې پروګرام په سمه توګه اجرا نه شي.

د ماکرو ځيني ناسم مثالونه په لاندې ډول دي:

```
# define SUM = 0
# define TOTAL 100 ;
# defineA 10 , B 20
# DEFINE COUNT 10
```

• که چيرې يوه جمله يا اوږد حروفي قيمت په #define کې داسې تعريف شي چې څو کرښې ولري، نو اړينه ده چې د هرې جملې په اخر کې (back slash) \ واچول شي او پاتې جمله په نوې کرښه کې وليکل شي .د \ ليکلو هدف دا دی چې مفسر (compiler) ته وښيي چې جمله لا دوام لري.

بېلګه 11.3 :لاندې پروګرام د دايرې شعاع پيدا کوي کله چې د هغې محيط ورکړل شوی وي .

```
// Program to find radius of a circle given the circumference  #
include <iostream.h>  # define  PI    3.14159
      main ( )
     {
          float  r, circum ;            cout << "\n " << " Enter the circumference of
a circle  : " ;            cin >> circum ;            r = circum / ( 2 * PI ) ;
cout << "\n" <<  "The radius of the given circle is : " << r ;            return 0 ;
    }
```

```
Output
    Enter the circumference of the circle :  8
    The radius of the given circle is :      1 . 27
```

د ماکرو ګټې

- د ماکرو کارول پروګرامونه ساده او منظم کوي.
- د ماکرو نوم بايد مانادار وي او کاروونکي سره مرسته وکړي چې په شي کوم ډول قيمت پکې ساتل شوی دی.
- د ماکرو کارول پروګرام کې بدلون راوستل اسانه کوي .دا اسانه ده چې د ماکرو قيمت بدل شي، پر ځای د دې چې په هر ځای کې د متغير قيمت بدل شي.
- د ماکرو کارول پروګرام لوستل اسانه کوي.

۱.۱۲ مفهوم Escape Sequences

د (backslash) / څخه وروسته يو ځانګړی حرف راځي او يو ځانګړی عمل ترسره کوي .د دې ډول ترکيب ته escape sequence ويل کيږي.(Deitel, 2012)

د بېلګې په توګه:

```
cout <<   " \n this is easy !!"   ;
```

دا جمله په نظر کې ونيسئ.

په دې جمله کې \n د escape sequence بېلګه ده او د دې لپاره کارول کيږي چې کرسر نوې کرښې ته لاړ شي او وروسته جمله this is easy په سکرين وښيي .که وغواړئ چې يو حروفي متن په څو کرښو کې چاپ کړئ، نو \n وکاروئ تر څو هغه متن چې د کرسر وروسته راځي په نوې کرښه کې چاپ شي .که د متن يا جملې په اخر کې \n نه وي، نو بله جمله به هم د مخکنۍ جملې په دوام په هماغه کرښه کې ښکاره شي.

بېلګه 12.3 :لاندې پروګرام د \n کارول ښيي.

main ( )
{
cout << “This \n string \n will \n be \n printed \n in \n 8 \n Lines. “; return 0 ;
}

Output This String
will **be** **printed** **in** **8** **Line**

که که وغواړئ د اوږدې جملې یوه برخه په بلې کرښه کې ښکاره شي، نو \n د هماغې مطلوبې برخې په پیل کې ولیکئ .په یاد ولرئ چې ټول escape sequence ګانې د \ څخه پیل کیږي .نور escape sequenceګانې چې په C++ کې شته، په لاندې ډول دي:

جدول 4.3 د escape sequence کرکټرونو تشریح

escape sequence	وظیفه
\a	برنامه د اجراپر مهال غږ کوی
?\	د سوال نښه ده
\b	د یوSpaceشاته تګ
\r	د output د پیل ته بیرته تګ.
\f	Form feed
\t	د tab په اندازه ټوپ وهل
\v	عمودی tab
0\	خالي کرکټر Ascll
\n	نوی کرښي ته تګ
”\	نښه doubie quotation
\\	Backslash
0\	خالي
’\	قوس نا خنک یګانه (single (quotation

۱.۱۳ متحولونه (Variables)

متحول یو نوم دی چې د RAM حافظې یوې برخې ته ورکول کیږي او هماغه برخه د ځان لپاره ځانګړې (Reserve) کوي .کله چې متغیر ته قیمت ورکړل شي، نو په هماغه ځانګړې شوې حافظه کې په موقتي ډول ذخیره کیږي.(Klousen, 2017)

متغیر یوازې یو قیمت په یوه وخت کې په ځان کې ساتلی شي .یعنې هر نوی قیمت چې په متغیر کې ذخیره شي، د مخکینې قیمت ځای نیسي .دا قیمت کېدای شي عددي یا حروفي وي .متغیر کولی شي د پروګرام د هر اجرا په وخت کې مختلف قیمتونه په ځان کې وساتي.

بېلګه 13.3

لاندې د C++ پروګرام برخه په نظر کې

```
:       int x, y, z
:       char p;
X = 1,                  /* value 1 is assigned to x */
Y = 2;                  /* value 2 is assigned to Y */
Z = 3;                  /* value 3 is assigned to Z */
X = Y + Z;              /* value X changes to 5 */
P = 'A'                 /* value 65 is assigned to P */
Y = 6;                  /* value of Y changes to 6 */
Z = 7;                  /* value of Z changes to 7 */
P = 'a';                /* value of P changes to 97 */
```

له پورته بېلګې څخه څرګندېږي چې یو متغیر کولی شي په یوه پروګرام کې مختلف قیمتونه ذخیره کړي.

په C++ کې د متغیر نومولو قواعد

- د متغیر نوم باید د حرف څخه پیل شي او وروسته تري حروف، اعداد یا د دواړو ترکیب راځي.
- د متغیر د نوم په جوړښت کې د (_) underscore پرته بل سمبول کارول اجازه نه لري.
- د C++ ژبه case sensitive ده، له همدې امله counter ، Counter او COUTNER درې جلا متغیرونه دي.
- د C++ کلیدي کلمې د متغیر د نوم په توګه نشي کارېدلی.
- د متغیر د نوم د حروفو ترمنځ space ورکول اجازه نه لري.
- ښه ده چې د متغیر نوم لنډ او مانادار وي.

بېلګه 14.3

total_Sum, avg _ height د مجاز متحولونو له جملې څخه دي، په داسې حال کې No#, th50, Lotus 123 د نامجاز متحولونو له جملې څخه ګڼل کېږي.

د ډيټا ډولونه(Data Types in C++)

په C++ ژبه کې څلور ډوله ډيټا شته چې عبارت دي له:(Backman 2012)

- بشپړ عدد يا Integer (int)
- حرف يا character (char)
- اعشاري د ساده دقت سره floating (float)
- اعشاري د دوه چنده دقت سره(double)

هر data type ځانګړي خصوصيات او ظرفيت لري چې په لاندې جدول کې تشريح شوي دي.

جدول 5.3 د ډيټا ډولونو تشريح

د ډيټا ډول (Data Type)	تشريح	اندازه په بايټ	ساحه تعريف د
Int	يوازې بشپړ اعداد په ځان کې ذخيره کوي	2	32767- الی 32768
Char	يوازې يو حروف په ځان کې ذخيره کوی	1	127- الی 128
Float	اعشاري اعداد د ساده دقت سره چې تر 7 رقمو پوری د اعشاري وروسته ښيي په ځان کی ذخيره کوی	4	3.4E-38 الی 3.4E+38
Double	اعشاري اعداد د دوه چنده دقت سره تر 15رقمو پوري د اعشاري ورسته ښيي ،په ځان کې ذخيزه کوي	8	1.7E-308 الی 1.7E+308

د بشپړو اعدادو ډولونه(Integer Data Types)

دا د ډيټا ډول (int) يوازې بشپړ اعداد لري او اعشاري اعداد نه ساتي.

د integer مختلف ډولونه په لاندې ډول دي:

a . بې نښې بشپړ عدد (unsigned int)

هغه متغير چې د(unsigned int) په ډول تعريف شي، په حقيقت کې خپل ټول بيتونه د مثبتو بشپړو اعدادو د ذخيره کولو لپاره کاروي او هېڅ بيت د نښې (مثبت يا منفي) د معلومولو لپاره نه ځانګړی کوي.

b . کوچنی نښه لرونکی بشپړ عدد(short int)

دا د ډيټا ډول د کوچنيو بشپړو اعدادو د ذخيره کولو لپاره کارول کېږي.

c . کوچنی بې نښې بشپړ عدد(unsigned short int)

کله چې متغير د (int short unsigned) په ډول تعريف شي، معنا يې دا ده چې يوازې مثبت کوچني بشپړ اعداد په ځان کې ساتلی شي.

d . لوی بشپړ عدد(long int)

دا د ډيټا ډول د لويو بشپړو اعدادو لپاره کارول کېږي.

e . لوی بې نښې بشپړ عدد(unsigned long int)

په دې حالت کې د نښې بيت له منځه ځي او حافظه دوه چنده کېږي، او کولی شي لوی مثبت بشپړ اعداد په ځان کې ذخيره کړي.

لاندې جدول د ډيټا ډولونه(data types) ، د اندازې ظرفيت (bytes in size) او د بشپړو اعدادو د تعريف ساحه (range) ښيي:

جدول 6.3 د ډيټا ډولونو تشريح

ساحه تعريف د (Range)	اندازه په بايټ (Size in bytes)	د ډيټا ډول (Data Type)
0 تر 65535	2	**کوچنی بي نښي بشپړ عدد** unsigned short
32768 تر - 32767 +	2	**کوچنی نښه لرونکي بشپړ عدد** signed short
0 تر 4294967295	4	**بشپړ عدد** unsigned int
+ 2147483647 - 2147483648	4	**بشپړ عدد** signed int
0الی 40273322.9 81 +E	8	**لوی بشپړ عدد** (long)
0 تر 255	1	(unsigned char) **بی نښي حرفي**

43

د اعشاري ساده دقت ډيټا ډول(Float Point Data Types)

موږ کولی شو د اعشاري ساده دقت ډيټا ډول د float کليدي دستور په کارولو سره تعريف کړو .

floatځلور بايټ حافظه ځانګړې کوي او تر شپږو اعشاري ځايونو پورې په عملياتو کې دقيق وي.

د اعشاري دوه چنده دقت ډيټا ډول(Double Precision Data Types)

دا ډيټا ډول کولی شي لوی اعداد په ډېر لوړ دقت سره محاسبه کړي او نتيجه وړاندې کړي double .
اته بايت حافظه ځانګړې کوي او تر 14 رقمو پورې د اعشاريه وروسته په لوړ دقت سره محاسبه ترسره کوي .د دې ډيټا ډول د تعريف لپاره د double کليدي دستور کارول کېږي.

د حرفي ډيټا ډول(Character Data Type)

دا ډيټا ډول کولی شي يو حرف يا د ASCII يو کرکټر په ځان کې ذخيره کړي .د دې ډيټا ډول ظرفيت يو بايت دی او د char کليدي دستور په وسيله تعريف کېږي .که د نښې بيت په نظر کې ونه نيول شي، دا ډيټا ډول کولی شي د 0 څخه تر 255 پورې عدد په ځان کې وساتي.

۱.۱۴د متغيرونو معرفي کول(Declaring Variables)

د متغيرونو معرفي کول د ډيټا ډول او د متغير نومونه لري چې په اخر کې يې سيمي کولن (;) ليکل کېږي .دا اعلاميه (declaration) په حقيقت کې مفسر (compiler) ته د متغير د نوم او ډول په اړه معلومات ورکوي.

د متغير د معرفي کولو جوړښت په لاندې ډول دی:

```
;data – type a1, a2, a3 , ..... an
```

دلته a1، a2 ، a3 او an د متغيرونو نومونه دي او د data-type په برخه کې له مخکې څخه معرفي شوی يو ډول لکه(int ، float)او نور ليکل کېږي .که له يوه څخه زيات متغيرونه تعريف شي، نو a1، a2 ، a3 د کامې په وسيله له يو بل څخه جلا کېږي.

بېلګه15.3
int x, y, z ; float avg, sum ; char ans ; double amount ;

دلته x ، y او z د int ډول څخه، avg او sum د float ډول څخه، ans د char ډول څخه او amount د double ډول څخه معرفي شوي دي .دغه متغيرونه په لاندې ډول هم معرفي کولی شو:

```
int x;
int y;
int z;
float avg;
float sum;
char ans;
```


د متغيرونو ډولونه په لاندې ډول هم معرفي کولی شو:

```
short int count ;
long int fact ;
```

ځکه چې short int او short يو شان مفهوم لري او long int او long هم يو شان مفهوم لري، نو پورته جملې په لاندې ډول هم ليکل کېدای شي:

```
short count ;
long fact;
```

۱.۱۴.۱ متغیرونو ته د قیمتونو ورکول(Assigning Values to Variables)

تاسو کولی شئ په دري طریقو سره قیمتونه په متغیرونو کې ذخیره کړئ:

1. د اعلان په برخه کې(Declaration part)
2. د پروګرام د اجرا په برخه کې (Executable part) د = نښې په کارولو سره
3. د Keyboard په وسیله

د= Assignment Statement

کله چې غواړو یو قیمت په متغیر کې ذخیره کړو، نو د = نښه کاروو.

د دې عمومي شکل په لاندې ډول دی:

```
Variable_name = Constants / Variable / Expression;
```

بیانیه لاندې په نظر کې ونیسئ:

```
Total = 0;
```

دا بیانیه ښیي چې قیمت 0 چې د = ښي لوري کې قرار لري، د = چپ لوري متغیر (Total) کې ذخیره کیږي .د = په ښي لوري کې کولی شو ثابت قیمت، متغیر او یا داسې عبارت ولیکو چې دواړه (ثابت او متغیر) ولري.

د بېلګې په توګه:

```
Sum = marks 1 + marks 2;
```

په دې فورمول کې د = ښي لوری ارزول کېږي او بیا د marks1 او marks2 مجموع په Sum متغیر کې ذخیره کېږي.

Count = Count + 1

په دې بیانیه کې د Count پخواني قیمت ته 1 اضافه کېږي او نوی نتیجه بیا په هماغه متغیر Countکې ذخیره کېږي .که د Count لومړنی قیمت 10 وي، نو نوی قیمت یې 11 کېږي.

مثال 16.3 د متغیرونو ته قیمت ورکول د اعلان په برخه کې (part Declaration):

```
// Assigning value to variables in the declaration part
        .
        .
        .
       int  x = 1 , y = 2,  z = 3 ;
float  avg = 0.0 , pi ;
char     opt = ‘ y ‘ ;
```

Examples of assigning values to variables in the executable part are :

د اجرایي برخې (executable part) کې متغیرونو ته قیمت ورکولو مثالونه:

```
pi = 3.14159 ;
  total = 0 ;
   opt = ‘N’ ;
```

C++ژبه کارووونکي ته اجازه ورکوي چې ببلابېل operators په یوه بیانیه یا فورمول کې وکاروي .ځینې مجاز عبارتونه په لاندې ډول دي:

```
;A = B = C = 0
; X1 = X2 = Large
```

نوت :د = نښې په چپ لوري کې بايد تل د يو متغير نوم موجود وي.

مثال 17.3 لاندې پروګرام د دوو بشپړو اعدادو (int) مجموع محاسبه کوي:

```
// program to find sum of two integer numbers
#include <iostream.h>
main ( ) {
// Declaration and assignment
int N1 =200, N2 = 300, Sum;
Sum = N1 + N2;
cout << "\n " << " Total = " << Sum;
return 0;
```

Output

```
Total  =  500
```

مثال 18.3 لاندې پروګرام د دايرې محيط محاسبه کوي:

```
// program to find circumference of a cricle
#include <iostream.h>
main ( )
{
// Declaration and assignment
float pi = 3.14150 , radius = 10 , circum ;
circum = 2 * pi * radius ;
cout << "\n " << " Circumference = " << circum ;
return 0 ; }
```

Output

Circumference = 62 . 831802

مثال 19.3 لاندې پروګرام د څلورو مضامینو د نمرو مجموعه او اوسط محاسبه کوي:

```
// Program to find total and average of marks obtained in 4
s ubjects.
#include <iostream.h>
main ( ) {
int m1, m2, m3, m4 ;
float total , avg ;
// assignment in executable part
m1 = 50 , m2 = 70 , m3 = 75 , m4 = 67 ;
total = m1 + m2 + m3 + m4 ;
avg = total / 4 ;
cout << "\n " << " The average marks = " << avg ;
return 0 ;
}
```

Output

The average marks = 65 . 500000

مثال 20.3 لاندې پروګرام د دایرې مساحت محاسبه کوي:

```
// program to compute area of a circle
 #include <iostream.h>
main ( )
{
// Declarations
float pi , r , area ;
// Assignment
pi = 3.14159 ;
r = 5 ;
// Calculation and printing
area = pi * r * r ;
cout << "\n " << " Area of circle = " << area ;
return 0 ;
}
```

```
Output
        Area of circle  = 78 . 539749
```


طریقه بله چې ارزښتونه په متغیرونو کې ذخیره کړي هغه د cin دستور په وسیله هم ممکنه ده . د C++ د داخلي (input) دستور دی چې کاروونکي ته اجازه ورکوي قیمتونه مستقیم د keyboardڅخه متغیرونو ته داخل کړي .په مقابل کې، cout د C++ د خارجي (output) دستور دی چې نتیجه په سکرین ښکاره کوي.

مثال 21.3 لاندې پروګرام د دوو بشپړو اعدادو (int) مجموعه محاسبه کوي چې قیمتونه یې د keyboardله لارې اخیستل کېږي:

```
  // program to find sum of two integer numbers
inserted via keyboard
 #include <iostream.h>
 main ( )
 {
 // Declaration of variables
 int N1, N2, Sum;
 cout << “Enter 2 Integer numbers to Sum : “;
 cin >> N1;
 cin >> N2;
 Sum = N1 + N2;
 cout << “\n “ << “ The Sum is = “ << Sum;
 return 0; }
```

Output

```
Enter 2 Integer numbers to Sum : 10   20
The Sum is = 30
```

مثال 22.3 لاندې پروګرام درې عددونه له keyboard څخه اخلي او د هغوی جمع، تفریق، ضرب او تقسیم محاسبه کوي:

```
/* program to find sum, Minus, Product and Division of three float numbers inserted from keyboard */
   # include <iostream.h>
```

```
main) (
{
// Declaration
float N1, N2, N3, Sum, Min, Prod, Div;
cout << "Enter 3 Integer numbers to Sum, Minus, Product and Divide : ";
cin >> N1 >> N2 >> N3;
Sum = N1 + N2 + N3;
Min = N1 - N2 - N3;
Prod = N1 * N2 * N3;
Div = N1 / N2 / N3;
cout << "\n " << Sum;
cout << "\n " << Min;
cout << "\n " << Prod;
cout << "\n " << Div;
return 0; }
```

Output

```
Enter 3 Integer numbers to Sum, Minus, Product and Divide 20  10  5
35
5
1000
0.4
```

مثال 23.3 لاندې پروګرام ساده ګټه (simple interest) محاسبه کوي:

```
        // program to calculate simple interest
#include <iostream.h>
main ( )
{
int year ;
float prin, rate, si ;
cout << " Enter principle , rate and period : " ;
cin >> prin >> rate >> year ;
```

```
        si = prin * rate * year / 100 ;
        cout << "\n " << " Simple interest = " << si ;
return 0 ;      }
```

Output

```
        Enter principle, rate and perios : 1000    5    2          Simple
interest  = 100. 0000000
```

مثال 24.3 لاندې پروګرام د دوو کرکټرونو (characters) داخلول ښيي:

```
 Program to find radius of a circle given the circumference
#include <iostream.h>
main) (
{
char x , y;
cout << "\n " << " Enter two characters separately : \t;"
cin >> x >> y;
cout << "\n" << "You entered " << x <<" and " << y << "
characters ;"
return 0;
}
```

Output

```
Enter two characters separately : a b
You entered a and b characters
```

څلورم فصل

دا فصل د عملګرونو(Operators) ، د عملګرونو ډولونو، عباراتو (Expressions) او د عباراتو ډولونو په اړه دی .هر یو له دې مفاهیمو څخه په تفصیل سره بحث شوی دی.

د C++ ژبه د ساده حسابي عملیاتو لپاره څو عملګرونه لري لکه :جمع، تفریق، ضرب، تقسیم، باقي مانده او نور منطقي عملیات.

عمل ګر (Operator) هغه سمبول دی چې د ځانګړو حسابي یا منطقي عملیاتو د اجرا لپاره کارول کیږي .هغه ډاټا چې پرې عملیات ترسره کیږي د (Operands) په نوم یادیږي.

عبارت (Expression) د عملګرونو او (Operands) ترکیب دی چې په پایله کې یو قیمت تولیدوي.(2007, Soulié)

لاندې مثال وګورئ:

```
12 * 5
```

په دې کې * د ضرب عملګر دی او 5,12 او Operands دي.

په عمومي ډول عملګرونه په څلورو ډلو وېشل کیږي:

i. یوګوني عملګر(Unary Operator)

ii. دوه ګوني عملګر(Binary Operator)

iii. دري ګوني عملګر(Ternary Operator)

iv. ځانګړي عملګر(Special Operator)

۱.۱۵ د یوګوني عملګرونه(Unary Operators)

یوګونی عملګر هغه عملګر دی چې یوازې په یو operand باندې اجرا کیږي .دا عملګر تل د operandمخکې راځي .د C++ په ژبه کې د یوګوني عملګرونو مختلف ډولونه شتون لري (1997, Stroustrup):

i . مثبت یوګونی (+) -(Unary plus)

ii . منفي یوګونی (-) -(Unary minus)

iii . زیاتوالی (++) -(Increment)

iv . کمښت (--) -(Decrement)

v . منطقي نفی (!) -(Logical NOT)

vi . بیتی مکمل (~) -(Bitwise complement)

vii . آدرس(&) -(Address)

منفي یوګونی (-) د تفریق لپاره نه بلکې د عدد نښه بدلوي .دا موجود قیمت په 1- ضرب کوي . همدارنګه مثبت یوګونی (+) موجود قیمت ته بدلون نه ورکوي، یوازې یې مثبت حالت ښیي.

مثال 1.4 د a قیمت یو زیاتوي:

```
 /* Program to print a message */
#include <iostream.h>
main ( )
{
int a = 10;
a++;
cout <<"The Value of a is: "<< a << endl;
return 0;
}
```

Output

The Value of a is: 11

دوه ګوني عملګرونه(Binary Operators)

دا هغه عملګرونه دي چې په دوو Operands باندې اجرا کېږي .د C++ ژبه کې دوه ګوني عملګرونه په پنځو ډلو وېشل شوي دي:(1995, Oualline)

i . حسابي عملګر(Arithmetic Operator)

ii . ارتباطي عملګر(Relational Operator)

iii . منطقي عملګر(Logical Operator)

iv . د مساوي ورکولو عملګر(Assignment Operator)

v . بیتی عملګر(Bitwise Operator)

حسابي عملګرونه(Arithmetic Operators)

په C++ ژبه کې پنځه حسابي عملګرونه شته چې په لاندې جدول کې ښودل شوي دي:

جدول 1.4 حسابي عملګرونه

عملګر	هـــــدف
+	جمع
*	تفریق
/	ضرب
%	تقسیم باقیمانده

په C++ کې د توان (power) لپاره مستقیم عملګر نشته، خو د pow() فنکشن د توان محاسبې لپاره کارول کېږي.

د بشپړ **عدد تقسيم** (Integer Division) هغه حالت دی چې یو بشپړ عدد په بل بشپړ عدد تقسیم شي او اعشاري برخه له منځه ځي.

عملګر % د باقي‌مانده (remainder) ترلاسه کولو لپاره کارول کېږي.

مثال :که a = 10 او b = 3 وي:

10 % 3 = 1

10 / 3 = 3

یادونه :د % عملګر یوازې په بشپړو عددونو (integers) باندې کار کوي.

مثال 2.4 متغیرونه a او b د بشپړو اعدادو (int) ډول څخه په نظر کې ونیسئ:

a = 16 , b = 3.

عبارتونه	نتایج
+16	16
-5	-5
a + b	19
a - b	13
a * b	48
a / b	۵(کېږي حذف برخه اعشاري)
a % b	۱(باقیمانده)

متحولونه C1 او C2 د حروفي (char) ډول دي او د P او T حروف لري:

C1 = ‘ A ‘ and C2 = ‘ a ‘
C1 = 65 , and C2 = 97.

عبارتونه	نتایج
C1 + C2	162
C1 + C2 + 5	167
’C1 + ‘1	114

یادونه :د ’1‘ قیمت 49 دی.

’1‘ + C1 بشپړ عددونو د تقسیم (division) کې که دواړه عددونه منفي وي یا یو منفي وي، نتیجه د کمپیوټر د سیستم پورې اړه لري.

مثال:

```
5 / 6 = 0
-5 / -6 = 0
```

0 = 6- / 5یا) 1- په Turbo C++ کې 0 ښیي(

په باقي‌مانده (modulus %) کې د نتیجې نښه د لومړي عدد له نښې سره ټراو لري:

```
-20 % 6 = -2
-20 % -6 = -2
20 % -6 = 2
```

که کله دواړه عددونه اعشاري وي، نو داسې عملیات ته Arithmetic Real ویل کیږي .په دې ډول عملیاتو کې عددونه کېدای شي اعشاري یا توان‌دار وي او نتیجه هم د اعشاري په شکل څرګندیږي.

۱.۱۶.۱ عملیات مختلط حسابي (Mixed-Mode Arithmetic Operations)

مختلط حسابي عملیاتو کې یو عدد د (int) بشپړ عدد وي او بل عدد د (real) اعشاري ډول وي . کله چې له دې دواړو څخه یو عدد اعشاري وي، نو نتیجه هم اعشاري کیږي.(2002, Prinz)

مثال:

15/2.5=7.5 و15/2 = 7

۱.۱۶.۲د حسابي عملګرونو د اولویت ترتیب (Precedence of Arathematic Operators)

هر عبارت د اولویت د قواعدو مطابق ارزول کېږي.

مثال:

4 * 5 / 2

دا عبارت له چپ څخه ښي خوا ته اجرا کېږي .لومړی 5 * 4 کېږي او نتیجه (20) بیا په 2 تقسیم کېږي.

جدول 2.4 د عملګرونو د اولویت تشریح

حق اولویت	عملګر	جهت فعالیت
1	( )	چپ څخه ښي ته
2	Unary	ښي څخه چپ ته
3	%, /, *	چپ څخه ښي ته
4	+, -	چپ څخه ښي ته

- هغه عبارت چې په قوسونو ( ) کې وي لومړی اجرا کېږي.

•که څو قوسونه وي، نو هغه چې په منځ کې وي لومړی اجرا کېږي او بیا له داخل څخه بهر ته یو په بل پسې اجرا کېږي.

•که په یوه عملیات کې دوه یا زیات داسې عملګرونه موجود وي چې د اولویت حق یې یو شان وي، نو عملیات له چپ څخه ښي لور ته اجرا کېږي .یعنې هغه عملګر چې مساوي نښې ته نږدې وي لومړی اجرا کېږي.

•قوسونه ( ) کولای شي د عملګرونو اولویت بدل کړي.

مثال 3.4 لاندې معادله په نظر کې ونیسئ:

$$X1 = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

اګر $a = 1$، $b = 4$ و $c = 4$

نو په C++ کې يې معادل عبارت داسې دی:

```
x    = ( -b + sqrt ( b * b – 4 *a*c) / ( 2 * a )
     = (( -4 ) + sqrt ( 4 * 4 – 4 *1 * 4 )) / ( 2 * 1)
     = ( -4 + sqrt (4 * 4 – 4 * 1 * 4 )) / ( 2 * 1 ) // Evaluation inner most prantheses
      = ( -4 + sqrt ( 16 – 4 * 4 )) / ( 2 * 1)
     = ( -4 + sqrt (16 – 16 )) / ( 2 + 1)
     = ( -4 + sqrt (0)) / ( 2 * 1)
     = ( -4 + 0) / ( 2 * 1)
     = -4 / 2
x1   = -2
```

مثال 4.4 لاندې عبارت په نظر کې ونیسئ:

```
( -5 * 2 / ( 6 + 4 - 2)  +  ( 2 / 3 + 5 ))
```

د هغه د ارزوني ترتیب په لاندې ډول ده

```
= ( -5 * 2 / ( 6 + 4 - 2) + ( 2 / 3 + 5 ))
= ( -5 * 2 / (10 – 2 ) + ( 2 / 3 + 5 ))
= ( -5 * 2 / 8 + (2 / 3+ 5))
= ( - 5* 2 / 8 + ( 0 + 5 ))
= ( -5*2 / 8 + 5)
=(-10 / 8 + 5 )
= -1 + 5
```

= 4

هغه پروګرام چې لاندې عبارت محاسبه کوي.

```
// Program to Evaluate a given Expression
#include <iostream.h>
 main (
{
 int result ;
  cout << " \n " << "The given expression is : (-5 *2/(6+4-2) + ( 2/3 + 5)) " ;
 result = (-5 * 2 / ( 6 + 4 -2 ) + ( 2/3+5));
 cout << "\n" << "Result of evaluation is : " << result ;
 return 0;
}
```

Output

```
The given expression is : (-5 *2/(6+4-2) + ( 2/3 + 5))
 Result of Evaluation  : 4
```

که که که له یوه څخه زیات قوسونه یو ځای موجود وي، نو د هغوی ارزونه له چپ څخه ښي لور ته ترسره کیږي .یعني لومړی چپ اړخ قوسونه اجرا کیږي او بیا هغه قوسونه چې ښي اړخ ته وي ورو ورو اجرا کیږي .قوسونه هغه وخت کارول کیږي چې د عملګرونو په اولویت کې شک یا ابهام موجود وي .همدارنګه د قوسونو کارول پروګرام لا روښانه او د لوستلو لپاره اسانه کوي.

مثال 5.4 لاندې عبارت په نظر کې ونیسئ:

```
++x  -  y ++      where  x = 6 ,  y = 2
++x  -  2         unary operators ( from right to left)
7    -  2         uniary operators
                  First  x = 6 , x = 7 (pre-increment)
   5              Substraction
```

۱.۱۷ د ارتباطي عملګرونه(Relational Operators)

دا عملګرونه په شرطی عملیاتو کې کارول کېږي، چیرې چې نتیجه True یا False وي .دا عملګرونه هر عبارت په منطقي ډول ارزوي.

که نتیجه ناسمې وي، نو هغه ته 0 بنودل کېږي، او که نتیجه سمه وي نو هغه ته د یو غیر صفر عدد په شکل بنودل کېږي.

مثال:

50 < 200

X > Y

دې ډول عبارتونو ته relational expressions ویل کېږي او د هغوی نتیجه یا True وي او یا False.که X = 10 او Y = 5 وي، نو:

Y > X → False

X > Y → True

په عمومي ډول د ارتباطي عملګرونو شپږ ډولونه شتون لري:

عمل ګر	معنی
<	له څخه لوی
>	له څخه کوچنی
=<	لوی یا مساوي
=>	کوچنی یا مساوي
==	مساوي
=!	نا مساوي

د وروستيو دوو عملګرونو ته equality operators هم ويل کيږي.

مثال 6.4 لاندې متغيرونه په نظر کې ونيسئ:

مثال ۶.۴

لاندی تام متحولونه په نظر کې ونيسي $i = 1,\ j = 2,\ k = 7$

عبارتونه	منطقي ارزښت	صحيح ارزښت
i. $10.5 <= 12$	سم	۱
ii. $i == 20$	ناسم	۰
iii. $5 > 20$	ناسم	۰
iv. $(I + j) > k$	ناسم	۰
v. $(j + k + 1) > (i + 5)$	سم	۱
vi. $k != 7$	ناسم	۰

په دوهم او دريم مثال کې لومړی د قوسونو دننه حسابي عمليات اجرا کيږي او وروسته نتيجه له k سره مقايسه کيږي.

عملګرونه حسابي (Arithmetic Operators) د ارتباطي عملګرونو (Relational Operators) په پرتله لوړ اولویت لري .ارتباطي عملګرونه د دوه قیمتونو د مقایسې لپاره په تصمیم نیونکو بیانیو کې کارول کیږي

مثال 7.4 لاندې پروګرام د ارتباطي عملګرونو (Relational Operators) کارول ښیي:

```
// Program to show usage of relational operators
#include <iostream.h>
main ( )
{
 int  i = 1, j = 2, k = 7 ;
 cout <<"\n" <<  " Value of  " << 10 << " <= " << 12 ;
 cout <<"\n"<< " Value of i" <<" ==" << 2 ;
 cout <<"\n"<< " Value of " << 4 << " < " << 5;
 cout <<"\n"<< " Value of " << i + j << " != " << k ;
 cout <<"\n"<< "Value of " << j + k + 1 << " > " << i + 5 ;
 cout <<"\n"<< "Value of " << k << " == " << 7 ;
 return 0 ;
}
```

Output

```
Value of  10 <= 12
Value of  i == 2
Value of  4 < 5
Value of 3  != 7
Value of 10 > 6
Value of 10 == 7
```

۱.۱۲.۱ عملګرونه منطقي(Logical Operators)

دا عملګرونه د دوو یا زیاتو منطقي عباراتو د یو ځای کولو لپاره کارول کیږي .په C++ کي دري منطقي عملګرونه شتهAND :، OR او.NOT

۳,۴جدول عملګرونه منطقي(Logical Operators)

عمل ګر	معنی
!	Not منطقی
&&	AND منطقی
||	OR منطقی

- د AND منطقي نتيجه هغه وخت True وي کله چې دواړه قيمتونه True وي، خو که له دې دوو څخه يو هم False وي نو نتيجه False کيږي.
- د OR منطقي نتيجه هغه وخت False وي کله چې دواړه قيمتونه False وي، خو که له دې دوو څخه يو هم True وي نو نتيجه True کيږي.
- په C++ کې د ! عملګر د NOT لپاره کارېږي چې د منطقي عبارت نتيجه برعکس ښيي.

جدول 4.4 د حقيقت جدول (Truth Table) د منطقي عملګرونو لپاره

مثال 8.4 که i = 8 ، x = 5.1 ، او c = 'a' وي، او i د int ډول، x اعشاري او c د char ډول وي، نو د منطقي عملګرونو استعمال په لاندې ډول دی:

عبارت(Expression)	منطقي قيمت(Logical Value)	عددي قيمت(Integer Value)
(i > 6) && (c = = 'a')	True	۱ (خلاف صفر)
i < = 6 || (c = = 'b')	False	۰
i >= 6 && (c = = 97)	True	۱
(x < 10) || (I > 10)	True	۱
! (i < = 4)	True	۱
(c ! = 'p')	True	۱

جدول 5.4 د منطقي عملګرونو د حقيقت جدول (Truth Table) تشريح

1شرط A	2 شرط B	نتيجه b && a	نتيجه b || a	نتيجه شرط اول ! a
(True) سم (True)سم (False) ناسم (False) ناسم	(True) سم (False) ناسم (True) سم (False) ناسم	(True) سم (False) ناسم (False) ناسم (False) ناسم	(True) سم (True)سم (True) سم (False) ناسم	(False) ناسم (False) ناسم (True) سم (True)سم

۱.۱۷.۲ عملګر = (Assignment Operators)

دا عملګر (=) هغه وخت کارول کيږي کله چې يو متغير، ثابت يا عبارت ته قيمت ورکړل شي.

د دې عمومي جوړښت په لاندې ډول دی:

```
variable = variable / constant / expression;
```

مثالونه:

```
X = 10 ;
```

په دې عبارت کې د X قيمت 10 دی.

```
Sum = B + X ;
```
په دې عبارت کې د X قيمت) چې 10 دی B (ته ورکول کيږي.

په دې عبارت کې د B او X مجموعه (20) په Sum کې ذخیره کېږي.

مثال 9.4 لاندې پروګرام د 4 عددونو مجموعه محاسبه کوي او په سکرین یې ښیي:

```
// Program to print sum of 4 numbers
#include <iostream.h >
  main ( )
{
     int  a, b, c, d, sum ;
    cout << "\n " << " Enter  4 numbers : " ;
   cin >> a >> b >> c >> d;
   sum =  a + b  + c + d ;     // assignment statement
   cout << "\n" << sum ;
   return 0 ;
}
```

Output

```
Enter 4 numbers : 10   15   22   40
 Sum = 87
```

```
Variable  operator  =  expression
```

مثال:

```
N = N + 5 ;
```

دا عبارت په لنډ ډول داسې لیکل کېږي:

```
N += 5 ;
```

دلته += د ترکیبي (Compound) عملګر په نوم یادیږي چې د (N = N + 5) لنډ شکل دی .دا معنا لري چې 5 د N موجود قیمت سره جمع کیږي او نتیجه بیا په N کې ذخیره کیږي.

د += معنا دا ده چې ښي طرف قیمت د چپ طرف متغیر سره جمع کوي او نتیجه بیا په هماغه متغیر کې ساتي.

ځینې نور ترکیبي عملګرونه:

جدول 6.4 د لنډیز عملګرونه

معنا	عمل ګر	= به شکل ترکیبی	= به شکل عادی
= و +	=+	n += 1	n = n + 1
= و -	=-	n -= 50	n = n – 50
= و *	=*	n *= 10	n = n * 10
= و /	=/	n /= 6	n = n / 6
= و %	=%	n %= 5	n = n % 5

د مختصر عملګرونو ګټې:(Advantages of Shortened Assignment Operators)

1. متغیر بیا په ښي طرف نه لیکل کیږي.
2. عبارت لنډ او د لوستلو لپاره اسانه کیږي.
3. د اجرا موثریت (efficiency) زیاتوي.

عملګرونو تر منځ توپیرد

1. == د مقایسې (Relational / Equality Operator) عملګر دی او د مساوي والي د چک کولو لپاره کارول کیږي .دا د دوو قیمتونو مقایسه کوي .مثال(If (x == 10 :

په داسي حال کي چې = د توظيف (Assignment Operator) عملګر دی او يو قيمت متغير ته ورکوي.

مثال:

Y = 20 ;

2. = =د متغير قيمت نه بدلوي، يوازي يې له بل قيمت سره مقايسه کوي .خو = د چپ طرف متغير قيمت بدلوي او د ښي طرف قيمت په کي ذخيره کوي.

څوعملګر= (Multiple Assignment)

C ++ مورِ ته اجازه راکوي چې څو = په يوه بيانيه کي وکاروو، لکه:

```
Variable = Variable1 = Variable2 = ……= Experssion;
```

10 = Z = Y = X

دا په اصل کي داسي معنا لري:

10 = (Z = (Y = X))

يعني لومړی X ته قيمت ورکول کيږي، بيا Y ته، او په پای کي Z ته .په پای کي دX ، Y او Z قيمتونه ټول 10 کيږي.

مثال 10.4 لاندې پروګرام د دوو عددونو ترمنځ لوی عدد پيدا کوي:

```
// Program to find Largest of 2 numbers
#include <iostream.h>
 main ( )
{
 int n, m, big ;
 cout << "\n" << "Enter two integer numbers : " ;
cin >> n  >> m ;
```

```
  big = ( n > m ) ? n : m ;
  cout << "\n" << "The Largest of " << n <<" and " << m << " is : " << big ;
}
```

Output

```
Enter two integer number : 10  56
The Largest of  10  and  56  is : 56
```

مثال 11.4 لاندې پروګرام چک کوي چې عدد زوج (even) دی که طاق:

```
// Program to find whether a numer is even or odd
# include <iostream.h<
  main( )
{
  int x ;

   cout << "\n" << "Enter a number; " :
 cin >> x;
 ( x  % 2 == 0 ) ?  cout << x << " is even " : cout <<x << " is odd ;"
}
```

Output

```
Enter a number :  17
17  is odd
Enter a number : 20
20  is even
```

مثال 12.4 لاندې پروګرام د څلورو مضامینو ترمنځ تر ټولو لوړه نمره (highest marks) پیدا کوي:

```
//Program to find the highest marks of a student in 4 Exams
# include <iostream.h<
  main( )
{
  int m1, m2, m3, m4, highest;
```

```
    cout << "\n " << "Enter the marks in 4 papers  : "  << "\n ";
   cin >> m1 >> m2 >> m3 >> m4;
   highest = m1 > m2 ? m1 :  m2;
   highest = highest  > m3 ? highest :  m3;
   highest = highest  > m4 ? highest :  m4  ;
   cout << "\n " << "Highest marks in 4 papers  =  " << highest;
  }
```

Output

```
Enter the marks in 4 papers :
56  74  66  60
Highest marks in 4 papers = 74
```

عملګرونه بیتی(Bitwise Operators)

دا عملګرونه په مستقیم ډول د عددونو په بیټونو (bits) باندې کار کوي .په C++ کې د بیتی عملګرونه په لاندې ډول دي:

i. بیتی(&) AND

ii. بیتی(|) OR

iii. بیتی(^) XOR

iv. بیتی(~) NOT

v. چپ ته بدلون (<<) Left Shift

vi. ښي ته بدلون(>>) Right Shift

بیتی عملګر(&) AND

دا عملګر د منطقي AND په څېر کار کوي، خو په بیټونو باندې .که دواړه بیټونه 1 وي نتیجه 1 وي، په بل صورت 0 وي.

بیتی عملګر (|) OR

دا عملګر د منطقي OR په څېر کار کوي .که لږ تر لږه یو بیت 1 وي، نتیجه 1 کېږي، او که دواړه 0 وي نو نتیجه 0 وي.

عملګر بیتی (~)NOT

دا عملګر د بیتونو ارزښتونه برعکس کوي، یعني 1 ته 0 او 0 ته 1 بدلوي.

عملګر بیتی (^) XOR

په دې عملګر کې که د دواړو اړخونو بیتونه یو شان وي 0) او 0 یا 1 او (1 نو نتیجه 0 وي، او که مختلف وي نو نتیجه 1 وي.

عملګر بیتی چپ لور ته بدلون (<<) Shift Left

دا عملګر د Operand چپ لوري بیتونه د ښي لوري Operand له مخې ټاکل شوي شمېر (n) په اندازه چپ لور ته انتقالوي.

00001010 = 10

00101000 = 2 << 10

عملګر بیتی ښي لور ته بدلون (>>) Shift Right

دا عملګر د Left Shift په څېر دی، خو بیتونه ښي لور ته انتقالوي.

0110100 = 100

00000110 = 100 >> 3

جدول د حقیقت (Truth Table) د XOR, NOT, OR, AND عملګرونو لپاره

X	Y	X~	Y~	AND	OR	XOR
۱	۱	۰	۰	۱	۱	۰
۱	۰	۰	۱	۰	۱	۱
۰	۱	۱	۰	۰	۱	۱
۰	۰	۱	۱	۰	۰	۰

جدول 7.4 د بیتی عملګرونو د حقیقت جدول

مثال 13.4 د بیتی عملګرونو استعمال په C++ کې:

```
#include <iostream.h>
    main ( )
{
  int And, Or, Not_a, Not_b, Xor;
   int a = 8 , b=5;
   And = a & b;
   Or = a | b;
   Not_a = ~a;
   Not_b = ~b;
   Xor = a ^ b;
   cout <<"The Value of a&b is: "<< And << endl;
   cout <<"The Value of a|b is: "<< Or << endl;
   cout <<"The Value of a NOT is: "<< Not_a << endl;
   cout <<"The Value of b NOT is: "<< Not_b << endl;
   cout <<"The Value of a^b is: "<< Xor << endl;   return 0;
  }
```

Output

```
The Value of a&b is: 0
The Value of a|b is: 13
The Value of a NOT is: -9
The Value of b NOT is: -6
The Value of a^b is: 13
```

عملګرونه دري تایی(Ternary Operators)

دا عملګرونه په درېو عناصرو باندې کار کوي .د (:?) عملګر د شرطی عملګر په نوم هم یادیږي .دا موضوع په پنځم فصل کې تفصیل سره تشریح شوې ده.

عملګرونه خاص(Special Operators)

دا ډول عملګرونه کې کامه (,) او sizeof شامل دي.

عملګر(sizeof)

i . دا عملګر د ورکړل شوي قیمت یا متغیر اندازه په بایټ (byte) کې ښیي.

ii . د دې عمومي جوړښت داسې دی:

```
sizeof(n)
```

دلته n کېدای شي متغیر، ثابت یا د معلوماتو ډول (data type) وي.

iii . دا یو یکتایي (Unary) عملګر دی او له ښي لوري څخه چپ لوري ته ارزول کېږي.

iv . دا عملګر د array او structure د اندازې معلومولو لپاره هم کارول کېږي .همدارنګه د dynamic memoryسره په کار کې هم استعمالېږي.

مثال 14.4 لاندې پروګرام د sizeof عملګر استعمال ښیي:

```
// Program to show usage of sizeof operator
  #define STRING "Usage of sizeof Operator"
  #include <iostream.h>
      main ( )
{
   char  a = ' p ' ;
   cout << "\n" << " The size of char is          : " << sizeof (char) ;
   cout << "\n" <<" Therefore the size of char a is  :" << sizeof (a) ;
   cout << "\n" <<" However the size of ' p ' is    :"<< sizeof ( 'p');
   cout << "\n\n" << "STRING   Usage of sizeof Operator\n";
   cout << " \n" << " The number of bytes in STRING IS :" << sizeof (STRING)
;
   cout <<"\n"  << "The size of short is          :" << sizeof (short);
   cout << "\n " << "The size of int is           :" << sizeof (int) ;
```

```
cout << "\n " << "The size of long is          :" << sizeof (long) ;
cout << " \n " <<"The size of float is         :" << sizeof (float) ;
cout << " \n " << "The size of double is        :" << sizeof (double) ;
}
```

Output

```
The size  of char is                : 1
Therefore the size of char a is     : 1
However the size of ' p ' is        : 1
STRING      " Usage of size Operator "
The number of bytes in STRING IS  : 25
The size of short is                : 2
The size of int is                  : 4
The size of long is                 : 4
The size of float is                : 4
The size of double is               : 8
```

۱.۱۸عبارتونه(Expressions)

عبارتونه هغه جوړښتونه دي چې له عملګرونو(operators) ، ثابتو (constants) او متغيرونو (variables)څخه جوړيږي .دا ممکن د فنکشنونو (functions) غږونه هم ولري چې قيمتونه بيرته راګرځوي.

په عمومي ډول په C++ کې څلور ډوله عبارتونه شته:

- ثابت عبارت(Constant Expression)
- حسابي عبارت(Arithmetic Expression)
- ارتباطي عبارت(Relational Expression)
- منطقي عبارت(Logical Expression)

هغه عبارت چې د څو نورو عبارتونو له ترکيب څخه جوړ شي، د مرکب عبارت (Compound Expression)په نوم يادیږي.

عبارت ثابت (Constant Expression) : هغه عبارت دی چې په جوړښت کې یوازې ثابت (constant)قیمتونه ولري.

عبارت حسابي(Arithmetic Expression) : هغه عبارت دی چې په ترکیب کې متغیرونه (variables)، ثابتونه (constants) او حسابي عملګرونه (arithmetic operators) ولري.

عبارت ارتباطي(Relational Expression) : هغه عبارت دی چې په جوړښت کې متغیرونه، ثابتونه، حسابي عملګرونه او مقایسوي عملګرونه ولري .د دې ډول عبارت نتیجه True یا False وي.

عبارت منطقي (Logical Expression) : هغه عبارت دی چې دوه یا له دوو څخه زیات ارتباطي عبارتونه د منطقي عملګرونو په وسیله وصل شوي وي .د دې نتیجه هم True یا False وي.

۱.۱۹ بیانیې(Statements)

بیانیه د پروګرام مهمه برخه ده چې د اجرا وړ وي.

د بیانیې ځانګړتیاوې:

- بیانیه د پروګرام یوه کوچنۍ برخه ده چې اجرا کېدای شي.
- ټولې بیانیې باید په سیمي کولن (;) ختمې شي.
- هغه دستورونه چې په یوه بلاک کې وي او یوځای اجرا کېږي، باید د { } قوسونو کې ولیکل شي.

د بیانیو ډولونه:

بیانیې انتخابي :(Selection Statements) لکه if اوswitch.

بیانیې تکراري :(Iteration/Loop Statements) لکهfor ، while اوdo-while .

بیانیې د پرش :(Jump Statements) لکهbreak، continue ، gotoاوreturn.

لیبل لرونکي بیانیې :(Labeled Statements) لکهcase ، default اوgoto .

عبارتي بیانیه :(Expression Statement) هغه بیانیه چې د یو یا څو عبارتو څخه جوړه شوې وي.

بلاک بیانیې :(Compound Statements) هغه بیانیې چې د { } قوسونو ترمنځ لیکل کېږي او څو بیانیې پکې شاملې وي.

۱.۱۹.۱ ارزیابی عبارتونه(Expressions of Evaluation)

•که غواړو یو بیان (expression) ارزونه کړو، نو باید الجبري افاده (algebraic expression)په C++ ښه بدله کړو.

•په دوهم قدم کې باید دا په (=) assignment statement شکل ولیکل شي.

•ټول متغیرونه چې په ښي طرف کې وي، مخکې له ارزوني باید قیمت ولري.

•هر ځای چې اړتیا وي باید مناسب عملګر (operator) ذکر شي.

•په C++ کې د توان (power) لپاره مستقیم عملګر نشته، نو د pow() فنکشن استعمالیږي.

جدول 8.4 د حسابي عبارتونو او د هغوی د C++ بدیل

عبارت های حسابی	عبارت های C++
1. ab + cd	1. a * b + c * d
2. (a + b)(a - b)	2. (a + b) * (a – b)
3. $\frac{a}{b}$ + d	3. (a / b) + d
4. $\frac{2x^2 + 3x - 1}{10}$	4. (2 * x * x + 3 * x - 1) / 10
5. root = $\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$	5. root = (-b+sqrt (b*b –4*a*c) / (2*a)
6. $a^2 - \frac{b}{2} + c^2$	6. a * a – b / 2 + c * c

د عملګرونو د اولویت قاعده(Precedence of Operators)

•هغه عملګر چې لوړ اولویت ولري، لومړی اجرا کیږي.

•که عملګرونه مساوي اولویت ولري، نو د چپ څخه ښي طرف ته اجرا کیږي.

مثال:

(a < 10 && a + b > 15)

که a = 20 او b = 1 وي:

لومړی + اجرا کیږي:

a + b = 21

بيا عبارت داسي کيږي:

a < 10 && 21 > 15

ii . دواړه ارتباطي عملګرونه (> او <) د منطقي AND (&&) په پرتله لوړ اولويت لري .نو لومړی د چپ طرف شرط او بيا د ښي طرف شرط ارزول کيږي.

20 < 10 && 21 > 15

false && true

iii.د AND منطقي عملګر نتيجه تل false وي کله چې يو شرط هم false وي، نو پايله 0 (false)کيږي.

جدول 9.4 د C++ د عملګرونو د اولويت ترتيب

د عملګرونو د لومړيتوب جدول

عملګر	د عملګر تشريح	د فعاليت لوری	د لومړيتوب درجه
. ، → ، [ ] ، ( )	د تابع غږول، کوچني قوسونه، د صنف عناصرو ته لاسرسي، د عناصرو تاکنه	له چپ خخه ښي ته	1
+ ، - ، ++ ، -- ، sizeof ، * ، ~ ، ! (type)	يو تايي جمع، يو تايي منفي، زياتول، کمول، منطقي NOT، د يو مکمل (One's complement)، د پوانټر اشاره، د متحول اندازه، د ډول بدلول (cast)	له ښي خخه چپ ته	2
* ، / ، %	ضرب، تقسيم او پاتی شونی	له چپ خخه ښي ته	3
+ ، -	جمع او تفريق	له چپ خخه ښي ته	4
<< ، >>	ښي او چپ شفت	له چپ خخه ښي ته	5
< ، <= ، > ، >=	اريکوی (مقايسوی) عملګرونه	له چپ خخه ښي ته	6
== ، !=	مساوی او نامساوی	له چپ خخه ښي ته	7
&	بت AND	له چپ خخه ښي ته	8
^	بت XOR	له چپ خخه ښي ته	9

ادامه: د عملګرونو د لومړیتوب جدول

عملګر	د عملګر تشریح	د فعالیت لوری	د لومړیتوب درجه
|	بټ OR	له چپ خخه شی ته	۱۰
&&	منطقی AND	له چپ خخه شی ته	۱۱
||	منطقی OR	له چپ خخه شی ته	۱۲
?:	شرطي عملګر (ترنري)	له ښی خخه چپ ته	۱۳
=	مساوي	له ښي خخه چپ ته	۱۴
&=	د بټ AND د شکل عملګرونه		
/=	د برابرولو عملګرونه		
%=	د باقي مانده عملګر		
+=	جمع د تاکل (اختياري)		
-=	منفي د تاکل (اختياري)		
&=	د بټ AND د تاکل (اختياري)		
^=	د بټ XOR د تاکل (اختياري)		
!=	منطقي NOT د تاکل (اختياري)		
=>>	د ښي شفت د تاکل (اختياري)		
=<<	د چپ شفت د تاکل (اختياري)		
,	کامه عملګر	له چپ خخه شي ته	۱۵

مثال 15.4: په لاندی مثال کې ولې د اوسط نمرات په ناسم دول محاسبه شوی ده؟

```
#include <iostream.h>
main ( )
{
   int Computer = 90;
   int English = 60;

   float Average = Computer + English / 2 ;  // د لومړیتوب د نااسم ترتیب اشتباه
   cout << "\n The Average = " << Average ;
   return  0 ;
}
```

(Output)

```
The Average = 120
```

در مثال پورته کي لیدل کیږي چې د اوسط محاسبه ناسم شوې وه، ځکه چې د + او / عملګرونه په یو ځای کي راغلي وو او د اولویت له مخي لومړی تقسیم اجرا شوی و.

له همدې امله اړینه ده چې د فورمول د صورت (numerator) برخه د قوسونو () په داخل کي ولیکل شي، تر څو لومړی جمع اجرا شي او بیا تقسیم ترسره شي.

مثال 16.4 لاندې پروګرام کي اوسط په سمه توګه محاسبه شوی دی:

```
#include <iostream.h>
    main ( )
    {
        int Computer = 90;
        int English  = 60;

        float Average =( Computer + English ) / 2 ;
        cout << " \n The Average = "<< Average;
        return 0 ;
    }
```

Output

```
The Average = 75
```

ځینې محاسباتي ستونزې (Some Computational Problems)

د mixed-mode expressions په وخت کي ځینې ستونزې رامنځته کیږي چې د غلطو پایلو سبب کیږي.

په عمومي ډول دوه مهمې ستونزې دي:

- که یو عدد په 0 باندې تقسیم شي، نو محاسباتي ستونزه رامنځته کیږي .دا کار د پروګرام غیر

عادي ختمېدو (abnormal termination) سبب کېږي .له همدې امله باید هېڅکله تقسیم په 0 ونه شي.

- Overflow او Underflow: دا هغه حالتونه دي چې کله نتیجه د متغیر له ظرفیت (range) څخه لوړه یا کمه شي .د دې لپاره باید متغیرونه د سم data type سره تعریف شي او داخلېدونکي

مثال 17.4 لاندې پروګرام د لاندې لړۍ (series) مجموعه محاسبه کوي:

```
/*  Program to find  the summation of the series
 S = 1 +1 / 2 + 1 + .....+ 1 / n  using cast operator   */
#include <iostream.h>
   main ( )
{
   float  s = 0 ;
   int  n,  i ;
   cout << " \n " << " Enter the value of n : " ;
   cin >> n ;
   for ( i = 1 ; i < = n ; i ++ )
   s = s + 1 / ( float ) i ;
   cout << " \n " << " Sum of series  : " << s ;
}
```

Output

```
Enter the value of n : 3
Sum of series  :  1 . 833
```

قیمتونه هم د رینج دننه وي.

دکتابونوتابعي(Library Functions)

په C++ کې مختلف کتابخانهيي (library) فنکشنونه شتون لري چې د ډيلاډيلو عملياتو لپاره کارول کيږي .دا فنکشنونه په ځانګړو header فايلونو کې موجود وي.

•يو کتابخانهيي فنکشن هغه وخت کارول کيدای شي کله چې د فنکشن نوم او اړوند arguments په پروګرام کې ذکر شي.

•arguments بايد د قوسونو ( ) دننه وليکل شي او د کامو (,) په وسيله جلا شي.

•argument کيدای شي ثابت(constant) ، متغير (variable) يا عبارت (expression) وي.

حسابي تابع(math.h)

دا يو مهم کتابخانه دی چې د رياضي عملياتو لپاره کارول کيږي او په header file کې شامل

وي(Meyers, 2005) .م

مثال 18.4 لاندې پروګرام د هر داخل شوي عدد مربع جذر (Square Root) محاسبه کوي:

```
// program to find Square Root of any number inserted from keyboard
#include <iostream.h>
#include <math.h>  // required for sqrt function
    main ( )
{
  // Declaration
  double Number, SQR;
  cout << "Enter a number to find its Square Root : ";
  cin >> Number;
  SQR = sqrt (Number);
  cout << " The square root of "<< Number << " = " << SQR;
  return 0;
  }
```

Output

```
Enter a number to find its Square Root : 36
The square root of  36 = 6
```

مثال 19.4 لاندې پروګرام هر عدد د ورکړل شوي توان (power) ته رسوي:

```
#include <iostream.h>
#include <math.h>  // required for pow function
    main ( )
{
  // Declaration
  double Base, Exponent, Result;
  cout << "Enter value for Base and Power : ";
  cin >> Base >> Exponent;
  Result = pow(Base, Exponent);
  cout << Base <<" ^ " << Exponent << " = " << Result;
  return 0;
  }
```

Output

```
Enter value for Base and Power : 2  3
2 ^ 3 = 8
```

مثال 20.4 لاندې پروګرام د هر عدد n م جذر (n-th root) محاسبه کوي:

```
#include <iostream.h>
#include <math.h>  // required for pow function
    main ( )
{
  // Declaration
  double Base, Root, Result;
  cout << "Enter value for Base and Root : ";
  cin >> Base >> Root;
  Result = pow(Base, 1/Root);
  cout << Base <<" Jazar " << Root << " = " << Result;
  return 0;
  }
```

Output

```
Enter value for Base and Root : 32    5
32 Jazar 5  = 2
```

مثال 21.4 لاندې پروګرام د داخل شوي عدد لوګاریتم (Logarithm) محاسبه کوي:

```
#include <iostream.h>
#include <math.h>  // required for log function
    main ( )
{
  // Declaration
  double Number, Log1, Log2;
  cout << "Enter a number to find its logarithm : ";
```

```
    cin >> Number;
    Log1 = log10(Number);    //Finds log of any number on base 10
    Log2 = log(Number);      //Finds log of any number on base e
    cout << " Log10 " << Number << " = " << Log1;
    cout << " LogE " << Number << " = " << Log2;
    return 0;
    }
```

Output

Enter a number to find its logarithm : 100

Log_{10} 100 = 2

Log_{E} 100 = 4.605

مثال 22.4 لاندې پروګرام د هر زاويې ساين (Sine) محاسبه کوي چې په درجې (degree) کې ورکړل شوی

```
#include <iostream.h>
#include <math.h>  // required for sin function
    main ( )
{
 // Declaration
 double Number, SN;
 cout << "Enter an angle size to find its Sin : ";
 cin >> Number;
 SN = sin(Number * 3.14159 / 180);
 cout << " Sin " << Number << " = " << SN;
 return 0;
 }
```

Output

Enter an angle size to find its Sin : 30

Sin 30 = 0.5

قابل یادونه ده چې cosine (cos) او tangent (tan) هم په همدې طریقه محاسبه کېدای شي.

مثال 23.4 لاندې پروګرام د هر زاویې cotangent کوتانجانت (محاسبه کوي چې په درجه (degree)

```
#include <iostream.h>
#include <math.h>  // required for Cotengent function
    main ( )
{
 // Declaration
 double Number, cot;
 cout << "Enter an angle size to find its Cotengent : ";
 cin >> Number;
 cot = 1/tan(Number * 3.14159 / 180);
 cout << " Cotengent " << Number << " = " << cot;
 return 0;
 }
```

Output

```
Enter an angle size to find its Cotengent : 45
Cotengent  45 = 1
```

تاسو کولای شئ چې secant (1/cos) او cosecant (1/sin) هم په همدې طریقه محاسبه کړئ.

مثال 24.4 لاندې پروګرام د e (2.7182) عدد په توان د داخل شوي عدد لوړوي:

```
#include <iostream.h>
#include <math.h>
    main ( )
{
  // Declaration
  double Number, R;
  cout << "Enter a number: ";
  cin >>Number;
  R = exp (Number);
  cout << " Result = " << R;
  return 0;
  }
```

Output

Enter a number : 2

Result = 7.389

مثال :25.4لاندي پروګرام د کيپورډ څخه داخل شوی هر عدد مطلق قيمت (absolute value)اخلي

```
#include <iostream.h>
#include <math.h>
    main ( )
{
  // Declaration
  int N, R;
  cout << "Enter a number: ";
  cin >>N;
  R = abs (N);
  cout << " Result = " << R;
  return 0;
}
```

Output

Enter a number : -200

Result = 200

پنځم فصل

په دې فصل کې شرطي دستورونه، د شرطي دستورونو ډولونه او د هغوی کارونې تر بحث لاندې نیول شوي او یو په بل پسې په تفصیل سره تشریح شوي دي.

په C++ ژبه کې دستورونه عموماً په هماغه ترتیب سره اجرا کېږي لکه څنګه چې لیکل شوي وي .دا ډول پروګرامونه مسلسل جوړښت (Sequential Structure) لري.

خو په ځینو حالاتو کې اړتیا وي چې د پروګرام اجرا مسلسل نه وي او د ځانګړو شرطونو له مخې ترسره شي .هغه پروګرامونه چې پکې پرېکړه کول د شرط پورې تړلي وي، انتخابي پروګرامونه (Selection Structure)بلل کېږي .نو ویلی شو چې شرطي دستورونه د بېلابېلو شرطونو پر بنسټ د کارونو او پرېکړو لپاره کارول کېږي.

په C++ ژبه کې شرطي بیانیې شته چې د اړتیا په وخت کې تر ې کار اخیستلای شو .دا بیانیې عبارت دي له:

i . بیانیه If

ii . بیانیه switch

iii . درې‌تایي عملګر(ternary operator)

iv . بیانیه goto

د goto بیانیه هغه وخت کارول کېږي چې د پروګرام یوه ځانګړې برخه باید د اړتیا له مخې اجرا شي.

۲۱ chapter د if شرطي بیانیه

د If شرطي بیانیه د کمپیوټر د پروګرامنګ ژبو له قوي بیانیو څخه ده .دا بیانیه ورکړل شوی شرط په منطقي ډول ارزوي؛ که شرط سم وي، نو لومړی عمل چې پروګرامر ټاکلی اجرا کېږي، او که شرط ناسم وي، دوهم عمل اجرا کېږي.

د If بيانيې ساده بڼه په لاندې ډول ده:

```
If (conditional expression)
S1
```

په (conditional expression) کې خپل شرط ليکو S1 .او S2 دوه بيانيې دي چې له ډلې يې يوه د کمپيوټر له خوا د شرط له ارزونې وروسته اجرا کيږي.

•په عمومي ډول، هر (conditional expression) يو ارزښت لري.

(conditional expression) •بايد د کوچنيو قوسونو ( ) دننه وليکل شي.

•S1 هغه وخت اجرا کيږي چې شرط ريښتيا وي.

• که شرط ناسم وي، نو په هغه صورت کې د S1 بيانيه له پامه غورځول کيږي او دوهمه بيانيه (S2)اجرا کيږي.

بيانيه **S1** کولای شي ساده يا مرکبه وي .ساده بيانيه يوازې يو سطر لري، خو مرکبه بيانيه څو سطرونه لري.

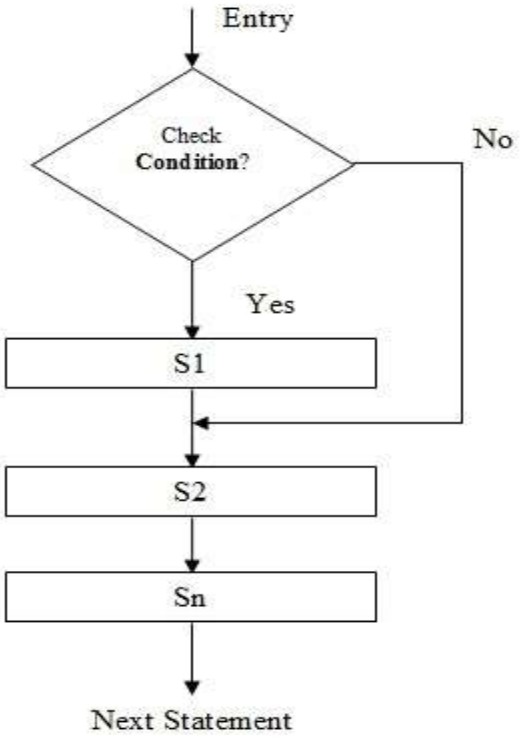
که شرط سم وي او اړتيا وي چې څو سطرونه اجرا شي، نو ټول سطرونه بايد د لويو قوسونو { } دننه وليکل شي.

بېلګه:

که شرط سم وي، ټولې هغه بيانيې چې د لويو قوسونو دننه دي اجرا کيږي.

د If شرطي دستور ساده بڼه په لاندې فلوچارت کې ښودل کېدای شي.

```
If (Condition)
{
S1;
S2;
S3;
```



شکل :1.5 د If شرطي بيانيي فلوچارت

بېلګه :1.5 لاندې پروګرام د if شرطي دستور د کارولو طريقه ښيي.

```
/* Program to show usage of simple if () */
#include <iostream.h>
main ( )
{
        int x , y = 10 ;
        cout << " \n  Enter  value of  x    : " ;
        cin >> x ;
        if ( x > 0 )
        cout << " \n  Sum = " << x + y;
        return 0 ;
}
```

Output

```
        Enter value of   x   :  5
        Sum  =  15
```

بېلګه :2.5 لاندې پروګرام له کیبورډ څخه یو عدد اخلي، که داخل شوی عدد مثبت وي، هغه په 10 ضرب کوي.

```
/* Program multiply only positive number * 10 inserted from keyboard */
#include <iostream.h>
main ( )
{
        int x ;
        cout << " \n Enter a number : " ;
        cin >> x ;
        if ( x > 0 )
        cout << " \n Result = " << x * 10;
        return 0 ;
}
```

Output

Enter a number : 20

Result = 200

یو شمېر نور ساده بېلګې د if شرطي دستور:

```
a) if ( balance = = 0)
                cout << " \n No more shopping !" ;
b) if ( a > b && a > c )
                cout << " \n Largest number is " << a ;
c) int rain = 1;
if ( rain = = 1)
{
                cout << " \n Wear a raincoat!" ;
                cout << " \n or hold an umbrella " ;
}

d) if ( a > 0 && b < 10 )
{
    Sum = a + b ;
    Diff = a - b ;
                cout << " \n Sum = " << Sum ;
                cout << " \n Diff = " << Diff ;
 }

e) if ( dues > 0 )
{
                cout << " \n Account number is overdue " << accno ;
                creadit = 0 ;
 }

f) if ( a = = b || a = = c || b = = c )
{
                cout << " \n It is an isosceles triangle " ;
 }
```

بېلګه :3.5 لاندې پروګرام د دوو داخل شوو عددونو تر منځ لوی عدد پیدا کوي.

```
/* Program to find largest of two numbers */
#include <iostream.h>
main ( )
{
      int a , b, large ;
      cout << " \n Type 2 numbers : " ;
      cin >> a >> b ;
      // Assuming largest number is a
      large = a ;
      if ( b > large )
      large = b ;
      cout << " \n Largest of "<< a << " and "<< b <<" is " << large ;
      return 0;
}
```

Output

```
Type 2 numbers : 15  48
Largest of 15 and 48 is 48
```

بېلګه :4.5 لاندې پروګرام د = او == ترمنځ توپیر ښیي کله چې په شرطي دستور کې وکارول شي.

```
*/Program to show what happens when = is given instead of == in if statement/*
#include <iostream.h<
main( )
{
      int x = 0 ;
      if ( x = 0) // condition becomes false since x = 0
      cout << " x is 0, x==0 is true, but this statement will not be output /n          ;"
        if (x!= 0)
      cout << " x is 0, x!=0 is false, so this statement will not be output /n;"
      if (x=15) // condition becomes true since value of x is not zero
```

```
        cout << " can you believe it, x is actually! " <<x;
        return 0 ;
    }
```

Output

```
Can you believe it, x is actually!  15
```

په څلورم مثال کي، په لومړي if کي 0 ارزښت x ته ورکول کېږي او پايله false کېږي، له همدې امله د cout دننه ليکنه نه ښکاره کېږي.

په دوهم if کي x د 0 خلاف ټاکل شوی، خو بيا هم پايله ناسمېږي او د cout ليکنه نه ښکاري.

په دريم if کي x ته 15 ارزښت ورکړل شوی چې د 0 خلاف دی، نو ځکه د cout پايله پر سکرين ښکاره کېږي.

```
If (Condition)
        S1; else
        S2;
        S3;
```

۱.۲۰د شرطي بيانيه(THE IF ( ) ELSE STATEMENT)

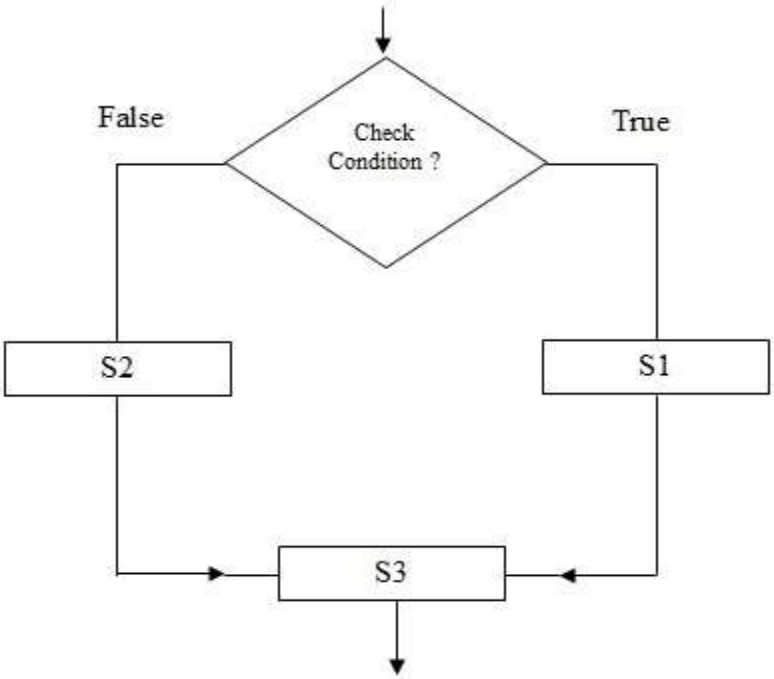
دا دستور دوه انتخابي ځوابونه لري(Eckel, 2000) .م

عمومي بڼه يې په لاندې ډول ده:

- که شرط سم وي، S1 اجرا کېږي، که نه نو S2 اجرا کېږي کله چې شرط ناسم وي.
- په هر حالت کې يوازې يوه بيانيه (S1 يا S2) اجرا کېږي، دواړه يوځای نه اجرا کېږي.
- که شرط سم يا ناسم وي، له S1 يا S2 څخه يوه اجرا کېږي او بيا S3 ته نوبت رسېږي.

د else–If شرطي دستور بڼه:

If (Condition) S1; else S2; S3;



شکل 2.5 فلوچارت بیانیهٔ شرطیif–else

بېلګه :5.5 لاندې پروګرام له کیبورډ څخه یو عدد اخلي، که داخل شوی عدد مثبت وي نو "Positive" او که منفي وي نو "Negative" پر سکرین ښیي

```
/* Program finds whether the entered number is positive or negative */
#include <iostream.h>
main ( )
{
        int N;
        cout << " \n  Enter  a number to find whether it is positive or negative " ;
        cin >> N ;
        if ( N > 0 )
        cout << " \n  The Number is Positive ";
        else
        cout << " \n  The Number is Negative ";
        return 0 ;
}
```

Output

```
Enter  a number to find whether it is positive or negative:  5
The Number is Positive

Enter  a number to find whether it is positive or negative:  - 5
The Number is Negative
```

بېلګه :6.5 لاندې پروګرام له کیبورډ څخه یو حرف اخلي، که داخل شوی حرف A وي نو "A" داخل شو "پیغام ښیي، او که بل څه وي نو" نادرست حرف "ښیي.

```
/* Program multiply only positive number * 10 inserted from keyboard */
#include  <iostream.h>
main ( )
{
        char chr ;
        cout << " \n  Enter  an alphabit character  : " ;
        cin >> chr ;
        if ( chr == 'A')
        cout << " \n  You Entered A ";
        else
        cout << " \n  You Entered a Wrong Character  ";
        return 0 ;
}
```

Output

```
Enter  an alphabit character  :  A
You Entered A

Enter  an alphabit character  :  X
Wrong Character
```

بېلګه 7.5: لاندې پروګرام د if–else د کارولو طریقه ښیي.

```
// Program to illustrate the usage of if ( ) - else
#include  <iostream.h>
#include <stdlib.h>    //srand , rand
#include <time.h>      // time
main ( )
{
        int magic, guess ;
        srand (time (NULL));   //initialize random seed
        magic =rand() % 5;    // generate random number between 1 to 5
        cout << " \n Guess the magic number : " ;
        cin >> guess ;
        if (guess == magic )
        cout << "\n YOU got it ******* Right ******* " << magic;
        else
        cout << "\n Sorry ! Better luck next time. " << magic;
        return 0;
}
```

Output

```
Guess the magic number:  2
Sorry ! Better luck next time.  1
```

بېلګه :8.5 لاندې پروګرام داخل شوی عدد ارزوي چې جفت دی که طاق.

```
// Program to determine if a given number is even or odd
#include  <iostream.h>
main ( )
{
        int num, remain ;
        cout << " \n Enter the number to be tested  :" ;
        cin >> num ;
```

```
        remain = num % 2 ;
        if (remain == 0 )
        cout << num << "  is Even" ;
        else
        cout << num << " is Odd" ;
        return 0;
}
```

Output

```
Enter the number to be tested : 1279
1279 is Odd
Enter the number to be tested : 38
38 is Even
Enter the number to be tested : 0
```

بېلګه :9.5 لاندې پروګرام دا ارزوي چې داخل شوی کال کبیسه دی که نه.

```
/* Program to check whether a given year is a leap year */
#include <iostream.h>
main ( )
{
        int yr, rem_4, rem_100, rem_400 ;
        cout << " \n Enter the 4 digit year to be checked  :" ;
        cin >> yr ;
        // to find the remainders of division by 4, 100 & 400
         rem_4 = yr % 4 ;
         rem_100 = yr % 100 ;
         rem_400 = yr % 400 ;
         if ((rem_4 == 0 && rem_100!=0) || rem_400 ==0)
        cout << "\n is a leap year " << yr ;
        else
        cout << "\n is not a leap year ok !" << yr ;
        return 0;
}
```

Output

```
Enter the 4 digit year to be checked : 2004
2004 is a leap year
Enter the 4 digit year to be checked : 1998
1998 is not a leap year ok !
Enter the 4 digit year to be checked : 3000
3000 is not a leap year ok !
```

بېلګه :10.5 لاندې پروګرام د داخل شوي عدد مطلق قیمت (absolute value) محاسبه کوي.

```
// Program to calculate absolute value of an integer
#include <iostream.h>
main ( )
{
        int num ;
        cout << " \n Type a number :" ;
        cin >> num;
        if (num < 0)
        {
          num = - num ;
          cout << "\n The absolute value is " << num ;
        }
        else
          cout << "\n The absolute value is " << num ;
          return 0 ; }
```

Output

```
Type a number : -456
The absolute value is 456
Type a number : 360
The absolute value is 360
Type a number : 0
```

```
        The absolute value is 0 (how ? )
```

بېلګه :11.5 لاندې پروګرام دوه داخل شوي عددونه په نزولي (Descending) ترتیب ښیي.

```
// Program to display  2 numbers in descending order
#include  <iostream.h>
main ( )
{
        int num1, num2 ;
        cout << " \n Enter 2 numbers :" ;
        cin >> num1 >> num2;
        cout <<  " Number in Descending order are : " ;
        if (num1 >= num2)
            cout << "\n " << num1 << num2 ;
        else
            cout << "\n " << num2 << "  " << num1 ;
            return 0 ;  }
```

Output

```
        Enter 2 numbers : 25  31
        31    25
        Enter 2 numbers : -40  10
        10   - 40
```

بېلګه :12.5 لاندې پروګرام د 4 داخل شوو عددونو څخه تر ټولو لوی او دوهم لوی عدد پیدا کوي.

```
/* Program to find the largest & second largest of 4 numbers and their average */
#include  <iostream.h>
main ( )
{
        int m1, m2, m3, m4, first, second ;
        float avg ;
        cout << " \n Enter the marks obtained in 4 papers  :" ;
        cin >> m1 >> m2 >> m3 >> m4;
        if (m1 > m2)
```

```
            first = m1, second = m2 ;
        else
            first = m2, second = m1 ;
        if (m3 > first)
            second = first, first = m3 ;
        else if ( m3 > second )
            second = m3 ;
        if (m4 > first)
            second = first, first = m4 ;
        else if ( m4 > second )
            second = m4 ;
        cout << "\n " << "Largest marks " << first ;
        cout << "\n " << "Second Largest marks " << second ;
        avg = ( first + second ) / 2.0 ;
        cout << "\n " << " Average of best two marks = " << avg ;
        return 0;
}
```

Output

```
Enter the marks obtained in 4 papers : 56  78  40  48
Largest marks = 78
Second Largest marks = 56
```

بېلګه :13.5 لاندې پروګرام په 4 مضامینو کې د هر مضمون نتیجه ښیي.

```
/* Program to declare result in each subject in 4 papers */
#include <iostream.h>
main ( )
{
    int m1, m2, m3, m4;
    cout << " \n Enter the marks in 4 papers \n " ;
    cin >> m1 >> m2 >> m3 >> m4;
    if (m1 > 59)
```

```
        cout << " FIRST CLASS in paper 1 \n";
    else if ( m1 > 39 )
        cout << " SECOND CLASS in paper 1 \n";
    else
        cout << " Fail in paper 1 \n" ;

    if (m2 > 59)
        cout << " FIRST CLASS in paper 2 \n";
    else if ( m2 > 39 )
        cout << " SECOND CLASS in paper 2 \n";
    else
        cout << " Fail in paper 2 \n" ;

    if (m3 > 59)
        cout << " FIRST CLASS in paper 3 \n";
    else if ( m3 > 39 )
        cout << " SECOND CLASS in paper 3 \n";
    else
        cout << " Fail in paper 3 \n" ;

    if (m4> 59)
        cout << " FIRST CLASS in paper 4 \n";
    else if ( m4 > 39 )
        cout << " SECOND CLASS in paper  4  \n";
    else
        cout << " Fail in paper 4  \n" ;
        return 0;
}
```

Output

```
Enter the marks in 4 papers :
38  40   24   60
```

```
    Fail in paper  1
SECOND CLASS in Paper  2
FAIL in paper  3
FIRST CLASS in paper  4
```

بېلګه :14.5 لاندې پروګرام په if–else شرطي دستور کې د (char) کارول ښيي.

```
// Program to show usage of char in if ( ) statement
#include  <iostream.h>
#include  <stdio.h>
main ( )
{
        char wish;
        cout << " \n Want to got to movie Y/ N ?:" ;
        wish = getchar( );
        if (wish == 'Y' || wish == 'N')
            cout << "\n " << " Go and get ready quickly ! " ;
        else
            cout << "\n " << " Go and start studying ! " ;
            return 0;
}
```

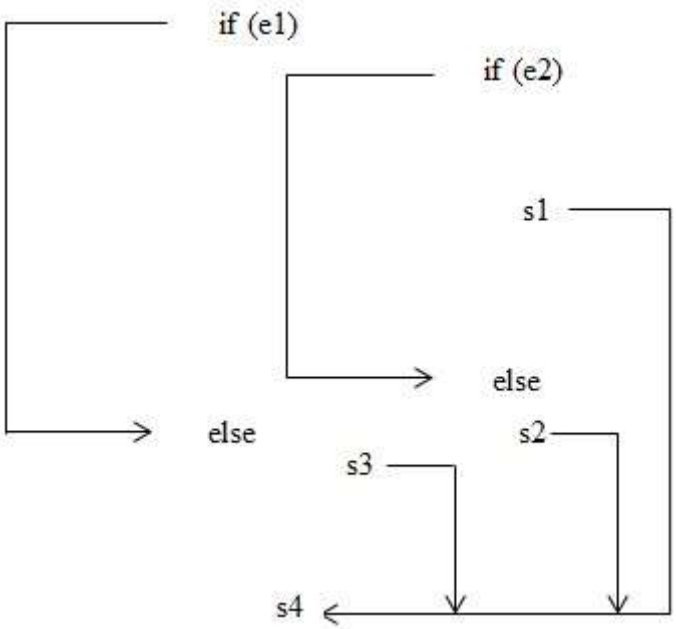
Output

```
Want to go to movie Y / N ? Y
Go and get ready quickly !
Want to go to movie Y / N ? n
Go and start studying !
Want to go to movie Y / N ? y
Go and start studying !
Want to go to movie Y / N ? A
Go and start studying !
```

۱.۲۱یو شرط د بل شرط دننه(Nested if Statement)

کله چې یو if د بل if په بلاک کې راشي، داسې شرطي بیانیه ته Nested if ویل کېږي.

د Nested if عمومي بڼه په لاندې ډول ده.(Soulié, 2007)



شکل :3.5 د Nested if شرطي بیانیې فلوچارت

دلته e1 او e2 شرطونه دي اوS1، S2 ، S3 او S4 بیانیې دي.

- که شرط e1 سم وي، نو وروسته شرط e2 کتل کېږي .که e2 هم سم وي، نو S1 اجرا کېږي او کنټرول S4 ته ځي .که e2 ناسم وي، نو S2 اجرا کېږي او وروسته S4 اجرا کېږي.
- که شرط e1 ناسم وي، نو S3 اجرا کېږي او وروسته S4 اجرا کېږي.
- کېدای شيS1 ، S2 او S3 نور else–if بیانیې هم ولري، دې حالت ته Multi–Layer Nestingویل کېږي.

بېلګه :15.5 لاندې پروګرام د nested if د کارولو طریقه ښيي.

```
// Program to show usage of nested if ( )
#include <iostream.h>
main ( )
{
        float hrs;
        cout << " \n  Type the line in hours :" ;
        cin >> hrs ;
        if (hrs >= 0.0 && hrs < 12.0)
            cout << "\n " << " Good Morning " ;
        else
            if (hrs >= 12.0 && hrs < 17.0 )
            cout << "\n " << "  Good Afternoon" ;
        else
            if (hrs >= 17.0 && hrs<21.0)
            cout <<"\n Good Evening ";
        else
            if (hrs >= 21.0 && hrs <24.0)
            cout << "\n Good Night ";
          else
            cout <<"\n Wrong Input ";
            return 0;
}
```

Output

```
Type the line in hours : 20.6
Good Evening
Type the line in hours : 5.0
Good Morning
Type the line in hours : 22
Good Night
Type the line in hours : 15
Good Afternoon
Type the line in hours : 50
Wrong Input
```

بېلګه :16.5 لاندې پروګرام د nested if په مرسته د 3 داخل شوو عددونو څخه لوی عدد پیدا کوي.

```
// Program to find the largest of three numbers  using nested  if ( ) ... else
#include  <iostream.h>
main ( )
{
        int a, b, c, large ;
        cout << " \n  Enter 3 numbers :" ;
        cin >> a >> b >> c ;
        if ( a > b )
                {if (a > c)
                        large = a;
                 else
                        large = c;
        else
                {if (b>c)
                        large = b;
                 else
                        large = c;
       cout << "\n Largest Number is " << large;  }
```

```
Output
        Enter 3 numbers :  22   6   11
        Largest Number is 22
        Enter 3 numbers :  10   50   42
        Largest Number is 50
        Enter 3 numbers :  13   6   80
        Largest Number is 80
```

بېلګه :17.5 لاندې پروګرام داخل شوی کرکټر لولي او معلوموي چې آیا هغه د الفبا حرف دی، عدد (digit) دی او که ځانګړی سمبول دی

```
   /* Program to read a character and determine whether it is an alphabet, digit or a
special character */
   #include  <iostream.h>
```

دا پروګرام د لوی او کوچنیو حروفو لپاره کارول کېږي .کله چې ch داسې ارزښت واخلي چې نه حرف وي او نه عدد، نو هغه د ځانګړي سمبول په توګه پېژندل کېږي.

بېلګه :18.5 لاندې پروګرام د دویم درجې معادلې جذرونه پیدا کوي.

```
/* Program to find the roots of a quadratic equaiton */
#include  <iostream.h> #include
<math.h>
main ( )
{
        float a, b, c,d;
```

```
float x, x1, x2;
float rpart, ipart;
cout << " \n  Enter 3 numbers  :" ;
cin >> a >> b>> c ;
// solution to equation of the form bx + c = 0
if (a==0)
 {
    x = - b / c ;
     cout << "\n Only root " << x ;
 }
 // Calculate discriminant  d
d = b * b - 4 * a * c;
 // Solution with real and distant roots
if (d > 0)
 {
    cout <<"\n  Real and district roots are : " ;
     x1 = (-b + sqrt (d)) / (2*a) ;
     x2 = (-b - sqrt (d)) / (2*a) ;
     cout <<"\n x1 = " <<x1  << "\n" <<" x2 =" << x2;
 }
 else
if (d ==0)
 {
   // Solution with repeated roots
    cout <<"\n  Repeated Roots are  : " ;
     x1 = -b  / (2*a) ;
     x2 = x1 ;
     cout <<"\n x1  & x2  = " <<x1 ;
 }
 else
       {
```

```
// Solution with complex roots
d = sqrt(abs(d));
rpart = -b / (2 * a) ;
ipart = d / (2 * a );
cout <<"\n Complex roots are : " ;
cout << "\n x1 = " << rpart << " + i " << ipart; // Real part of a complex root
cout << "\n x2 = " << rpart << " + i " << ipart; // Imaginary part of a
complex root
}
x = - b / c ;
cout << "\n Only root " << x ;
}
```

Output

```
Enter 3 numbers : 1 4 1
Real and district roots are :
x1 = -0.267
x2 = -3.732
Enter 3 numbers : 4 4 1
Repeated roots are :
x1 & x 2 = -0.5
Enter 3 numbers : 1 2 3
Complex roots are :
x1 = -1 + i 1.41421
x2 = -1 + i 1.41421
```

مثال 19.5: لاندې پروګرام 2 عددونه د کیبورډ څخه اخلي او د داخل شوي علامې(+ ،- ،* ، /)مطابق عمل ترسره کوي.

```
// Program to simulate the 4 arthmetic operations +, -, *, /
// for 2 real number depending on the operators
#include <iostream.h>
main ( )
{


    float num1, num2 ;
    char op;
    cout << " \n  Enter an operator  :" ;
    cin >> op;
    cout << " \n  Enter 2 numbers :" ;
    cin >> num1 >> num2 ;
    if (op=='+')
        cout << "\n Sum = " << num1 + num2 ;
    else
    if (op == '-')
        cout  <<" \n Difference =  " << num1 – num2 ;
    else
    if (op =='*')
        cout  <<" \n Product ==  " << num1 * num2 ;
    else
    if ((op == '/') && num1 != 0)
        cout  <<" \n Quotient  =  " << num1 /  num2 ;
    else
        cout <<" \n Error in Input ";
        return 0;
}
```

Output

```
 Enter an operator : +
Enter 2 nummbers : 5  4
Sum = 9.00
Enter an operator : -
Enter 2 nummbers : 5  8
Difference  = -3.00
Enter an operator : *
Enter 2 nummbers : 12.4  6
Product = 74.40
Enter an operator : /
Enter 2 nummbers : 12  4
Quotient = 3.00
Enter an operator : &
Enter 2 nummbers : 2  3
Error in Input
```

مثال 20.5: لاندې پروګرام د داخل شویو 3 عددونو ترمنځ تر ټولو لوی او دوهم لوی عدد پیدا کوي

```
/* Program to find the  largest & second largest of 3 numbers */
#include <iostream.h>
main ( )
{
      int a, b, c, first, second ;
      char op;
      cout << " \n  Enter 3 numbers  :" ;
      cin >> a >> b >> c ;
      if (a > b)
          if (a>c)
          {
            first = a ;
            if (b > c)
                second = b ;
            else
                second = c;
          }
          else
          {
            first = c;
            second = a;
          }
      else
          if (b > c)
          {
            first = b;
            if (a>c)
                second = a;
            else
                second = c;
```

```
                }
            else
                {
                 first = c;
                 second = b;
                }
    cout << "\n Largest is = " << first << "\n Second largest is = " <<second;
    }
```

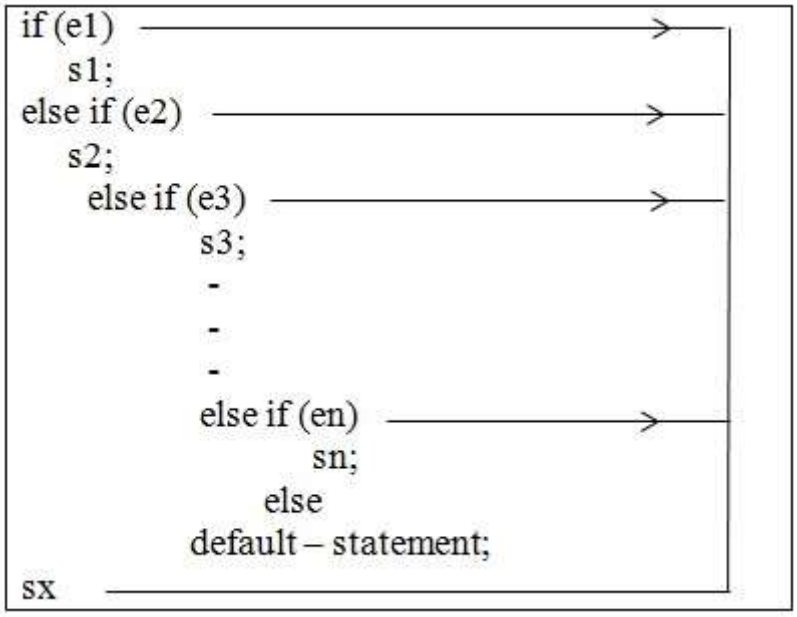
Output

```
Enter 3 numbers  15   2   10
Largest is  15
Second Largest is 10
Enter 3 numbers  28    65   5
Largest is  65
Second Largest is 28
Enter 3 numbers  79   12   6
Largest is  79
Second Largest is 12
```

۱.۲۲ بیان THE ELSE-IF()

دا دشرطي بيان بله پرمختللي بڼه ده چې د هغي په کارولو سره مونږ کولای شو NESTED ELSE-IF(یو بل کی د ننه شرطونه) تشریح کړو په حقیقت کی دا له IF-ELSEجوړښت څخه جوړ دی چی هر په کی هر ELSE خپله یوه IF بیان هم لري.

IF-ELSEعمومي شکل په لاندی ډول دی:



شکل 4.5 فلوچارټ د رشطي بيانيیif - else statement

- sx عادي بيانيه ده او default بيانيي sn, s2, s1 او رشطونه en ..e3, e2, e1 دي
- په دې فلوچارټ کې رشطونه له پورته څخه ښکته چک کيږي
- په لومړي مرحله کې e1 امتحان کيږي، که سم وي نو s1 اجرا کيږي او کنټرول sx ته انتقاليږي، که نه نو e2 امتحان کيږي، که سم وي s2 اجرا کيږي او کنټرول sx ته انتقاليږي او ټول پاتې بيانيې له پامه غورځول کيږي

•که ټول رشطونه ناسم وي en ... e1, e2 ، په دي صورت کي عادي بيانيي (Default Statement) اجرا کيږي او وروسته sx بيانيه اجرا کيږي

مثال 21.5: لاندی پروګرام د داخل شوي عدد نښه (علامه) پيدا کوي

```
/* Program to find sign of a number */
#include <iostream.h>
main ( )
{
        int num , sign;
        cout << " \n  Enter a number please :" ;
        cin >> num;
        // check if number is negative
        if (num< 0)
                sign = -1 ;
        else if (num==0)
                sign = 0;
         else // number must be positive
                sign = 1;
         cout <<" sign = " << sign;
          return 0;
}
```

Output

```
Enter a number please :-259
sign = -1
Enter a number please : 0
sign = 0
Enter a number please :5
sign = 1
```

يو استاد دري امتحانونه(امتحان1، امتحان 2او نهايي امتحان)له محصلينو اخلي چې هر يو 50 نمري لري.

له امتحان 1او امتحان 2څخه تر ټولو لوړي نمري د هر محصل لپاره انتخابيږي او د نهايي امتحان له نمري سره جمع کيږي.

يادونه :ټولي نمري له 100 څخه حسابيږي.

پاتي منفي عددونه په لاندي ډول محاسبه کيږي

Marks	Grade
0 - 39 40 - 49 50 - 59 60 - 74 75 - 100	F D C B A

د پروګرام نويسي د ژبي په کارولو سره په C++ پروګرام داسي وليکئ چې د دري امتحانونو نمري د کيبورډ څخه واخلي او وروسته د نهايی نمرې محاسبه کړي او په سکرين يې ښکاره کړي.

مثال 22.5: لاندی پروګرام د ورکړل شوي قاعدي مطابق د يو محصل نهايی نمره محاسبه کوي او په سکرين يی

```
//Program to print the grade of a student
#include  <iostream.h>
main ( )
{
        int test1, test2, final, marks;
        char grade;
        cout << " \n  Enter marks in Test1, Test2, final :" ;
        cin>> test1 >>test2 >>final;
        // calculate total marks out of 100
        if (test1>test2)
                marks = final + test1 ;
        else
                marks = final + test2 ;
         // assigning grades
        if (marks >= 0 && marks <40)
                grade = 'F' ;
        else if (marks <50 )
                grade = 'D';
        else if (marks <60 )
                grade = 'C';
```

```
        else if (marks <75 )
                grade = 'B';
        else if (marks <= 100 )
                grade = 'A';
        else
                grade = 'X' ;
        if (grade != 'X')
        cout <<" Grade is = " << grade ;
        else
        cout << "Error in input !!";
        return 0;
  }
```

Output

```
Enter marks in Test1, Test2, final : 50  50  25
Grade is  A
Enter marks in Test1, Test2, final : 20  10  30
Grade is  C
Enter marks in Test1, Test2, final : 20  10  15
Grade is  F
Enter marks in Test1, Test2, final : 40  39  80
Error in input !!
```

یو کمپنی 4 کټګوري کارمندان لري .د هغوی د معاشاتو (salary) د ورکړې قاعده په لاندې ډول ده:

Catagory1 → 30% of salary

Catoagory2 → 20% of salary

Catoagory3 → 15% of salary

Catoagory4 → 10% of salary

مثال **23.5:** د ورکړل شوی قاعدی مطابق داسی پروګرام ولیکئ چې د کارمند معاش او کټګوري په نظر کی ونیسي او د کرایی مقدار معلوم کړي

```
// Program to calculate rent allowance besed on employee category
#include  <iostream.h>
main ( )
{
        int cat ;
        float sal, rent ;
        cout << " \n  Enter the category and salary of employee  :" ;
        cin >> cat >> sal ;
        if (cat == 1)
                rent = sal * 0.3 ;
        else if (cat == 2)
                rent = sal * 0.2 ;
        else if (cat == 3)
                rent = sal * 0.15 ;
        else if (cat == 4)
                rent = sal * 0.1 ;
        else
                {
                cout << "\n Wrong Input " ;
                }
        cout << "\n The rent allowance = " << rent ;
        return 0;
}
```

Output

```
Enter the citatory and salary of Employee : 1   35000
The rent allowance = 10500.00
Enter the citatory and salary of Employee :4   10000
The rent allowance = 1000.00
Enter the citatory and salary of Employee : 23   3000
Wrong Input
```

۱.۲۳ شرطی بیانیهswitch

د switch څخه استفاده هغه وخت اغیزمنه وي چې په پروګرام کې څو بیانیې موجودې وي او مونږ وغواړو چې د ځانګړي شرط له مخې یوه ځانګړې بیانیه اجرا شي .د داسې مسئلې لپاره مونږ کولای شو else-if هم وکاروو، خو کله چې شرطونه ډېر شي نو else-if پېچلی کېږي، ځکه تاسو اړتیا لرئ چې د مسئلې د حل لپاره ډېرې بیانیې ولیکئ .په داسې حالت کې غوره ده چې د switch شرطی بیانیه وکارول شي.(Backman, 2012)

د switch بیانیې عمومي شکل په لاندې ډول دی:

```
switch ( variable or expression )
{
case Val1 : s1;
break;
case Val2 : s2;
break;
…….. :
……. :
case Valn : sn
: break;
default : sd
break;
}
statement – x ;
```

شکل 5.5 د switch بیانیه

• switch یو کلیدي دستور (keyword) دی چې د قوسونو (Parenthesis) دننه لیکل شوی متحول (variable) یا عبارت (expression) چک کوي switch .له ځان سره د case قیمتونو لست لري (case1, case2, case3) چې د case labels په نوم یادیږي.

• د متحول یا عبارت (expression or variable) قیمت باید عدد تام (int) یا یو حرف (character)وي .د case label قیمتونه لکه (case1, case2, case3 ...) باید ثابت وي او نتیجه یې هم عدد تام وي.

• د هر case قیمت باید ځانګړی (unique) وي .قیمتونه په هر ترتیب (order) کې راتلای شي.

• بیانیه S1 اوS2 کولای شی یوسطر یا له یوڅخه زیات سطرونه ولري.

• د هر label case په آخر کې باید (:) colon ولیکل شي او ټولcase ونه د یو pair of curly braces (braces)دننه لیکل کیږي.

• کله چې بیانیه switch اجرا شي، د مربوط case ټول بیانیې هم اجرا کیږي .که د switch ارزښت له هیڅ case سره برابر نه وي، نو د default ټول بیانیې اجرا کیږي.

• د default لیکل اختیاري دي .که وجود ونه لري نو x بیانیه اجرا کیږي.

• د هر case په آخر کې د break لیکل ضروري دي، تر څو د هغه case اجرا پای ته ورسوي .که break ونه لیکل شي، نو ټولcase ونه یو په بل پسې اجرا کیږي.

•د اعشاري (decimal) ارزښت کارول په case label کې جواز نه لري.

switch •کېدای شي د nested شکل په توګه هم وکارول شي.

لاندې څو switch پروګرامونه قرار لري:

مثال 24.5: لاندې پروګرام د Julian Date ښيي

```
/* Program to find the Julian Date */
#include <iostream.h>
main ( )
{
int dd, mm, yy;
int leap, julian = 0;
cout << " \n Enter a valid date :" ;
cin >> dd >> mm >> yy;
// To check for leap year
leap = (yy % 4 == 0 && yy % 100 != 0) || yy % 400 == 0;
julian + dd;
switch ( mm - 1) {
case 11: julian = julian + 30 ;
case 10: iulian = iulian + 31 ;
```

```
case 9: julian = julian + 30 ;
case 8: julian = julian + 31 ;
case 7 : julian = julian + 31 ;
case 6 : julian = julian + 30 ;
case 5 : julian = julian + 31 ;
case 4 : julian = julian + 30 ;
case 3 : julian = julian + 31 ;
case 2 : if ( leap );
julian += 29;
julian += 28;
case 1 : julian = julian + 31;
}
cout << "\n Julian day is : " << julian ;
 return 0;
}
```

Output

Enter a valid date : 23 4 2005

Julian day is : 119

مثال 25.5: لاندې پروګرام 2 عددونه د کیبورډ څخه اخلي او د داخل شوي علامې (+، -، *، /) مطابق عمل ترسره کوي

```
/* Program to stimulate the 4 arithmetic operations +, -, *, / for 2 real numbers
depending on the operator */
#include <iostream.h>
main ( )
{
        float num1, num2;
        char op;
        cout << " \n Enter an operator :" ;
        cin >> op;
        cout << " \n Enter 2 numbers : ";
        cin >> num1 >> num2 ;
```

```
switch ( op)
{
        case '+':
                cout << " \n Sum = " << num1 + num2 ;
                break;
        case '-':
                cout << " \n Difference = " << num1 - num2 ;
                break;
        case '*':
                cout << " \n Product = " << num1 * num2 ;
                break;
        case '/':
                if (num1 != 0)
                cout << " \n Quotient = " << num1 / num2 ;
                else
                cout << " \n Division by zero ? ";
                break;
        default :
                cout <<" \n Error in Input " ;
                return 0;
  }
  }
```

Output

```
Enter an operator :   +
Enter  2  numbers  :  6   9
Sum = 15.00
```

مثال 26.5: د ورکړل شوي قاعدي مطابق داسي پروګرام ولیکئ چې د کارمند معاش او کټګوري په نظر کې ونیسي او د کرایي اندازه معلومه کړي

```
/* Program to calculate rent allowance based on employee atagory */
#include <iostream.h>
main ( )
{
        int cat ;
        float sal, rent;
        cout << " \n Enter the category and salary of Employee :" ;
        cin >> cat >> sal;
        switch ( cat )
        {
                case 1:
                        rent = sal * 0.3;
                        break;
                case 2:
                        rent = sal * 0.2;
                        break;
                case 3:
                        rent = sal * 0.15 ;
                        break;
                case 4:
                        rent = sal * 0.1;
                        break;
                default :
                        cout <<"\n Wrong Input ";
                        break;
        }
                cout <<" \n The rent allowance = " << rent;
                return 0;
}
```

Output

```
Enter the category and salary of Employee : 2    36000
The allowance  = 7200.00
```

مثال 27.5: لاندې پروګرام د دويمې درجې (Quadratic) معادلې ريښې پيدا کوي او په سکرين يې ښيي

```
/* Program to find the roots of a quadratic equation using switch statement */
#include <iostream.h>
main ( )
{
int flag;
float a, b, c, d;
float x, x1, x2;
float rpart, ipart;
cout << " \n Enter 3 numbers :" ;
cin >> a >> b >> c;
// solution to equation of the form bx + c = 0
if (a==0)
{
x = -b / c;
cout << "Only root = " << x;
}
d = b * b - 4 * a * c ;
if ( d > 0 )
flag = 1;
else if ( d == 0)
flag = 2;
else
flag = 3;
// Solution using switch-case default statement
switch ( flag )
```

```
{
case 1:
cout << "\n Real and distinct roots are : " ;
x1 = (-b + sqrt (d)) / (2 * a) ;
x2 = (-b - sqrt (d)) / (2 * a) ;
cout <<"\n x1 = " << x1 << "\n x2 = " << x2 ;
break;
case 2:
cout << "\n Repeated roots are : " ;
x1 = -b / (2 * a) ;
x2 = x1 ;
cout <<"\n x1 & x2 = " << x1 ;
break;
case 3:
d = sqrt (abs (d)) ;
rpart = -b / (2 * a) ; // real part of a complex root
ipart = -b / (2 * a) ; // imaginary part of a complex root
cout <<"\n complex root are :" ;
cout << "\n x1 = " << rpart;
cout << "\n x2 = " << ipart;
}
}
```

Output

```
Enter 3 numbers : 2 6 4
Real and distinct roots are :
x1 = -1.000
x2 = -2.000
```

۱.۲۴ شرطی عملګر (The Conditional Operator)

په ډېرو مواردو کې دا عملګر د دوه اړخیز تصمیم نیولو لپاره کارول کېږي .دا عملګر د سوالیه نښه ؟ او کولن (:) نښو له ترکیب څخه جوړ دی او درې قیمتونو (Operands) ته اړتیا لري . (Overland, 2013)

عمومي شکل یې په لاندې ډول دی:

```
Expression1 ? Expression2 ; Experssion3 ;
```

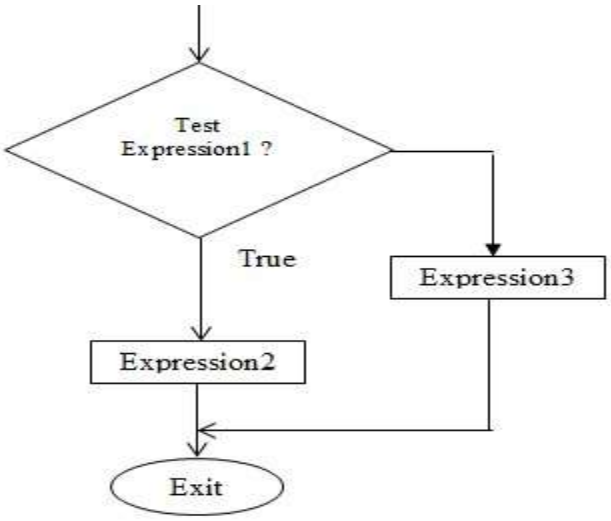
•لومړی شرط (Expression1) په لومړي قدم کې چک کېږي .که سم وي نو Expression2 اجرا کېږي او که شرط ناسم وي نو Expression3 اجرا کېږي.

•تل له دې دوو شرطونو څخه یوازې یو اجرا کېږي، امکان نه لري چې دواړه په یو وخت اجرا شي.

•تاسو کولی شئ د if-else دستور چې د conditional operator معادل دی، په لاندې ډول وکاروئ:

```
if (Expression1)
Expression 2;
else
Expression 3;
```

د شرطی عملګر (Conditional Operator) فلوچارټ په شکل 5 .6کې ښودل شوی دی:



شکل 6.5 د Conditional عملګر فلوچارټ

مثال 28.5: لاندې پروګرام د x ارزښت د کیبورډ څخه اخلي او لاندې معادلې ارزیابي کوي

```
/* Write a program to read the value of x and evaluate the following function
y = x + 15 for x > 0
y = 1.92x + 5 for x = 0
and y = 2x + 10 for x < 0 */
// Program to show usage of ? : operator
#include <iostream.h>
main ( )
{
float x, y;
cout << " \n Enter a value of x :" ;
cin >> x;
y = ( x > 0 ) ? x + 15 : ((x<0) ? 2 * x + 10 : 1.92 * x + 5 ) ;
cout << "\n When x = " << x <<" y = " << y;
return 0;
}
```

Output

Enter a value of x : 3

When x = 3, y = 18

Enter a value of x : -5

When x = -5, y = 0

Enter a value of x : 0

When x = 0, y = 5

يو د رخت پلورنځی د خپلو پيرودونکو لپاره لاندې تخفيفونه په نظر کې نيولي دي:

د پيرودو مقدار	**تخفيف**
Purchase Amount	**Discount**
0-100	5%
101-200	7.5%
201-300	10%
له 300څخه زيات	15%

برنامه وليکئ چې د تخفيف له وضع کولو وروسته خالص قيمت (Net Amount) وښيي.

مثال 29.5: پورته مسئله په لاندې ډول پروګرام شوې ده:

```
// A cloth showroom offers discount on parchase of items as follow:
#include <iostream.h>
main ( )
{
float amount, net;
cout << " \n Enter the purchase amount please :" ;
cin >> amount ;
if (amount <= 0)
{
cout << "\n Invalid amount ";
}
// computation of net amount
net = (amount <= 100 ) ? amount - amount * 0.05 :
(amount <= 200) ? amount - amount * 0.075 :
(amount <= 300 ) ? amount - amount * 0.01 :
amount - amount * 0.15;
cout << "\n The amount after discount = " << net;
return 0;
}
```

Output

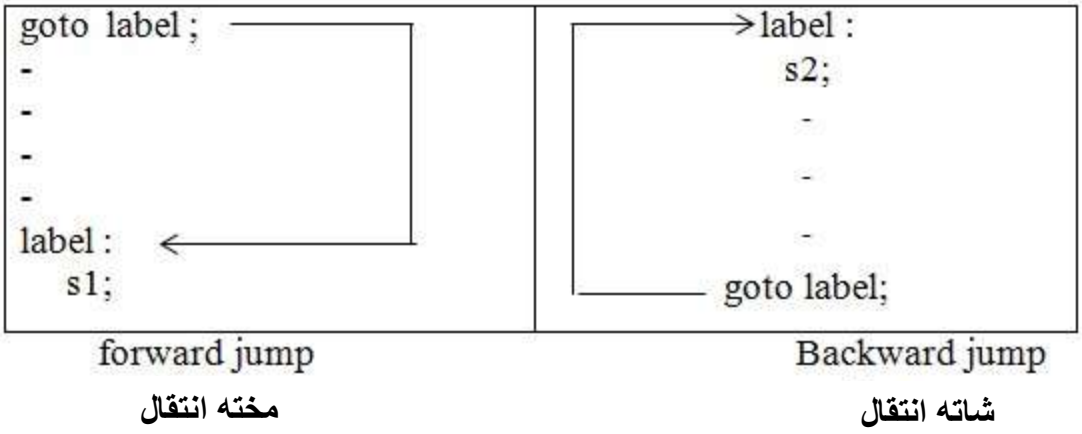
```
Enter the purchase amount please : 110
The amount after discount = 101. 75
Enter the purchase amount please : 542.9
The amount after discount = 461.47
```

بیانیه goto

د دې بیانیې په کارولو سره تاسو کولی شئ د پروګرام هره برخه چې وغواړئ اجرا کړئ .یادونه باید وشي چې goto یو شرطی بیانیه نه ده .عمومي شکل يې په لاندې ډول دی(عدلیار 1381) :

دلته label هغه نوم دی چې هغه ځای ته اشاره کوي چې پروګرام باید هلته لاړ شي.

- د لیبل نوم باید د متغیر (variable) د نوم د جوړولو ټول اصول مراعت کړي .لیبل کېدای شي حروف، شمېرې او انډر سکور ولري، خو لومړنی کرکټر يې باید حتماً حرف وي.
- د لیبل نوم او د هغې وروسته دوه نقطې (:) هغه ځای کې لیکل کېږي چې غواړئ پروګرام له هغه ځایه پیل شي.
- د goto او لیبل عمومي شکل په لاندې ډول دی:



شکل 7.5 د goto بیانیه

- کله چې لیبل د goto وروسته راشي، دې ته forward jump ویل کېږي، او کله چې لیبل د gotoڅخه مخکې اعلان شوی وي، دې ته backward jump ویل کېږي لکه څنګه چې په شکل 7.5کې ښودل شوي دي.

•هر لیبل باید په یو پروګرام کې ځانګړی (unique) وي، یعني دوه مختلفې بیانیې نشي کولی یو شان لیبل نوم وکاروي.

•د goto بیانیه د if شرط او لوپونو (loop) سره یو ځای هم کارول کېدای شي.

•کله کله goto د بې پایه تکرار (infinite loop) سبب هم ګرځي، چې دې حالت ته infinite loopingویل کیږي.

مثال 30.5: لاندې پروګرام د goto بیانیې د کارولو طریقه ښیي:

```
/ Program to show usage of goto function
#include <iostream.h<
main) (
{
float r, circum ;
first : cout << "\n Enter Radius ( enter 0 to exit; " : )
cin >> r;
if ( r != 0)
{
circum = 2 * 3.14159 * r;
cout <<"\n Circumference = " <<circum;
goto first ;
}
return 0;
}
```

Output

```
Enter radius (enter 0 for exit ) : 5
Circumference = 31.416
Enter radius (enter 0 for exit ) : 0
```

مثال 31.5: لاندې پروګرام هم د goto بیانیې د استعمال طریقه ښیي

```
// Program to show usage of goto statement
#include <iostream.h<
main) (
{
int a , b , big;
cout << "\n Enter a and b; "
cin >> a >> b;
if ( a > b(
{
big = a;
goto finish ;
}
if ( a <= b(
{
big = b;
goto finish;
}
finish:
cout << "\n Biggest of "<< a << " and " << b <<" is : " << big;
}
```

Output

```
Enter a and b : 14 38
Biggest of 14 and 38 is 38
```

مثال 32.5: لاندې پروګرام د () if او goto په کارولو سره داخل شوی عدد برعکس کوي

```
// Program to reverse a number using if ( ) and goto
#include <iostream.h<
main) (
{
int n, rem, rev = 0;
cout << "\n Enter a number; " :
cin >> n;
start:
if ( n == 0(
// Terminate the loop when n = 0
goto finish;
rem = n % 10;
rev = rev * 10 + rem;
n = n / 10;
goto start ;
finish:
cout << "\n The reverse number is : " << rev; }
```

Output

```
Enter a number : 1234
The reverse number is 4321
Enter a number : -8329
The reverse number is –9238
```

شپږم فصل

دا فصل د حلقې مفاهیم، د حلقو ډولونه او په پروګرامنګ کې د هغې اهمیت بیانوي چې یو په بل پسې په تفصیل سره تشریح کېږي.

کله چې وغواړو د یو پروګرام بیانیې څو ځله تکرار کړو، نو اړینه ده چې د حلقوي دستورونو څخه استفاده وکړو .لکه څنګه چې پوهېږو، کمپیوټر د دې توان لري چې د یو پروګرام مختلف عبارتونه په تکراري ډول اجرا کړي .دا ځانګړتیا موږ ته اجازه راکوي چې د اړتیا له مخې د پروګرام مختلف عبارتونه څو ځله تکرار کړو .که کمپیوټر دا خاصیت نه درلود، نو موږ به اړ وو چې د ځینو مسایلو د حل لپاره زرګونه کرښې (lines) کوډ ولیکو.

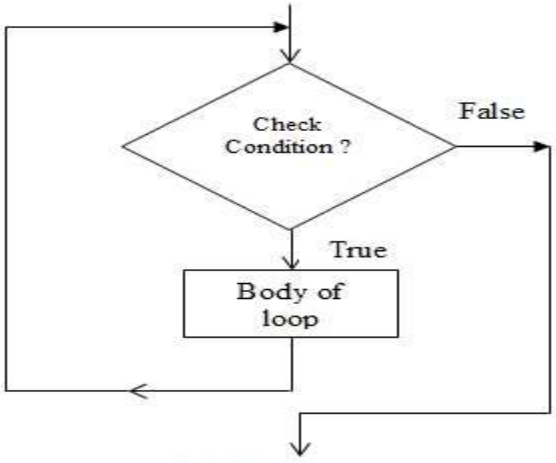
حلقوي جوړښت (Looping Structure) هغه جوړښت ته ویل کېږي چې پکې د پروګرام یو شمېر بیانیې تر هغه وخته تکرار اجرا کېږي څو تاکل شوی شرط پوره شي .حلقه (Loop) له دوو برخو څخه جوړه ده :د حلقې بدنه (Body of the loop) او کنټرول کوونکي بیانیې (Control statements).

کنټرول کوونکي بیانیه (Control Statement) شرط معاینه کوي چې پوره شوی که نه، او د هغه مطابق د حلقې په بدنه کې موجودې بیانیې اجرا کوي.(Horton, 2012)

کنټرول کوونکی جوړښت په دوو برخو وېشل کېږي:

a) Entry-controlled loops

b) Exit-controlled loops

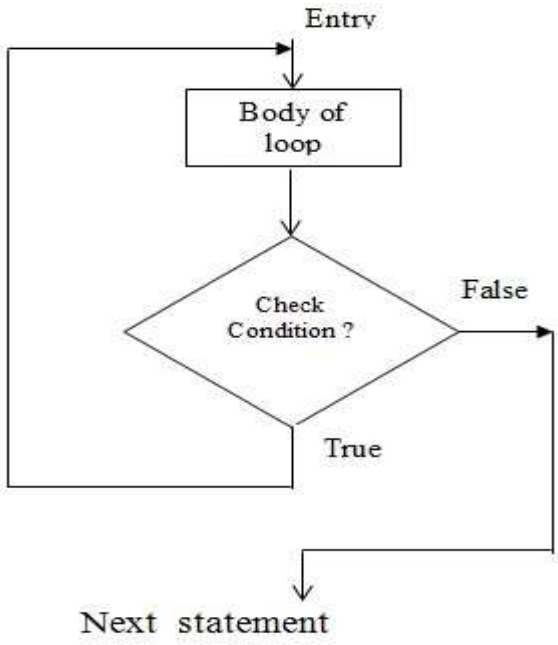


1.6 شکل فلوچارتEntry-controlled loop

په هغه حلقه کې چې شرط لومړی ارزول کیږي او وروسته د بیانیو مجموعه په تکراري ډول اجرا کیږي(test-Pre) ، دا ډول جوړښت د **د خول کنټرول شوې حلقه** (Entry-controlled loop)بلل کیږي .په دې حالت کې د حلقې بدنه یوازې هغه وخت اجرا کیږي چې شرط سم وي، که نه نو د حلقې دننه بیانیې اجرا نه کیږي او کنټرول د حلقې له بدنې څخه بهر بیانیو ته سپارل کیږي .نو په دې حالت کې د پروګرام بدنه تر هغه وخته اجرا کیږي چې شرط سم وي او کله چې شرط ناسم شي، حلقه پای ته رسیږي.

په هغه حلقه کې چې شرط په پای کې معاینه کیږي، دا جوړښت د خروج کنټرول شوې حلقه (Exit-controlled loop)بلل کیږي .په دې حالت کې د حلقې بدنه لږ تر لږه یو ځل اجرا کیږي، ځکه چې شرط په پای کې وي او هلته ارزول کیږي .(test-Post) په دې حالت کې هم د حلقې بدنه تر هغه وخته تکرار اجرا کیږي چې شرط سم وي او کله چې شرط ناسم شي، د حلقې بدنه بیا نه تکرارېږي.

شکل 2.6 **د خروج کنټرول شوې حلقې** د کار طریقه ښیي.



Entry – controlled Loop فلوچارټ

Exit-controlled loop شکل فلوچارټ 2.6

که یو پروګرام داسې جوړ شوی وي چې شرط هیڅکله ناسم نه شي، نو پروګرام په نه ختمېدونکي ډول اجرا کېږي چې دې ډول حلقې ته بېپایانه حلقه (Infinite loop) ویل کېږي .له همدې امله اړینه ده چې د حلقې د تعریف پر مهال احتیاط وشي څو حلقه بېپایانه بڼه غوره نه کړي (Kolev & Nakov, 2013).

په عمومي ډول، د حلقوي جوړښت لاندې مرحلې لري:

a) د محاسب (counter) لپاره د پیل ارزښت ټاکل(Initializing the counter)

b) د حلقې په بدنه کې د بیانیو اجرا کول (Executing the statements in the domain of the loop)

c) د پروګرام د شرط ارزونه؛ یعنې دا چې داخل شوی ارزښت شرط پوره کوي که نه (Testing the condition in the control statement)

d) د هر ځل اجرا وروسته د محاسب زياتول(Incrementing the counter)

Chapter 22د حلقو ډولونه(Types of Looping Constructs)

په C++ ژبه کې دري ډوله حلقوي جوړښتونه شتون لري:(Glassborow, 2004)

a) The for (; ;) Loop

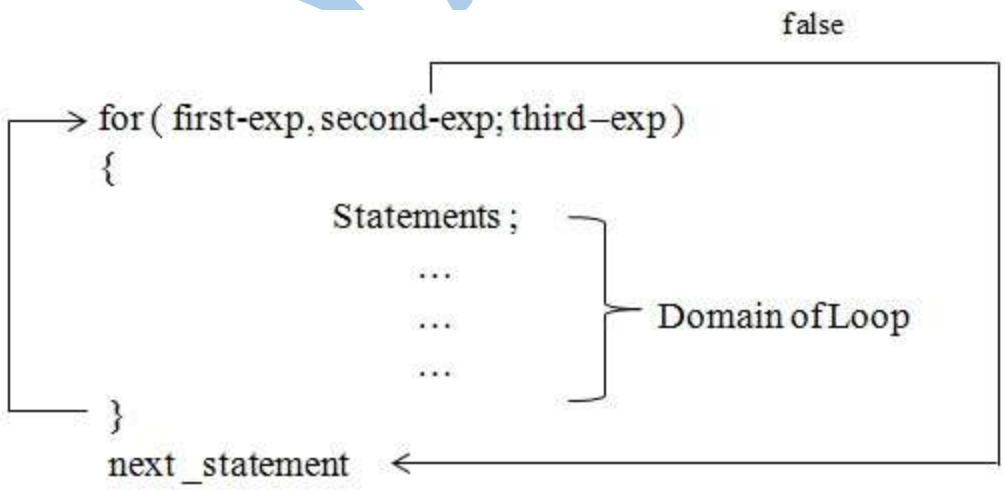
b) The while ( ) Loop

c) The do – while ( ) Loop

1.25.1بيانيه THE FOR (; ;)

for د C++ ژبې له مشهورو حلقو څخه ده چې په پروګرامونو کې ډېر کارول کېږي .په دې حلقه کې هم شرط په پيل کې ارزول کېږي (test-pre) او که سم وي، د حلقې بدنه اجرا کېږي (body of the loop).

د for عمومي شکل په لاندې ډول دی:



شکل 3.6 د for حلقې عمومي بڼه

-1متحول exp_first لومړنی ارزښت ساتي چې حلقه له هغه څخه پیل کیږي.

-2متحول exp_second د حلقې شرط دی چې که سم وي، د حلقې بدنه اجرا کیږي.

-3متحول exp_third د زیاتولو (increment) یا کمولو (decrement) ارزښت په ځان کې ساتي.

لاندې مثال ته پام وکړئ:

```
for ( k = 1 ; k <= 10 ; k ++ )
cout <<" \ n  " << k;
```

په دې مثال کې k له 1 څخه پیل کیږي .دوهمه برخه (k <= 10) شرط دی چې k تر 10 پورې ارزښت اخیستلی شي .په درېیمه برخه کې k د هر ځل اجرا وروسته 1 زیاتیږي .نو ویلی شو چې پروګرام له 1 څخه تر 10 پورې شمېرې اخلي او په سکرین یې ښکاره کوي، او وروسته حلقه ختمیږي.

```
1
2
3

10
```

که پورته مثال داسې ولیکل شي:

```
for ( k = 10 ; k > 0 ; k -- )
cout <<" \ n  " << k;
```

نتیجه به داسې وي:

```
   10
9
8
.                                                    .
.
1
```

-4د لومړۍ او دوهمې برخې ترمنځ د سيمي کولن (;) کارول ضروري دي.

-5که د for په جوړښت کې کومه برخه موجوده نه وي، بيا هم اړينه ده چې د هماغې برخې سيمي کولن (;) موجود وي.

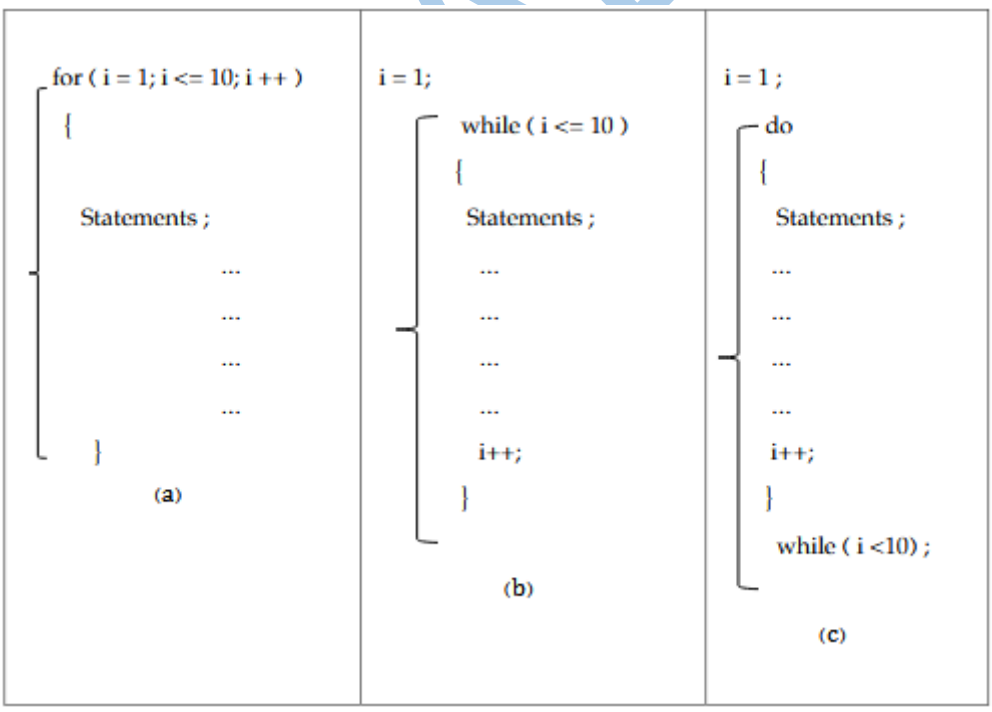
لاندې مثال ته پام وکړئ:

```
for ( ; I > 10 ; ++ )
```

دلته ليدل کېږي چې د for په جوړښت کې لومړۍ برخه نشته، خو سيمي کولن (;) پکې ليکل شوی دی.

-6د for بدنه کولی شي يو يا څو بيانيې ولري .که له يوې څخه زياتې وي، نو بايد د لويو قوسونو { }دننه وليکل شي.

-7د for حلقې بڼه والی د while او do-while په نسبت دا دی چې د for ټولې برخې په يوه کرښه کې ليکل کېږي (د for بيانيه نسبت نورو ته ساده ده.)



شکل 4.6 د درې واړو حلقو د جوړښت پرتله

-9د سيمي کولن (;) عملګر په (; ;) for بيانيه کې د Comma Operator in په نوم کارول کېږي.

په (; ;) for بيانيه کې تاسو کولی شئ څو متحولونه په يو وخت کې لومړني ارزښتونه ورکړئ، په دې شرط چې ترمنځ يې د سيمي کولن عملګر وکاروئ .کله چې دا ارزښتونه سره جلا شي، اجرا يې له چپ څخه ښي ته ارزول کېږي.

لاندې د پروګرام برخه وګورئ:

```
 x = 2 ;
 for ( i = 1 ; i <= 10 ; i++)
 {
...
...
...
x ++ ;
}
```

تاسو کولی شئ پورته مسئله د کالن (comma) په کارولو سره داسې وليکئ:

```
for ( x =2; i = 1; i <= 10 ; x++, i++)
{
  ...
}
```

په دې مثال کې د for لومړۍ برخه (Initialization Section) او د زياتولو برخه (Incrementation Section)دواړه دوه متحولونه لري.

-10د for حلقې د ارزوني برخه (condition test) هم کولی شي څو شرطونه ولري چې په ګډه (combination) کارول کېږي .په لاندې مثال کې i د حلقې داخلي متحول دی او sum

خارجي متحول دی:

```
sum = 0;
for ( i = 1; i <= 10 && sum <= 100 ; i++ )
{
 sum = sum + i;
 cout << i << sum;
```

-11د for (; ;) حلقه د ځنډ (delay loop) حلقې په توګه هم کارول کېدای شي:

```
for ( x = 1; x <= 1500 ; x ++ ) ;
```

دا حلقه د وخت ځنډ (delay) رامنځته کوي او 1500 ځله پرته له دې چې کومه بیانیه اجرا کړي تکرارېږي .سیمي کولن (;) چې د for حلقې په پای کې راځي د خالي بیانیې (Null Statement)په نوم یادېږي .په دې حالت کې د C++ کمپایلر دا بیانیه تېروتنه نه ګڼي(یعني د forحلقې په پای کې د سیمي کولن شتون.)

مثال 1.6: لاندې پروګرام داخل شوی عدد معاینه کوي چې آیا عدد اولي (prime) دی که نه

```
/* program to check whether a number is prime or not */
#include <iostream.h>
main ( )
{
int n, i ;
cout << " \n Enter a positive number please : " ;
cin >>n;
if ( n <= 1)
{
cout << "\n The number is not prime : " << n;
exit(0);
}
for ( i = 2; i <= n/2; i++ )
{
if ( n % i == 0 )
{
cout <<"\n The number is not prime : " << n;
exit (0);
}
} // end of for ( )
cout <<"\n The number is prime : " << n;
return 0;
} // end of main ( )
```

Output

```
Enter a positive number please : 11
The number is prime : 11
Enter a positive number please : 10
The number is not prime : 10
```

مثال 2.6: لاندې پروګرام د لاندې سلسلې مجموعه محاسبه کوي او په سکرین يې ښيي

```
// Program to find the sum of the series 1 + x + x ^2 + …. + x ^ n
#include <iostream.h>
#include <math.h>
main ( )
{
float i, x , sum = 1 ;
int n ;
cout << " \n Enter the number of terms : " ;
cin >>n;
cout <<" Enter the value of x " ;
cin >> x;
for ( i = 1; i <= n ; i++ )
sum = sum + pow (x, i) ;
cout <<"\n Summation of series = " << sum ;
return 0;
}
```

Output

```
Enter the number of terms : 10
Enter the value of x : 2
Summation of series : 2047
```

مثال 3.6: لاندې پروګرام د لاندې سلسلې مجموعه محاسبه کوي

```
// Program to print the sum of the series S = 1 + 1/2 + 1/3 + ….+ 1/n
#include <iostream.h>
main ( )
{
int n, i ;
float sum = 0 ;
```

```
cout << " \n Enter value of n : " ;
cin >>n;
for ( i = 1; i <= n ; i++ )
sum = sum + 1.0 / i ;
cout <<"\n Summation = " << sum ;
return 0;
}
```

Output

```
Enter value of n : 10
Summation = 2.929
```

مثال 4.6: لاندې پروګرام داخل شوی عدد معاینه کوي چې آیا یو کامل عدد (perfect number) دی که نه؟

```
// Program to find whether a given number is a perfect number or not
#include <iostream.h>
main ( )
{
int i , n, sum = 1;
cout << " \n Enter a number : " ;
cin >>n;
for ( i = 2; i <= n / 2 ; i++ )
{
if ( n % i ==0 )
sum = sum + i ;
}
if ( n == sum )
cout <<"\n This is a perfect number " << n ;
else
cout <<"\n This is not a perfect number " << n;
return 0;
}
```

```
Output
Enter a number : 6
This is a perfect number : 6
Enter a number : 5
This is not a perfect number : 5
```

نکات چې د for (; ;) حلقې د استعمال پر وخت باید په نظر کې ونیول شي:

i . که د حلقې ابتدایي (initial) ارزښت د وروستي ارزښت څخه لوی وي، نو په دې حالت کې حلقه نه اجرا کیږي.

ii .که په یو پروګرام کې دوه یا زیاتې حلقې موجودې وي، د هرې حلقې متحولونه باید بېلابېل نومونه ولري.

iv . د حلقې د اجرا شمېر د لاندې فورمول له مخې محاسبه کیږي:

$$n = \text{int}\left[\frac{Upper\ limit - lower\ limit}{stepsize}\right] + 1$$

د مثال په توګه:

```
for example in the loop
for ( i = 0 ; i < 10 ; i = i + 2 )
    {
            .....
    }
```

$$n = \text{int}\left[\frac{9-0}{2}\right] + 1 = 5$$

Output

Enter a number : 6

This is a perfect number : 6

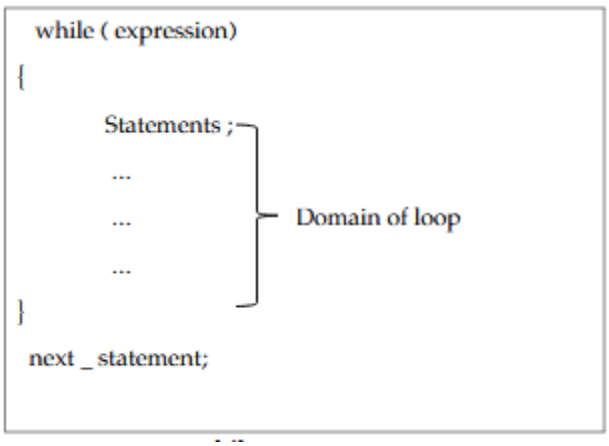
Enter a number : 5

This is not a perfect number : 5

۱.۲۵.۲ بیانیه THE WHILE ( )

ساختار حلقه while په پیل کي شرط ارزیابي کوي او د Pre- testحلقو له ډلي څخه شمېرل کېږي.

عمومي شکل یی په لاندي ډول ده:



شکل 5.6 حلقهwhile

بيانيه ( ) while په لاندې ډول عمل کوي:

❖ په لومړي قدم کې شرط ارزول کېږي، که شرط سم وي نو د حلقې بدنه اجرا کېږي او بيا شرط بيا ارزول کېږي.

❖ که شرط بيا هم سم وي، د حلقې بدنه په تکراري ډول اجرا کېږي.

❖ که شرط سم نه وي، نو د حلقې بدنه نه اجرا کېږي او د حلقې څخه بهر بيانيې (next statement)اجرا کېږي.

❖ حلقه کېدای شي يو يا څو بيانيې ولري .که يوازې يوه بيانيه وي نو د قوسونو (Braces) اړتيا نشته، خو که له يوې څخه زياتې وي نو بايد ټولې بيانيې د لويو قوسونو { } دننه وليکل شي.

❖ تر هغه وخته چې شرط سم وي، د حلقې د بدن ټولې بيانيې په تکراري ډول اجرا کېږي.

مثال 5.6: لاندې پروګرام د ( ) while په کارولو سره طبيعي عددونه له 1 څخه تر n پورې په سکرين ښيي

```
// Program to generate first N natural number
#include <iostream.h>
main ( ) {
int i, n ;
// initialize i to 1
i = 1;
cout << " \n Enter the value of n : " ;
cin >> n;
while ( i <= n )
{
cout << "\n " << i ;
i++;
}
return 0;
}
```

```
Output

Enter the value of n: 5

 1

 2

 3

 4

 5
```

په دې مثال کې لومړنی ارزښت i = 1 دی او حلقه 5 ځله اجرا کیږي .هر ځل چې حلقه اجرا کیږي، د i ارزښت د 1 واحد په اندازه زیاتیږي .کله چې د i ارزښت 6 شي، شرط ناسم کیږي او حلقه پای ته رسیږي.

مثال 6.6: لاندې پروګرام د لومړیو n طبیعي عددونو مجموعه محاسبه کوي او په سکرین یې ښیي

```
// program to find the sum of first n natural numbers 1 + 2 + 3 + ..... + n

#include <iostream.h>

main ( ) {

int sum = 0, i = 1, n;

cout << " \n Enter the value of n : " ;

cin >> n;

while ( i <= n )

{

sum = sum + i;

i ++ ;

}

cout << "\n Sum = " << sum;

return 0;

}
```

```
Output
Enter the value of n: 5
Sum = 15
Enter the value of n : 0
Sum = 0
```

مثال 7.6: لاندې پروګرام د Euclid د الګوریتم په کارولو سره د دوو مثبتو عددونو تر ټولو لوی مشترک مقسوم (GCD)او LCM محاسبه کوي

```
* program to calculate the LCM and GCD of two positive integers using Euclid's
algorithm/*
#include <iostream.h>
main) (
{
int m, n, temp, remain, lcm, gcd;
cout << " \n Enter two integer m & n; " :
cin >> m >> n;
// Assign m * n to temp to retain the original value of m and n

temp = m * n;
remain = m % n;
while ( remain != 0(
{
m = n;
n = remain;
remain = m % n;
}
gcd = n;
lcm = temp/ gcd;
cout << "\n LCD = " <<lcm << " GCD = " << gcd;
return 0;
}
```

Output

Enter two integers m & n : 8 13

LCD = 104, GCD = 1

Enter two integers m & n : 12 8

LCD = 24, GCD = 4

مثال 8.6: لاندې پروګرام د دوو عددونو ترمنځ تر ټولو کوچنی مشترک مضرب (LCM) او تر ټولو لوی مشترک مقسوم (GCD) ښيي

```
/* program to calculate the LCM and GCD of two positive integers */
#include <iostream.h>
main ( )
{
int m, n, temp, lcm, gcd;
cout << " \n Enter two integer m & n : " ;
cin >> m >> n;
// Assign m * n to tem to retain the original value of m and n
temp = m * n ;
// Calculation of gcd
while ( m != n )
148
{
if ( m > n)
m = m – n ;
else
n = n – m ;
}
gcd = n ;
lcm = temp / gcd ;
cout << "\n LCM = " <<lcm << " GCD = " << gcd ;
}
```

Output

Enter two integers m & n : 12 16

LCD = 48, GCD = 4

مثال 9.6: لاندې پروګرام داخل شوی عدد برعکس (reverse) کوي

```
/* program to reserve an integer number */
#include <iostream.h>
main ( )
{
        int num, rev, remain;
        cout << " \n Enter a number : " ;
        cin >> num;
        // initialize rev to 0
        rev = 0;
        while ( num != 0 )
{
        remain = num % 10;
        rev = rev * 10 + remain ;
        num = num / 10 ;
        return 0;
}
```

Dry Run

rev = 0, num = 678

remain	rev	num
8	8	67
7	87	6
6	**876**	0

Output

```
Enter a number : 590
Reversed number is : 95
Enter a number : 678
Reversed number is : 876
```

توليدوي عددونه تسلسل Fibonacci د پروګرام لاندې 10.6: مثال

```
// program to generate the first n Fiabanocci Number
#include <iostream.h>
main (
{
int i = 3, fib1 = 1, fib2 =1, fib, n;
cout << " \n How many Terms ? " ;
cin >> n;
cout<< fib1 << "\n" << fib2 <<"\n" ;
while ( i < n) {
fib = fib1 + fib2 ;
cout << fib <<"\n";
fib1 = fib2 ;
fib2 = fib ;
i++; }
return 0; }
```

Output

```
How many Term ? 5
1
1
2
3
```

مثال 11.6: لاندې پروګرام د داخل شوي عدد د جفت (even) او طاق (odd) ارقامو شمېر محاسبه کوي

```
/* program to count the number of even and odd digits in a number */
#include <iostream.h>
main ( )
{
int num, r, odd, even ;
cout << " \n Enter an integer number : " ;
cin >> num;
odd = 0, even = 0 ;
while ( num != 0)
{
r = num % 10;
num = num / 10 ;
if (r % 2 == 0)
even ++;
else
odd++;
}
cout <<"\n Number od Odd digits : " << odd ;
cout <<"\n Number of Even digits : " << even;
return 0 ;
}
```

Output

```
Enter an integer number : 1213
Number of Odd digits : 3
Number of Even digits : 1
```

۱.۲۵.۳ بیانیه THE DO - WHILE ( )

بیانیه do- while وروسته له دې چې د حلقې بدنه اجرا شي، شرط ارزیابي کوي او د Post-test حلقو له ډلې څخه شمېرل کېږي.

عمومي شکل يې په لاندې ډول دی:

```
do
{
        Statements ;
            ...
            ...
            ...
}
while ( condition );
next _statement;
```

Body of Loop

شکل 6.6 حلقه do- - while

- ❖ په جوړښت while–do کې د حلقې بدنه لومړی اجرا کېږي.
- ❖شرط په دوهم پړاو کې ارزول کېږي .که شرط سم وي، نو د حلقې بدنه بیا په تکراري ډول اجرا کېږي.
- ❖که شرط سم نه وي، نو د حلقې بدنه نور نه اجرا کېږي او د حلقې څخه بهر بیانیې اجرا کېږي.

مثال 12.6: لاندې پروګرام د داخل شوي ارزښت اعتبار (validation) کوي

```
// program to validate your input
#include <iostream.h>
main ( )
{
int num;
do
{
152
cout << " \n Input a positive number : " ;
cin >> num;
}
while ( num <= 0);
return 0;
}
```

Output

```
Input a positive number : -10
Input a positive number : 0
Input a positive number : 5
```

برنامه تر هغه وخته تکرار کيږي چې داخل شوی عدد 0 یا منفي وي .هر کله چې مثبت عدد داخل شي، حلقه ختمیږي او برنامه نور قیمت نه اخلي.

مثال 13.6: لاندې پروګرام د داخل شوي عدد د ضرب جدول (Multiplication Table) په سکرین ښيي

```
// program to print multiplication table of a given number
#include <iostream.h>
main ( )
{
int i, n, prod;
cout <<"\n Enter a number : ";
cin >> n ;
i = 1;
cout <<"\n Multiplication Table \n\n";
do
{
prod = i * n ;
cout <<"\n " <<i <<" * " << n << " = " << prod;
i = i++;
}
while ( i <= 10);
return 0;

}
```

Output

Enter a number : 4

Multiplication Table

1 * 4 = 4

2 * 4 = 8

3 * 4 = 12

4 * 4 = 16

5 * 4 = 20

6 * 4 = 24

7 * 4 = 28

8 * 4 = 32

9 * 4 = 36

10 * 4 = 40

مثال 14.6: لاندې پروګرام د داخل شوي عدد فکتوریل محاسبه کوي

```
// program to find the factorial of a number
#include <iostream.h>
main ( )
{
int i = 1, n;
long fact = 1 ;
cout <<"\n Input a number please :";
cin >> n;
do
{
fact = fact * i ;
i ++;
}
while ( i <= n );
cout <<"\n Factorial of " << n << " = " << fact ;
```

```
return 0;
}
```

Output

```
Input a number please : 6
Factorial of 6 = 720
Input a number please : 1
Factorial of 6 = 1
Input a number please : 0
Factorial of 6 = 1
```

مثال 15.6: لاندې پروګرام د داخل شوي عدد د ارقامو مجموعه (sum of digits) محاسبه کوي

```
// program to find the sum of digits of a number
#include <iostream.h>
main ( )
{
int num, sum = 0, rem ;
cout <<"\n Enter a positive number : ";
cin >> num;
do
{
rem = num % 10 ;
sum = sum + rem ;
num = num / 10 ;
}
while ( num > 0 );
cout <<"\n Sum of the digits = " << sum ;
return 0;
}
```

Output

Enter a positive number : 246

Sum of the digits = 12

Enter a positive number: 0

Sum of the digits = 0

مثال 16.6: لاندې پروګرام له 1 څخه تر n پورې هغه عددونه ښيي چې په 2 او 3 دواړو بشپړ تقسيم

```
// program to print all the numbers divisable by 2 and 3 between 1 and n
#include <iostream.h>
main ( )
{
int num= 1, n, rem2, rem3;
cout <<"\n Enter value of n : ";
cin >> n;
do
{
rem2 = num % 2 ;
rem3 = num % 3;
if (rem2 == 0 && rem3 == 0)
cout <<num << "\n";
num ++ ;
}
while ( num <= n );
return 0;
}
```

Output

Enter value of n : 20

6

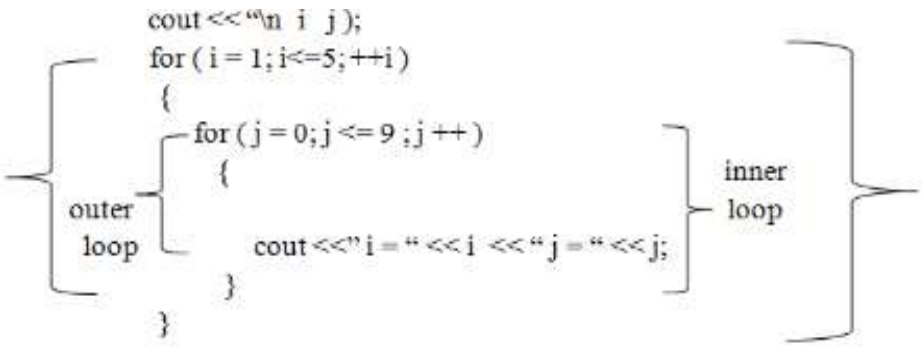
12

18

حلقه په حلقه(Nested FOR Loops)

کله چې يو for حلقه د بل for حلقې دننه ځای پر ځای شي، دې حالت ته Nested FOR Loopويل کيږي .په دې حالت کې داخلي حلقه د بهرني حلقې د هر index لپاره اجرا کيږي.

شکل 7.6 د Nested FOR Loop طرز کار ښيي:



شکل 7.6 Nested for loop

❖په لومړي قدم کې i = 1 ټاکل کيږي او شرط i < = 5 ارزول کيږي .که شرط سم وي نو حلقه اجرا کيږي.

❖په دوهم قدم کې j = 0 ټاکل کيږي او شرط j < = 9 ارزول کيږي .که شرط سم وي نو داخلي حلقه اجرا کيږي او نتيجه په سکرين ښيي:

0 1

❖داخلي حلقه تر هغه وخته تکرارېږي تر څو چې شرط سم وي .کله چې j = 10 شي، نو د بهرني حلقې i ته 1 اضافه کيږي او i = 2 کيږي.

I	j
1	0
1	1
1	2
:	
1	9
2	0
2	1
:	
5	9

تاسو کولای شئ تر 15 مرحلو پوري حلقې (nested loops) د یو بل دننه وکاروئ.

مثال 17.6: لاندې پروګرام د 2 څخه تر 10 پورې د ضرب جدول د nested loop په کارولو سره ښيي

```
#include <iostream.h>
main()
{
for (int a = 2; a<=10; a++)
{
for (int b=1; b<=10; b++)
{
cout <<b <<" * " << a <<" = " <<b *a <<"\n";
if (b>=10)
cout <<"\n";
} }
return 0;
}
```

Output

```
1 * 2 = 2
2 * 2 = 4
....
1 * 3 = 3
2 * 3 = 6
....
1 * 4 = 4
2 * 4 = 8
....
....
1 * 10 = 10
2 * 10 = 20
3 * 10 = 30
4 * 10 = 40
```

```
5 * 10 = 50
6 * 10 = 60
7 * 10 = 70
8 * 10 = 80
9 * 10 = 90
10 * 10 = 100
```

مثال 18.6: لاندې پروګرام ډیجیټل ساعت (Digital Clock) جوړوي

```
#include <iostream.h>
main()
{
int H, M, S;
for (H =1; H<=12; H++)
{
for (M =0; M<=59; M++)
{
for (S = 0; S<=59; S++)
{
for (int MS = 0; MS <2000 ; MS++) // used to slow down second speed
cout <<H <<" : " << M <<" : " << S <<"\n";
}}}
return 0;
}
```

Output

```
1 : 0 : 0
1 : 0 : 1
1 : 0 : 2
1 : 0 : 3
1 : 0 : 4
```

:

:

:

12 : 59 : 59

مثال 19.6 الف :لاندې پروګرام د عددونو په مرسته یو هندسي شکل جوړوي

```
// Program to generate a given pattern with numbers
#include <iostream.h>
main ( )
{
int r, c ; // r is row and c is column
cout << " \n \n Number Pattern :\n \n \n " ;
for ( r = 1; r <= 4; r ++ ) //Outer loop for rows
{
for ( c = 1; c <= r; c++ ) // Inner loop for column
cout << c << " " ;
cout <<"\n " ;
}
return 0;
}
```

Output

```
Number Pattern
1
1 2
1 2 3
1 2 3 4
```

مثال 19.6 ج :لاندې پروګرام د عددونو په مرسته بل هندسي شکل جوړوي

```
// Program to generate a given pattern
160
#include <iostream.h>
main ( )
{
int r, c, num = 6 ; // r is row and c is column
cout << "\n \n \n Number Pattern : \n \n \n " ;
for ( r =num ; r >= 1; r -- ) //Outer loop for rows
{
for ( c = 1; c <= r; c++ ) // Inner loop for column
cout << r <<" ";
cout <<"\n " ;
}
return 0;
}
```

Output

```
Number Pattern
6 6 6 6 6 6
5 5 5 5 5
4 4 4 4
3 3 3
2 2
1
```

مثال 19.6 ج :لاندې پروګرام د عددونو په مرسته بل هندسي شکل جوړوي

```
// Program to generate a given pattern with numbers
#include <iostream.h>
main ( )
{
int r, c1, c2, n, m, p ; // r is row and c1, c2 are columns
cout << " \n Enter the number of rows : " ;
cin >> n;
cout <<"\n \n Number Pattern \n \n ";
161
p = 1;
for ( r =1 ; r <= n ; r++ ) //Outer loop for rows
{
for ( c1 = 1; c1<2*n-r; c1++ ) // Inner loop for column
cout << " ";
for ( c2 = 1; c2 <=r; c2++, p++)
cout <<p << " ";
for ( c2 = r-1; c2 >= 1; c2--, p--)
{
m = p-2;
cout <<m << " ";
}
cout <<"\n" ;
}
// return 0; }
```

Output

Enter the number of rows : 5

Number Pattern

1

2 3 2

3 4 5 4 3

4 5 6 7 4 5 4

5 6 7 8 9 8 7 6 5

مثال 20.6 د :لاندي پروګرام په بازه کي د (prime numbers) اول نمبرونه پيدا کوي، د شرط له مخي چې m کوچنی له n څخه وي او دواړه int وي

```
// Program to find prime numbers in the range m to n where m and n are positive
integer and m < n */
#include <iostream.h>
main ( )
{
int m, n, i, j, isprime ;
```

```
        cout << " \n  Enter Lower and Upper limit of the range  m  &  n  : " ;
        cin >> m >> n;
        if ( m<= 1)
        m = 2 ;                  // starting value of the range is made 2
        cout << "\n The prime numbers are : \n \n " ;
        for ( i =m ; i <= n ; i ++ )
           {
             isprime = 1;
             // This loop checks  I for a prime number
             for (  j = 2; j<= i/2 ; j++ )
               {
                  if ( i % j == 0 )
                   {
                    isprime =0 ;
                    break;
                    }                    // End of  j  loop
               }
           if ( isprime )
           cout <<"\n " << i ;
          }                              // End of i loop
           return 0;
}
```

Output

```
Enter Lower and Upper limit of the range m  &   n :   5   25
The prime numbers are
                5
                7
                11
                13
                17
                19
                23
```

پرش در حلقه‌ها(Jump in Loops)

کله داسي حالت ته ورسيږي چې اړتيا وي(Stroustrup, 1997) :

a) د حلقې اجرا بند شي.

b) حلقه په بشپړه توګه ختمه شي.

c) د حلقې يو برخه نظرانداز شي.

d) د داخلي حلقې څخه بهر ووتل شي او بېرته بيروني حلقې ته لاړ شو د شرطونو له مخې .

داسي مختلفې لارې شته چې د فور عمليات پرې ترسره کېدای شي.

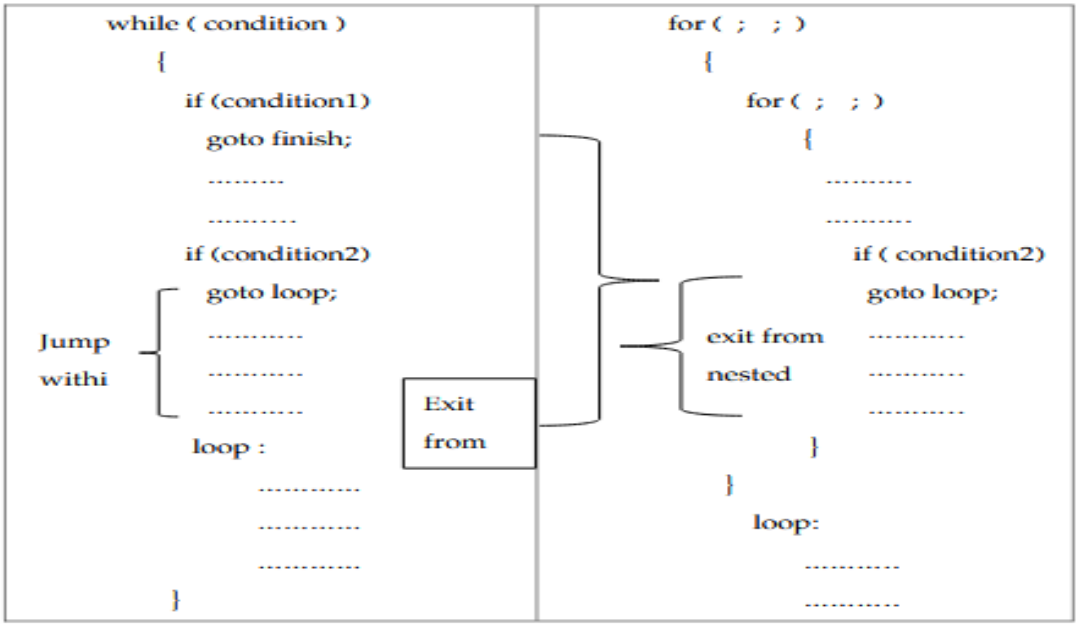
تاسو کولای شئ د لاندې دستورونو په وسيله د فور عمليات ترسره کړئ:

goto، break ، continue اوexit

I . بيانيهgoto

د goto بيانيه کولای شئ دwhile ، do-while او for حلقو دننه وکاروئ، تر څو د پروګرام د يوې برخې اجرا ودروئ او له بل ځای څخه يې بيا پيل کړئ .همدارنګه دا د nested loops څخه د وتلو لپاره هم کارول کېږي.(Bailey, 2007)

لاندې شکل د goto په وسيله د حلقې څخه د وتلو طرز ښيي:



شکل 8.6 د حلقو (loops) دننه د goto استعمال

ii . بیانیهbreak

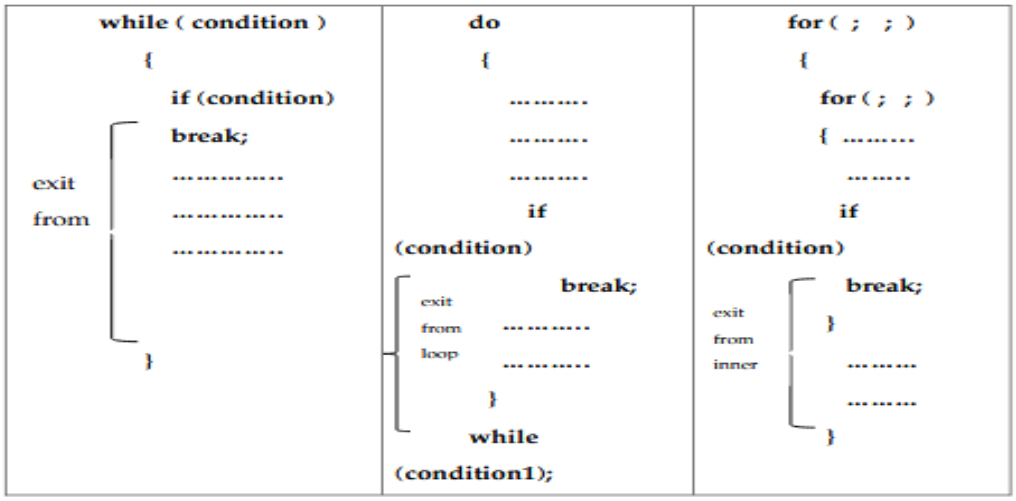
دا بیانیه نه یوازې په switch کې کارول کېږي، بلکې پهwhile ، do-while او for حلقو کې هم کارول کېدای شي.

breakد حلقې یا switch څخه د وتلو لپاره کارول کېږي .باید وویل شي چې break یوازې هغه حلقه ختموي چې پکې موجود وي، او نشي کولای له nested loops څخه په یو ځل بهر شي (Langtangen, 2006).

د break عمومي شکل:

```
break ;
```

لاندې څو مثالونه د break بیانیه ښیي:



شکل 9.6 د break استعمال په حلقو کې

iii. بیانیهcontinue

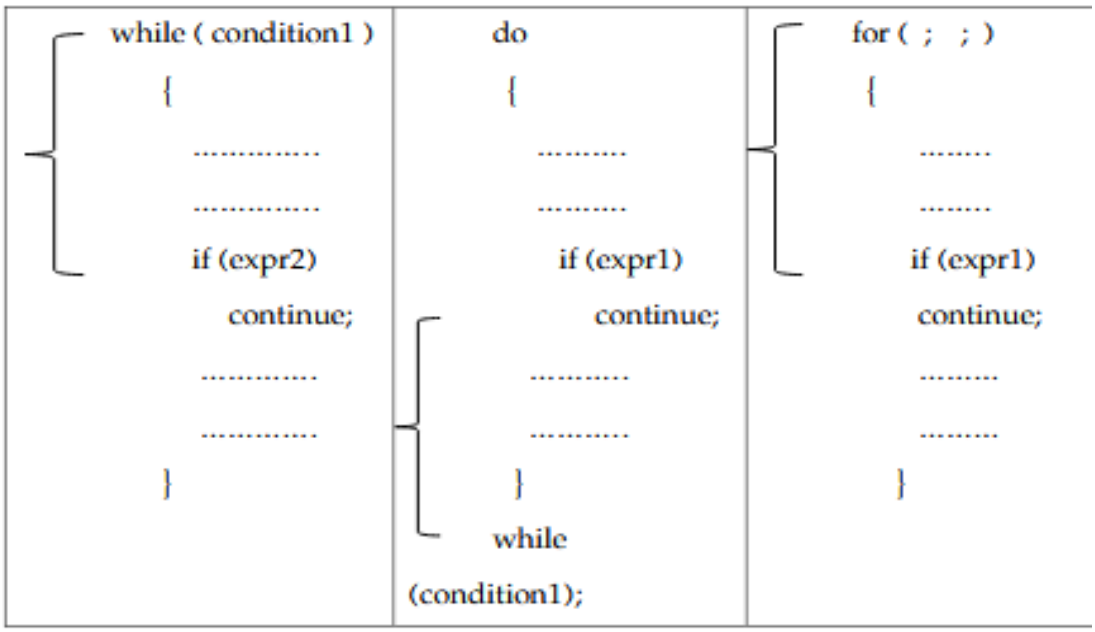
بیانیه continue د حلقي اجرا ته دوام ورکوي او د حلقي اجرا نه بندوي.

شکل عمومی continue یی په لاندي ډول ده :

```
continue ;
```

تاسو کولای شئ دا دستور پهwhile ، do-while او for حلقو کي وکاروئ .بیانیه continue باید د حلقي د بدن (body) دننه تعریف شي، ځکه له حلقي بهر هیڅ ګټه نه لري.

طرز استفاده continue په مختلفو حلقو کي:



شکل 10.6 د continue استعمال په حلقو کي

تابع exit ( )

دا تابع په فوري ډول د پروګرام اجرا ختموي .عمومي شکل يي قرار ذيل دی:(Eckel, 2000)

```
exit ( n ) ;
```

❖دا تابع ټول خلاص فایلونه بندوي او یو عدد n د هغه تابع ته ورګرځوي چې دا یې بللې وي (caller).

❖د پروګرام د حالت (exit status) نمایندګي کوي .که n = 0 وي، معنا یې دا ده چې پروګرام بې له غلطیو (no errors) اجرا شوی .او که n ≠ 0 وي، نو دا ښیي چې په پروګرام کې خطا (errors)موجوده ده.

❖د n ارزښت د خطاوو د تشخیص لپاره هم کارول کېدای شي.

❖د default یا پیش‌فرض قیمت n عموماً 0 وي، له همدې امله د exit تابع عمومي شکل داسې دی :

exit ( ) ;

مقایسه حلقه (while) و(do–while)

جدول 1.6 مقایسه حلقه while وdo–while

do – while ( )	while ( )
1) شرط د حلقې په پای کې ارزول کېږي. دې ډول حلقې ته Exit- controlled loop او post-test هم وایي.	1) شرط د حلقې په پیل کې ارزول کېږي. دې ډول حلقې ته Entry- controlled loop او pre-test هم وایي.
2) تر هغه وخته چې شرط سم وي، while- حلقه اجرا کېږي.	2) حلقه while تر هغه وخته اجرا کېږي چې شرط سم وي.
3) حلقه do-while لږ تر لږه یو خل اجرا کېږي، که شرط ناسم هم وي.	3) حلقه while هیڅ اجرا نه کېږي که شرط ناسم وي.

اووم فصل

په دې فصل کې د صف (Array) مفهوم، د صف ډولونه او د هغې د استعمال موارد تر بحث لاندې نیول شوي دي.

ځینې پروګرامونه اړتیا لري چې څو ببلابېل معلومات چې یو ډول (Data type) ولري لکه x1 ، x2، xn ... x3 په خپل داخل کې ذخیره کړي .په داسې حالت کې د صف (Array) استعمال غوره وي.

صف (Array) یو ځانګړی ډول ډیټا ټایپ دی چې هغه عناصر په کې ذخیره کیږي چې یو شان نوعیت (data type) ولري.

د صف مهم خصوصیات په لاندې ډول دي:(Oualline, 1995)

1. صف کولای شي حرفي، عددي (integer) او اعشاري (float) ارزښتونه ذخیره کړي.
2. د صف ټول عناصر باید یو شان data type ولري.
3. د صف هر انفرادي عنصر ته Variable Subscripted ویل کیږي.
4. هر عنصر یو index لري.
5. د صف اندازه (range) د [ ] square brackets دننه ټاکل کیږي.
6. د صف index تل له 0 څخه شروع کیږي.

مثال:

```
int marks [50] ;
```

په دې مثال کې marks د صف نوم دی چې 50 پرلهپسې ځایونه په حافظه کې ریزرف کوي تر څو 50عددي ارزښتونه په کې ذخیره شي.

د صف عناصر په لاندې ډول لیکل کیږي:

```
marks[0], marks[1], marks[3] ... marks[49]
```

7. د صف عناصر په حافظه کې په مسلسل ډول ذخیره کیږي.

8. د subscript شمېر د صف بعد (dimension) ټاکي.

Types of Array صف ډولونه (Chapter 23)

په عمومي ډول دري ډوله صفونه شتون لري:(Jones & Aitken, 2014)

یو بعدي صف(One Dimensional Array)

دوه بعدي صف(Two Dimensional Array)

څو بعدي صف(Multi Dimensional Array)

صف یو بعدي(One Dimensional Array)

هغه صف چې عناصر یې یوازې یو index (subscript) ولري، هغه ته یو بعدي صف ویل کیږي.

مثال :لاندې یو بعدي صف (One Dimensional Array) په پام کې ونیسئ:

```
int marks [5] ;
```

په دې مثال کې ښکاري چې د مربع قوسونو [ ] په داخل کې یو عدد شتون لري او دا ښیي چې دا صف یو بعدي صف دی.

صف د marks په نوم لاندې د 5 محصلینو نمرې ذخیره کوي:

Marks [0]	Marks [1]	Marks [2]	Marks [3]	Marks [4]
40	38	12	44	35

د دې صف ارزښتونه په لاندې ډول هم ليکل کېدای شي:

```
Marks [ 0 ] = 40;
Marks [ 1 ] = 38;
Marks [ 2 ] = 12;
Marks [ 3 ] = 44;
Marks [ 4 ] = 35;
```

۱.۲۷د صف تعريف(Definition of Array)

مخکې له دې چې صف په پروګرام کې وکارول شي، بايد اول تعريف شي .صف د نوعيت(type) ، نوم (name) او اندازه (size) په وسيله تعريفېږي.(Rama, 2011)

د صف عمومي شکل:

```
data type array_name [ size ];
```

- Data type :د متحول ډول ټاکي لکه int، float ، char يا double
- Array name : د صف نوم دی او د identifier قواعد تعقيبوي
- Size : د صف د عناصرو اعظمي شمېر ټاکي

مثال1.7

```
int x [ 100 ] ;
char text [ 80 ] ;
float sal [ 50 ] ;
```

په دې مثال کې کولای شي x تر 100 پورې عددي (integer) ارزښتونه ذخیره کړي، text تر 80پورې حروف ذخیره کوي، او sal تر 50 پورې اعشاري شمېرې ذخیره کوي.

متوجه اوسئ چې:

```
int max_stud = 50 ;
int marks [ max _ stud ] ; // not allowed
```

په پورته مثال کې دوهمه جمله سمه نه ده، ځکه د صف اندازه باید تل ثابت عدد (constant integer)وي. تاسو کولای شئ max_stud د ثابت په توګه داسې تعریف کړئ:

```
#define MAX_STUD 50
int marks [MAX_STUD];
```

مثال ۲,۷دا پروګرام له کیبورډ څخه 3 صحیح عددونه اخلي او د هغوی مجموعه په سکرین ښیي

```
/* Program to insert 3 int values from keyboard and displays sum of them */
#include <iostream.h>
main()
{
int Array [3];
cout <<"Enter 3 Integer values..?";
cin >> Array[0];
cin >> Array[1];
cin >> Array[2];
cout <<"Total = "<< Array [0] + Array [1] + Array [2];
return 0;
}
```

Output

Enter 3 Integer values..?

5

10

20

Total = 35

مثال ۳,۷دا پروګرام د انګلیسي حروفو صف ښیي.

```
#include <iostream.h>
main ( )
{
char Array [ ]= {'a','b','c','d','e'};
cout <<Array[0] <<"\n";
cout <<Array[1] <<"\n";
cout <<Array[2] <<"\n";
cout <<Array[3] <<"\n";
cout <<Array[4]<<"\n";
return 0;
}
```

Output

a

b

c

d

e

مثال 4.7 دا پروګرام ډاټا په صف (array) کې داخلوي او بيا يې په سکرين ښکاره کوي.

```
/* Program to write the data into an array and print them */
#include <iostream.h>
main ( ) {
int a [ 10 ] , i ;
/* Write the elements into the array */
cout << " Enter five elements " ;
for ( i = 0; i <5 ; i++)
cin >> a [i];
/* Output the data items of array a */
cout << " \n Elements of the array are : \n " ;
for ( i = 0; i < 5 ; i++)
cout << "\n " << a [ i ] ;
return 0; }
```

Output

```
Enter five elements
2
10
25
15
72
Elements of the array are :
2
10
25
15
72
```

مثال لاندي پروګرام د داخل شوو ۵ عددونو ترمنځ مجموعه ،اوسط ،تر ټولو لوی کوچنی ،عدد ترلاسه کوی.

```
/* Program to find the Sum, Average, Largest and Smallest of n number */
#include <iostream.h>
main ( )
{
int a [ 10 ] , i, large, small, n ;
float avg, sum ;
175
/* Input elements into the array */
cout << " \n How many elements (n < 10) ? " ;
cin >> n ;
cout << " Enter the elements: "<< n;
for ( i = 0; i < n ; i++)
cin >> a [i];
/* Finding sum, largest and smallest */
large = a [ 0 ] ; small = a [ 0 ] ; sum = 0 ;
for ( i = 0; i < n ; i++)
{
sum = sum + a [ i ] ;
if ( a [ i ] < small )
small = a [ i ] ;
}
/* Finding average value */
avg = sum / n ;
/* output result */
cout << "\n sum = " << sum <<" Average = " <<avg ;
cout << "\n largest = " << large <<" Smallest = " << small ;
return 0;
}
```

Output

How many elements (n < 10) ? 5

Enter the 5 elements :

67

25

40

33

6

Sum = 171.00 Average = 34. 20

largest = 67 smallest = 6

۱.۲۸د صف (Array) د يو بعدي صف قيمت ورکول (Initializing of one Dimensional Array)

د صف عناصر کولای شو د صف د تعريف پر وخت هم مقدار ورکړو:(Nawaz, 2006) .

عمومي بڼه:

```
static type array_name [ size ] = { list of values } ;
```

تشريح:

→ staticد ذخيره کولو ډول

→ typeد معلوماتو ډول(int, float, char) او نور

→ array_nameد صف نوم

→ sizeد صف اندازه (څو عناصر لري)

list of values → د صف د ارزښتونو لیست چې د کامې (,) په واسطه بېل شوي وي

(Nakov & Kolev, 2013)

مثال ۶.۷

The statement static int primes [5] = { 2 , 3, 5, 7, 11 };

Will assign the values as follows:

primes [0] = 2, primes [1] = 3, primes [2] = 5, primes [3] = 7, and primes [4] = 11

Similarly char color [ ] = "RED" ;

Will assign color [0] = "R", color [1] = 'E', color [2] = 'D' and color [3] = '\0'

'R'	'E'	'D'	'\0'

static int num [10] = { 1, 2, 3, 4} ;

will initialize the first 4 array elements to 1, 2, 3, 4 and the remaining 6 must be zero.

که قیمتونه د for ترتیب مطابق داخل شي نو په لاندی ډول به وښودل شي.

مثال۷,۷:که د ۵ محصلانو نمری په ترتیب سره ۶۶،۷۰،۴۵،۸۹او ۸۱ وي نوتاسو کولای شي. تاسو کولای شي د forحلقي په کارولو سره قیمتونه د صف عناصرو ته په لاندی ډول ورکړي :

consider the marks of 5 students to be 89, 45, 70, 66, and 81. A for ( ) loop can be used to assign these values to the array elements as follow:

```
int marks [ 5 ] , i ;
cout <<" \ n Enter the marks: " ;
for ( i = 0; i<5; i++ )
{
cin >> marks [ i ] ;
```

هغه ډاټا چې د لوپ (loop) په مرسته داخل شي، هماغسې په همدې بڼه بڼودل کېږي:

89	45	70	66	81

ډاټا بايد تل د حلقې (loop) په شکل صف ته داخل شي ،ځکه چې د هر cinلپاره دزير نويس i(subscript) ارزښت بدليږي .په لومړي ځل cinکی marks(0)لوستل کېږي ورسته marks(1)او همداسي دنورو قيمتونو لپاره ادامه مومي.

۱.۲۹د يو بعدي صف (Processing of one Dimensional Array)

تاسو نشئ کولی يو حسابي عمليه په ټول صف باندې يوځای اجرا کړئ .د مثال په ډول، که غواړئ دوه صفونه a او b چې يو شان اندازه لري سره جمع کړئ، نو دا عمليه بايد د هر عنصر لپاره جلا ترسره شي.(2013, Overland)

مثال:

```
c[0] = a[0] + b[0]
c[1] = a[1] + b[1]
```

له همدې املې، هغه دوه صفونه چې يو شانdata type ، يو شان بُعد، يو شان اندازه، يو شان عمليه او يو شان مقايسوي عملګر ولري، اړتيا لري چې د هر عنصر لپاره جلا پروسس شي .د دې لپاره د forحلقه کارول کېږي، ځکه په هر ځل اجرا کې يو عنصر پروسس کوي .د حلقې د اجرا شمېر او د پروسس کېدونکو عناصرو شمېر سره برابر وي.

مثال 8.7: لاندې پروګرام ۳ عددونه له کیبورد څخه اخلي او بیا یې ښیي.

```
#include <iostream.h>
main ( )
{
int Array [3];
cout <<"Enter 3 Integer values..?"; // Input from KEYBOARD Scanner
cin>> Array[0];
cin>> Array[1];
cin>> Array[2];
cout << Array [0] << "\n" << Array [1] << "\n" << Array [2];
178
return 0;
```

Output

```
Enter 3 Integer values..? 1 2 3
1
2
3
```

مثال 9.7 دا پروګرام د دوو یو بعدي صفونو (1-D array) عناصر سره جمع کوي:

```
/* Programs to add Two 1-D array */
#include <iostream.h>
main ( )
{
int a [ 10 ], b [ 10 ], c [ 10 ] ;
int i ;
cout << " \n Enter 10 elements of array a : \n " ;
for ( i = 0; i < 10 ; i++)
cin >> a [i];
cout << " \n Enter 10 elements of array b : \n " ;
for ( i = 0; i < 10 ; i++)
cin >> b [i];
/* Finding sum of array a and b */
for ( i = 0; i < 10 ; i++
c [ i] = a [ i ] + b [ i ] ;
// Printing the resultant array c
cout <<"\n Resultant array:\n ";
for ( i = 0; i < 10 ; i++)
cout <<"\t" << c [ i ] ;
return 0;
}
```

Dry Run

Finding sum of array a & b

i	a [i]	b [i]	c [i]
0	a [0] =1	b [0] =2	c [0] =3
1	a [1] =2	b [1] =3	c [1] =5
2	a [2] =3	b [2] =5	c [2] =8
3	a [3] =4	b [3] =7	c [3] =11
:			
:			
9	a [9] =10	b [9] =31	c [9]=41

Output

Enter 10 elements of array A

1 2 3 4 5 6 7 8 9 10

Enter 10 elements of array B

2 3 5 7 11 13 19 23 29 31

Resultant array :

3 5 8 11 16 19 26 31 38 41

مثال 10.7دا پروګرام د Fibonacci سلسله د n عناصرو لپاره په صف کې جوړوي:

```
/* Programs to generate first n Fabonacci terms using an anrray */
#include <iostream.h>
main ( )
{
int fibo [50], i, n;
cout << " \n How many terms of the series ? " ;
cin >>n;
// Initialize the first and second term
fibo [0] = 0; fibo [1] =1;
for ( i = 2; i < n ; i++)
fibo [ i] = fibo [ i-2 ] + fibo [ i-1 ] ;
cout <<"\n The Fibonacci terms are " ;
for ( i = 0; i < n ; i++)
cout << fibo [i] << " ";
return 0;
}
```

Output

How many terms of the series ? 10

The Fibonacci terms are :

0 1 1 2 3 5 8 13 21 34

مثال 11.7دا پروګرام د لسمي (Decimal) عدد په هر بل قاعده (Base) بدلوي، لکه:Hexa Decimal

```
/*  Programs to convert a Decimal number to Hexa Decimal */
#include <iostream.h>
main ( )
{
int num, rem, a[10], base, i=0, k;
cout << " \n Enter a number:  " ;
cin >>num;
cout <<" \n Enter  Base: ";
cin >> base;
cout <<"\n The output:  ";
while (num > 0 )

{
  rem = num % base ;   // converting to base equivalent
  a [i] = rem;
  i++;
  num = num / base ;
}
 i--;                    // Print the converted number
for (k=1; k>=0; k--)
{
   if ( a[k] <=9 )
     cout << a[k];
   else if (a[k] ==10)
    cout << "A";
      else if (a[k] ==11)
        cout << "B";
          else if (a[k] ==12)
            cout << "C";
              else if (a[k] ==13)
                cout << "D";
                  else if (a[k] ==14)
                    cout << "E";
                      else if (a[k] ==15)
                        cout << "F";

                          else
                            {
                              cout << "\n Error ";
                              exit (0);
                            }
  }
  return 0;
```

```
}
```

Output

```
Enter a number  : 105
Enter Base      : 16
The Output      : 69
```

دوه بُعدي صف(Two Dimensional Arrays)

دوه بُعدي صف له سطر (row) او ستون (column) څخه جوړ وي .د دوه بُعدي صف هر عنصر مونږ د دوو زیرنویسونو (subscripts) په وسیله ترلاسه کولی شو، چې لومړی زیرنویس د سطر او دوهم زیرنویس د ستون استازیتوب کوي.(Lippman, 2002)

```
type array-name [ row size ] [ col size ];
```

- دلته row size او col size د سطرونو او ستونونو د اعظمي قیمت شمېر ښیي چې په صف کې ذخیره کېږي.
- type د ډاټا ډول ښیي.
- array-name د صف نوم ساتي.

مثال :7.12 د يو دوه بُعدي صف تعريف چې ۵ سطرونه او ۶ ستونونه لري

د دوه بُعدي صف پېژندنه

```
float mat[5][6];
```

دا په مسلسل ډول 120 بايټه حافظه reserve کوي .په صف کي 6 × 5 = 30 عناصر شته، او هر عنصر د اعشاري عدد په ډول د ذخيره کېدو لپاره 4 بايټه حافظي ته اړتيا لري.

انفرادي عناصر عبارت دي له:

```
mat[0][0]    mat[0][1]   mat[0][2]    mat[0][3]    mat[0][4]     mat[0][5]
mat[1][0]    mat[1][1]   mat[1][2]    mat[1][3]    mat[1][4]     mat[1] [5]
:
:
mat[4][0]    mat[4][1]   mat[4][2]    mat[4][3]    mat[4][4]     mat[4] [5]
```

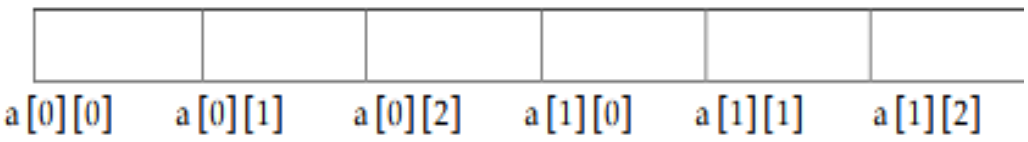
❖د صف قيمت له 0 څخه پيل او تر (size - 1) پورې دوام کوي.

❖د دوه بُعدي صف عناصر د سطر په شکل ذخيره کېږي.

مثال:

```
int a[2][3]
```

د ذخيره کولو طريقه:



د دوه بعدي صف (2D Array) قیمت ورکول (Initializing Two Dimensional Arrays)

په دوه بعدي صف کي ابتدايي ارزښتونه د { } قوسونو په منځ کي ورکول کيږي(Horton, 2012) .م

مثال

```
int mat [3][3] = { 19, 8, 11, 25, 4, 16, 0, 8, 5 };
mat [0] [0] = 19,  mat [0] [1] = 8,  mat [0] [2] = 11
mat [1] [0] = 25,  mat [1] [1] = 4,  mat [1] [2] = 16
mat [2] [0] = 0,    mat [2] [1] = 8,  mat [2] [2] = 5
```

:

تاسو کولای شئ پورتنی مثال په معکوس ډول په لاندي بڼه ولیکئ:

```
int mat [3] [3] =  {
        { 19,  8,  11 },
        { 25,  4,  16 },
        { 0  , 8  ,  5  },
};
```

په یاد ولرئ چي د هري جوړي قوسونو وروسته باید کامه اضافه شي .په عمومي ډول، د دي لپاره چي د دوه بُعدي صف هر عنصر پروسس شي، اړتیا ده چي له Nested loop څخه استفاده وشي.

```
int a [ 5 ] [ 3 ], i ;
  for (i = 0; i <5; i++)
     for (j = 0; j <5; j++)
          a [ i ] [ j ] = 0 ;
```

لومړی حلقه (Loop) د سطر (Row) د قیمت د بدلون لپاره او دوهمه حلقه د ستون (Column) د قیمت د بدلون لپاره کارول کيږي.

تاسو کولای شئ ساده عملیات لکه جمع، تفریق، ضرب او همدارنګه د سطر په ستون یا د ستون په سطر بدلول د دوه بُعدي صف په وسیله اجرا کړئ.

د یادونې وړ ده چې د matrix سازګاري باید په پام کې ونیول شي، یعني په ځینو حالتونو کې باید د سطرونو او ستونونو شمېر سره مساوي وي، که نه نو ځینې عملیات نه اجرا کېږي.

مثال 13.7: لاندې برنامه د A او B مکسونه لوستل کوي او د محاسبې وروسته یې جمع شوي نتیجه په

```
// Programs to read two matrices A and B and display their sum
#include <iostream.h>
main ( )
{
int a [ 10 ] [ 10 ], b [ 10 ] [ 10 ], sum [ 10 ] [ 10 ], i, j, n ;
cout << " \n Enter the order of matrix : " ;
cin >>n;
/* Reading the elements of matrices a and b */
cout <<"\n Enter the elements of matix a : \n";
for (i =0; i <n; i++)
  for ( j=0; j<n; j++)
    cin >> a[ i ][ j ];
cout <<"\n Enter the elements of matix b : \n";
 for (i =0; i <n; i++)
   for ( j=0; j<n; j++)
     cin >> b[ i ][ j ];
// Finding the sum of the two matrices
for (i =0; i <n; i++)
  for ( j=0; j<n; j++)
    sum [ i ] [ j ] = a [ i ] [ j ] + b [ i ] [ j ];
/* Printing the resultant matrix */
cout <<"\n The Summation matrix : \n " ;
for ( i=0 ; i <n ; i ++ )
 {
    for ( j = 0; j <n; j++ )
   {
     cout << sum [ i ][ j ] <<" ";
    }
```

```
out << "\n" ; // generate line after each row

}

return 0;

{
```

Output

```
Enter the order to matix : 3

Enter the elements of matrix A:

1 2 3 4 5 6 7 8 9

Enter the elements of matrix B:

2 4 6 8 10 12 14 16 18

The summation matrix :

3 6 9

12 15 18
```

مثال 14.7: لاندې برنامه یو مکس (Matrix) لوستل کوي او په نتیجه کې یې سطرونه (Rows) ستنو (Columns)ته او ستنې سطرونو ته بدلوي (Transpose) کوي

```
// Program to read a matrix and output its transpose

#include <iostream.h>

main ( )

{

int a [ 10 ] [ 10 ], trans [ 10 ] [ 10 ] ;

int i, j, r, c;

cout << " \n Enter no of rows and no of columns of a matrix : " ;

cin >> r >> c ;

cout <<"\n Enter the matix elements of a : \n";

// Rending and assigning to trans matrix

for (i =0; i <r; i++)

for ( j=0; j<c; j++)

{

cin >> a [ i ] [ j ] ;

trans [ j ] [ i ] = a [ i ][ j ]; // Exchange rows and column positions
```

```
}
cout << "\n The given matix a : \n" ;
for ( i = 0 ; i<r ; i++)
{
for ( j=0; j <c ; j++ )
{
cout << a[ i ][ j ] << " ";
}
cout << "\n" ;
}
cout << "\n The Transposed matrix : \n" ;
for ( i=0; i <c ; i++ )
{
for (j=0; j<r; j++)
{
cout << trans [ i ] [ j ]<< " ";
}
cout<<"\n";
}
```

Output

```
Enter no of rows and no of columns of a matrix : 3 2
Enter the matrix elements of a :
1 3
5 7
9 11
The given matrix A:
1 3
5 7
9 11
The Transposed matrix :
1 5 9
3 7 11
```

مثال 15.7: لاندې برنامه دوه مکسونه لوستل کوي او د هغوی د ضرب نتیجه محاسبه کوي

```
// Program to read two matrices and find their product
#include <iostream.h>
main ( )
{
int a [ 10 ] [ 10 ], b [ 10 ] [ 10 ], prod [ 10 ] [ 10 ], i , j , k, n ;
cout << " \n Enter the order of matrix: " ;
cin >> n;
cout <<"\n Enter the elements of matrix a : \n";
for (i =0; i <n; i++)
for ( j=0; j<n; j++)
cin >> a [ i ][ j ];
cout <<"\n Enter the elements of matrix b : \n";
for (i =0; i <n; i++)
for ( j=0; j<n; j++)
cin >> b [ i ][ j ];
// Compute product of matrix a & b
for ( i = 0 ; i<n ; i++)
for ( j=0; j<n; j++)
{
prod [ i ][ j ] = 0 ;
for (k=0; k <n; k++)
prod [ i ][ j ] = prod [ i ][ j ] + a [ i ][ k ] * b [ k ][ j ];
}
cout << "\n The product of a and b matrix : \n" ;
for ( i=0; i<n; i++)
{
for ( j=0; j<n; j++)
```

```
{
cout << prod [ i ] [ j ] <<" ";
}
cout <<"\n" ;
}
return 0;
}
```

Output

```
Enter the order of matrix : 3
Enter the elements of matrix a :
1 2 3
1 2 3
1 2 3
Enter the elements of matrix b :
2 4 6
2 4 6
2 4 6
The product of a and b matrix :
12 24 36
12 24 36
12 24 36
```

مثال 16.7: لاندي برنامه د مربع مکس د قطر (Diagonal) د عناصرو جمع محاسبه کوي

```
// Program to print the sum of the diagnoals of a squar matrix
#include <iostream.h>
main ( )
{
```

189

```
int mat [ 10 ] [ 10 ], i , j , n ;
int dsum1 = 0, dsum2 = 0; // sum of diagonals
cout << " \n Enter the order of matrix: " ;
cin >> n;
cout <<"\n Enter the elements of matrix a : \n";
for (i =0; i <n; i++)
for ( j=0; j<n; j++)
{
cin >> mat [ i ][ j ] ;
// To calculate sum of left to right diagonal elements
if ( i == j )
dsum1 = dsum1 + mat [ i ] [ j ] ;
// T o calculate sum o right to left diagonal elements
if ( i + j == n-1 )
dsum2 = dsum2 + mat [ i ] [ j ] ;
}
cout <<"\n Sum of left to right diagonal = " <<dsum1;
cout <<"\n Sum of right to left diagonal = " <<dsum2;
return 0;
}
```

Output

```
Enter the order of matrix : 3
Enter the elements of matrix :
1 5 9
2 3 7
6 8 10
Sum of left to right diagonal = 14
Sum of right to left diagonal = 18
```

څو بُعدي صف(Multi-Dimensional Arrays)

کله چې یو صف له ۲ بُعدو څخه زیات بُعدونه ولري، هغه ته څو بُعدي صف(Multi-Dimensional Array)ویل کېږي.

د هغه عمومي شکل په لاندې ډول دی:

```
data-type array-name [S1] [S2] ..... [Sn];
```

دلته S1 ، Sn ... S2 مثبت صحیح عددونه دي چې په صف کې د عناصرو شمېر ښیي، او هر یو یې یو زیرنویس (subscript) لري.

Data-typeد متحول نوعیت ښیي او array-name د صف نوم دی.

له همدې امله:

په دوه بُعدي صف کې ۲ زیرنویسونه (subscripts) وي.

په درې بُعدي صف کې ۳ زیرنویسونه وي.

او په n بُعدي صف کې n زیرنویسونه موجود وي.

مثال : 7 .17د څو بُعدي صف درې بېلګې

```
int lamps [ 3 ] [ 3 ] [ 4 ];
float tri  [ 3 ] [ 3 ] [ 3 ];
int table  [ 4 ] [ 4 ] [ 5] [ 2 ];
```

2] ;

په پورتني مثال کې:

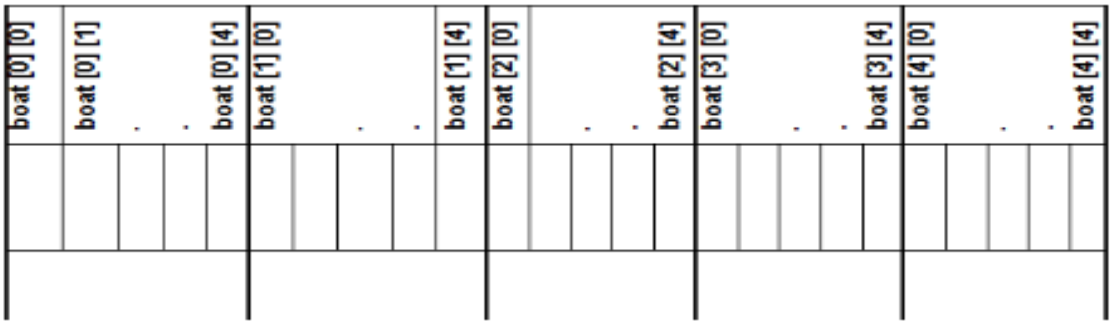
lampsيو ۳ بُعدي صف دی چې 3 × 3 × 4 = 36 صحيح عناصر لري.

triهم يو ۳ بُعدي صف دی چې 27 عناصر لري.

او table يو ۴ بُعدي صف دی چې 160 د int ډول عناصر لري.

مونږ کولای شو دوه بُعدي صف په لاندې ډول وښيو:

```
int boat [ 5 ] [ 5 ] ;
```



لاندې دوه بُعدي صف هم په نظر کې ونيسئ:

```
int std [ 50 ] [ 5 ] ;
```

په دې مثال کې د ۵۰ محصلانو نمرې په ۵ مضمونونو کې ښودل شوې دي.

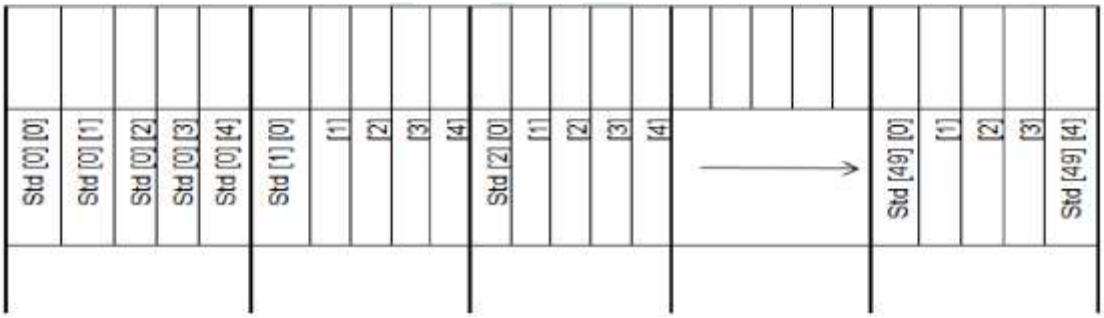
std[0][0] لومړي محصل هغه نمرې ښيي چې په لومړي مضمون کې يې ترلاسه کړې دي.

std[0][1] لومړي محصل د دوهم مضمون نمرې ښيي، او همداسې تر آخره پورې ادامه لري.

std[49][4] هغه نمرې ښيي چې ۵۰م محصل په پنځم مضمون کې ترلاسه کړې دي.

Marks	Sub 0	Sub 1	Sub 2	Sub 3	Sub 4
Students 0					
Students 1					
Students 2					
:					
:					
Students 49					

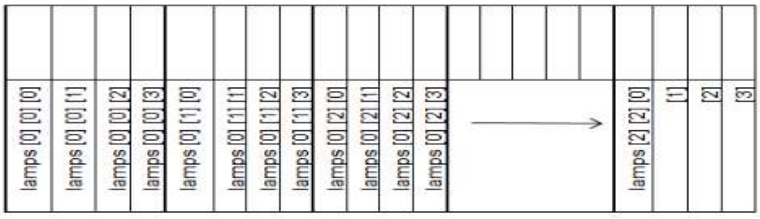
د صف عناصر د کمپيوټر په حافظه کې په لاندې ډول ذخيره کيږي.



همدارنګه ۳ بُعدي صف:

```
int lamps [3] [3] [4];
```

په لاندې ډول ښودل کيږي:



اتم فصل

دا فصل د تابع (Function) اړوند مفاهيم لري لکه :د تابع جوړول، د تابع استعمال، تابع ته د قيمت ليږل، او له تابع څخه د قيمت بيرته ترلاسه کول.

تر اوسه پورې چې کومې برنامې ليکل شوې دي، يوازې يو اصلي تابع (main) لري.
په اوږدو او پيچلو برنامو کې چې څو ببلابېلې او خپلواکې برخې لري، ښه دا ده چې د هرې برخې لپاره جلا برنامه) تابع (وليکل شي .هغه جلا برخه چې د مسئلې د يوې برخې د حل لپاره ليکل کېږي، تابع بلل کېږي.
تابع په اصل کې د پروګرام هغه برخه ده چې د څو دستورو څخه جوړه وي او يو مشخص کار ترسره کوي .هر پروګرام په C++ کې لږ تر لږه يو تابع لري چې هغه main() دی، خو يو پروګرام کولای شي څو تابع ولري .د پروګرام د پيل نقطه main() تابع وي.
په C++ ژبه کې توابع عموماً په دوو برخو وېشل کېږي:

1. د کتابخانې توابع(Library Functions)

2. د کارونکي لخوا تعريف شوې توابع(User-defined Functions)

۱.۳۰د کتابخانې توابع

ځينې توابع چې په ډېرو برنامو کې کارول کېږي، مخکې له مخکې ليکل شوي وي او د کمپايلر سره يو ځای وي .لکه:

sin()د زاويې محاسبې لپاره

cos()

clrscr()د سکرين پاکولو لپاره

دې ډول توابعو ته کتابخانوي توابع ویل کیږي .مور یوازې د هغوی نوم کاروو او په مناسب ډول یې استعمالوو.

1.8

مثال 8.1لاندې برنامه د کتابخانوي تابع <iostream.h> په مرسته د "Hello World!" پیغام ښیي:

```
// program to Print “Hello World!” using Library Function
#include <iostream.h>
main ( )
{
cout << " Hello World! ";
return 0;
}
```

Output

Hello world!

مثال 2.8 لاندې برنامه د کتابخانوي تابع <math.h> په مرسته د داخل شوي عدد مربع جذر (Square Root) محاسبه کوي:

```
// program to find Square Root of any number inserted from keyboard
#include <iostream.h>
#include <math.h>// required for sqrt function
main ( )
{
// Declaration
double Number, SQR;
cout << "Enter a number to find its Square Root : ";
cin >> Number;
SQR = sqrt (Number);
cout << " The square root of "<< Number << " = " << SQR;
return 0;
}
```

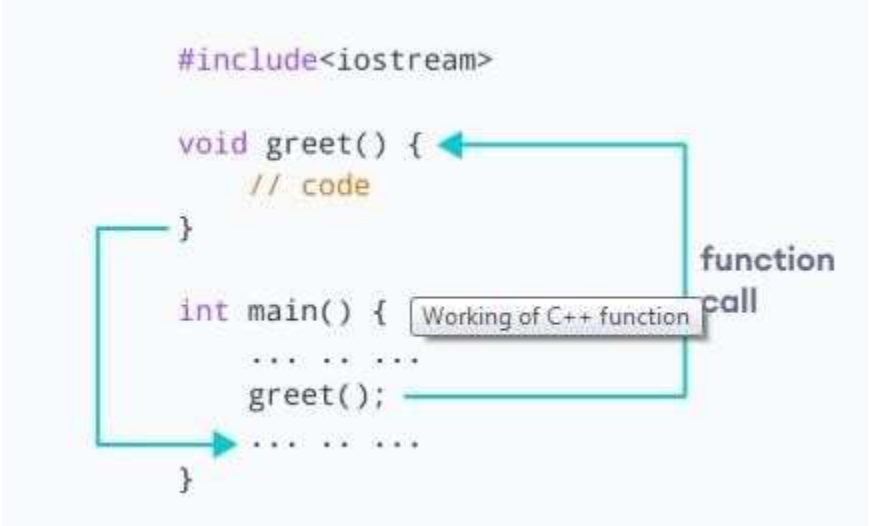
Output

```
Enter a number to find its Square Root : 36
The square root of 36 = 6
```

د کارونکي لخوا تعریف شوي توابع

د کارونکي لخوا تعریف شوي تابع هغه تابع ده چې موږ يې خپله جوړوو او د اړتیا په وخت کې يې په برنامه کې کاروو (call) کوو.

د تابع د جوړولو لپاره، تابع د main() په شان د پروګرام په یوه برخه کې تعریف کېږي، او وروسته يې د main() تابع دننه د اړتیا په ځای کې call کوو.



شکل 1.8 د تابع کارکر

د مثال 3.8 لاندې برنامه د کارونکي لخوا تعریف شوې تابع په کارولو سره "Hello World!"ښیي

```
// program to Print “Hello World!” using user defined function
#include <iostream.h>
void greet() {
cout <<”Hello World!”;
}
main ( )
{
// calling the function
greet();
return 0;
}
```

Output

Hello World!

۳۲.۱انوشتن توابع

هر تابع څلور بېلابېل برخې لري چې عبارت دي له:

1. د تابع اعلان(Function Declaration)
2. د تابع تعريف(Function Definition)
3. د تابع د بيرته ورکوونکي قيمت(Return Value)
4. د تابع غږول(Function Call)

د تابع اعلان(Function Declaration)

د دې لپاره چې کمپايلر پوه شي په يوه برنامه کې کوم تابعونه موجود دي او د هغوی کارول په نورو برخو کې ممکن شي، بايد تابع د main() تابع نه مخکې اعلان شي.

د تابع اعلان کمپايلر ته د تابع د نوم، د بيرته ستنېدونکي قيمت ډول، او د پارامترونو ډولونه ښيي.

د تابع نوم :هر تابع بايد يو نوم ولري .د نوم ټاکل د متغيرونو د نومولو له قوانينو پيروي کوي .د تابع نوم بايد د هغه کار سره سم وي چې تابع يې ترسره کوي.لکه :د جمع لپاره add ياsum

د تابع د بيرته ستنېدونکي قيمت ډول :هر تابع د خپلو داخل شوو قيمتونو په اساس نتيجه بيرته ورکوي.تابع کولای شي هيڅ قيمت، يو قيمت يا څو قيمتونه بيرته ورکړي.که تابع هيڅ قيمت نه ورکوي، نو void وي.که يو عدد بيرته ورکړي، نو د مثال په ډول int وي.

د پارامترونو ډول :تابع ممکن يو يا څو داخلېدونکي قيمتونه ولري، او کله ناکله هيڅ داخلېدونکی قيمت هم نه لري.که تابع هيڅ input ونه لري، نو void د argument په توګه ليکل کېږي.

په عمومي ډول د تابع اعلان داسې دی:

```
return_type function_name(parameter list)
```

د تابع تعريف(Function Definition)

دا برخه د تابع اصلي بدنه جوړوي او ټول هغه دستورونه چې يو تابع يې بايد ترسره کړي په دې برخه کې معرفي کيږي.

```
return_type function_name( parameter list ) {
  //Function Definition
  body of the function
}
```

د تابع بېرته راګرځېدونکي قيمتون(Function Return Value)

هغه مشخص قيمت چې له يو تابع څخه بېرته راګرځي، د تابع د بېرته راګرځېدونکي قيمت په نوم يادېږي .کله چې د بېرته راګرځولو بيانيه اجرا شي، بېرته راګرځېدونکی قيمت له تابع څخه د تابع غږوونکي ته بېرته ستنېږي .يو تابع کولای شي هېڅ قيمت بېرته راونه ګرځوي يا يو او څو قيمتونه بېرته راوګرځوي.

هغه تابعي چې هېڅ مقدار نه بېرته راګرځوي:

کله ناکله په پروګرام کې له داسې تابعو څخه استفاده کوو چې هغه تابعي د غږېدو وروسته غوښتل شوي عمليات ترسره کوي او تمه شوي خروجی توليد او چاپ کوي او هېڅ مقدار بېرته غږوونکي تابع ته نه سپاري.

مثال 4.8: لاندې پروګرام هغه تابع بنيي چې نه ورودي لري او نه خروجي، او يوازې پيغام د کاروونکي له خوا چوړ شوي تابع کې **print** کوي.

```
// Function that return no value
#include <iostream.h>
void print(void);
int main()
{
print();
return 0;
}
void print( void )
{
cout << " No value Function!";
}
```

Output

```
No value Function!
```

هغه تابعي چې يو مقدار بېرته راګرځوي :دا تابعي له غږېدو وروسته يو قيمت بېرته غږوونکي تابع ته ستنوي.

مثال 5.8: لاندې پروګرام د X متحول قيمت تابع Scope() ته لېږي او بېرته يې راګرځوي.

```
// program to Program to return value of A from function Scope()
#include <iostream.h>
void scope(int);
int main()
{
int x = 10;
cout << "first value of A=" << x << "\n";
scope(x);
cout << "third value of A=" << x;
return 0;
}
void scope(int a)
{
a++;
cout << "second value of A=" << a << "\n";
}
```

Output

```
first value of A= 10
second value of A= 11
third value of A= 10
```

مثال 6.8: لاندې پروګرام يو عدد اخلي، په factorial() تابع کې يې فکتوريل محاسبه کوي او بېرته يې راګرځوي.

```
// Program to Find Factorial using user define Function
#include <iostream.h>
```

```
int factorial(int);
int main()
{
int n;
cout<<"Enter a number to find factorial: ";
cin >> n;
cout << "Factorial of " << n <<" = " << factorial(n);
return 0;
}
int factorial(int n)
{
if (n > 1)
{
return n*factorial(n-1);
}
else
{
return 1;
}
}
```

Output

```
Enter a number to find factorial: 5
Factorial of 5 = 120
```

د تابع غږول(Call Function)

کله چې غواړو له تابع څخه استفاده وکړو(په اصلي پروګرام کې)باید د تابع د نوم په وسیله هغه وغواړو .

په دې ډول چې د اړوند تابع نوم لیکو او د تابع د نوم مخې ته په پارامتر کې هغه قیمتونه لیکو چې غواړو تابع ته یې ولیږو.

تر اوسه مو په C++ کې ډېر کتابخانه‌يي تابعې غږولي دي.

هر تابع کولای شي بل تابع وغواړي يا call کړي، په دې شرط چې د غږول شوي تابع اعلان د غږوونکي تابع څخه مخکې موجود وي.

نو له هغې نقطې څخه چې د تابع اعلان موجود وي، هغه تابع د پروګرام د ټولو نورو تابعو لپاره د پېژندلو او غږولو وړ وي.

همدارنګه هر فرعي تابع له بل فرعي تابع څخه هم غږېدلی شي، خو يوازې main() تابع ده چې د پروګرام نور تابعې يې نشي غږولی.

مثال 7.8: لاندې پروګرام د دوو عددونو ترمنځ تر ټولو کوچنی عدد د کارووونکي له خوا معرفي شوي فنکشن کې ښيي(د تابع د غږولو طريقه)

```
// program to the minimum number using function
#include <iostream.h>
int minimum(int a , int b);
int main()
{
int num1 , num2 , result;
cout<<"Enter 2 numbers to find the minimum : "<<endl;
cin>>num1>>num2;
result=minimum(num1 , num2);
cout<<"the minimum number between "<<num1<<" and "<<num2<<" is
"<<result<<endl;
system("pause");
return 0;
}
int minimum(int a , int b)
{
if(a<b)
return a;
if(b<a)
return b;
return 1;
```

Output

first value of A= 10

second value of A= 11

third value of A= 10

۱.۳۳ د تابعونو ته د پارامترونو د لیږلو طریقې

پارامترونه په دوه طریقو له غږوونکي تابع څخه غږېدونکي تابع ته لیږل کېدای شي .دغه طریقې عبارت دي له:

۱ - د قیمت له لارې غږول(Call by Value)

۲ - د آدرس له لارې غږول(Call by Reference)

د قیمت له لارې غږول(Call by Value)

په دې طریقه کې د پارامتر یوه کاپي د کمپیوټر په حافظه کې ساتل کېږي، او هغه بدلونونه چې د تابع په تعریف کې پر متغیر راځي، د تابع څخه بهر د اصلي متغیر پر قیمت هیڅ اغیز نه کوي.

مثال 8.8: لاندې پروګرام 10 عدد مربع کوي او په output کې يې ښيي.

```
// Call by Value
#include <iostream.h>
int sqr(int x);
int main()
{
int num = 10;
cout << "square = " << sqr(num) << "and number = " << num;
return 0;
}
int sqr(int x);
{
x = x * x;
return x;
}
```

Output

```
square = 100 and number = 10
```

مثال 9.8: لاندې پروګرام د X قیمت بېرته راګرځوي.

```
// Call by value
#include <iostream.h>
int returnEight()
```

```
{
return 8; // return the specific value 8 back to the caller
}
int main()
{
cout <<"return value is: "<< returnEight() << '\n';
cout <<"return value +2 is: "<< returnEight() + 2 << '\n';
returnEight();
return 0;
```

Output

```
return value is: 8
return value +2 is:10
```

د آدرس له لارې غږول (Call by Reference)

په دې طريقه کې، بدلونونه د موجود آدرس په متغير کې کاپي کېږي، او د تابع څخه بهر د پارامتر بدلونونه د تابع دننه د متغير له بدلونونو سره تړلي وي.

مثال 10.8: لاندې پروګرام د x او y قيمتونه له يو بل سره بدلوي.

```
// Call by reference
#include<iostream.h>
void swap(int *x, int *y)
{
int swap;
swap=*x;
```

```
*x=*y;
*y=swap;
}
int main()
{
int x=500, y=100;
swap(&x, &y); // passing value to function
cout<<"Value of x is: "<<x<<endl;
cout<<"Value of y is: "<<y<<endl;
return 0;
}
```

Output

Value of x is: 100

Value of y is: 500

مثال 11.8: لاندې پروګرام وخت د ثانیو پر اساس اخلي او د یو تابع په وسیله یې ساعت، دقیقه او ثانیې ته بدلوي او په سکرین کې یې ښیي.

```
// The program change the seconds from integer to Clock format
#include <iostream.h>
void convert(int s);
int main()
{
int seconds;
```

```
cin>> seconds;
convert(seconds);
return 0;
}
void convert(int s)
{
int second,h,m;
second = s % 60;
h= s/3600;
m=(s/60) % 60;
cout<< "Time is: " << h<< ":"<< m << ":" << second;
}
```

Output

```
2500
0:41:40
```

مثال 12.8: لاندې پروګرام د کیبورد له لارې داخل شوو دوو عددونو ترمنځ تر ټولو لوی عدد د کاروونکي له خوا معرفي شوي تابع په وسیله ښیي.

```
// The program find the maximum number between two numbers
#include <iostream.h>
int max(int num1, int num2);
int main () {
int a;
int b;
int ret;
cout<<"Enter two values for a and b \n";
cin>>a>>b;
ret = max(a, b);
cout << "Max value is : " << ret << endl;
return 0;
}
int max(int num1, int num2) {
int result;
if (num1 > num2)
result = num1;
else
result = num2;
return result;
}
```

Output

```
Enter two values for a and b
560 500
Max value is : 560
```

مثال 13.8: لاندې پروګرام له کیبورډ څخه دوه عددونه اخلي، د sum تابع ته یې لیږي او بیا د هغوی مجموعه بېرته اصلي تابع ته راګرځوي.

```
// The program take to numbers sum them in a function and return the sum
#include <iostream>
using namespace std;
float sum(float, float); // = float sum(float num1, float num2);
int main()
{
float num1, num2 ,numSum;
cout << "Enter first number :" ;
cin >> num1;
cout << "Enter second number :" ;
cin >> num2;
numSum = sum(num1, num2);
cout <<"The sum is: " <<numSum;
return 0;
}
float sum(float f1, float f2) // float sum(float num1, float num2);
{
float fSum = f1 + f2;
return fSum;
}
```

Output

```
Enter first number: 20
Enter second number: 25
The sum is: 45
```