**OOP in C++ : အပိုင္း (၈) – Relational Operators**

စာေရးသူအေနနဲ႔ Robert Lafore ေရးတဲ႔ Object-Oriented Programming in C++ စာအုပ္ေလးကို ကိုးကား ဘာသာျပန္ၿပီး ေရးသားခဲ႔တာ အခုဆိုရင္ အပိုင္း(၈) ေရာက္ခဲ႔ပါၿပီ။ ယခင္က ေရးသားခဲ႔သမွ် အပိုင္း (၇) ပိုင္းဟာ အဲဒီစာအုပ္ထဲက Chapter(2) C++ Programming Basics ကို ဆီေလ်ာ္ေအာင္ ဘာသာျပန္ထားခဲ႔တာပါ။ ယခု အပတ္မွာေတာ႔ အဲဒီ Chapter(2) တစ္ခုလံုးရဲ႕ အႏွစ္ခ်ဳပ္ကို ျပန္လည္ ေဆြးေႏြးၿပီး relational operators မ်ားအေၾကာင္းကို ဆက္လက္ တင္ျပသြားမွာပါ။

**Summary (Chapter 2)**

ဒီ Chapter မွာ C++ ရဲ႕ အေျခခံအက်ဆံုး အစိတ္အပိုင္းက function ေတြ ျဖစ္တယ္ဆိုတာကို ေလ႔လာခဲ႔ၿပီးပါၿပီ။ ပရိုဂရမ္ တစ္ပုဒ္မွာ ပထမဆံုး အလုပ္လုပ္တဲ႔ function ကေတာ႔ main() ပဲ ျဖစ္ပါတယ္။ function တစ္ခုကို statements ေတြနဲ႔ ဖြဲ႕စည္းထားတာ ျဖစ္ၿပီး ၄င္းတို႔က ကြန္ပ်ဴတာကို ညႊန္ၾကား ခိုင္းေစေနတာပဲ ျဖစ္ပါတယ္။ Statements တိုင္းဟာ semicolon (;) နဲ႔ အဆံုးသတ္ေလ႔ ရွိပါတယ္။ Statement တစ္ခုမွာ expressions ေတြ တစ္ခုထက္ မက ပိုၿပီး ပါႏိုင္ပါတယ္။ Expression ဆိုတာကေတာ႔ တန္ဖိုးတစ္ခု ထြက္လာေအာင္ variables ေတြနဲ႔ operators ေတြ စုစည္းၿပီး ေရးသားထားတာပဲ ျဖစ္ပါတယ္။

C++ မွာ output value ေတြကို screen မွာ ျပသေပးဖို႔ cout object နဲ႔ insertion operator (<<) ကို အသံုးျပဳေလ႔ ရွိပါတယ္။ Input ေတြကို ဖတ္ယူဖို႔ အတြက္ေတာ႔ cin နဲ႔ extraction operator (>>) ကို အသံုးျပဳၿပီး standard input device (keyboard) ကေန ထည္႔သြင္းေပးလိုက္တဲ႔ တန္ဖိုးေတြကို variable ေတြထဲကို ထည္႔သြင္းေပးပါတယ္။

C++ မွာ built in data types ေတြ အမ်ားအျပားရွိပါတယ္။ char,int,long နဲ႔ short တို႔ဟာ integer data type ေတြျဖစ္ၾကၿပီး float, double နဲ႔ long double တို႔ကေတာ႔ floating-point types ေတြပဲ ျဖစ္ပါတယ္။ အဲဒီ types အားလံုးဟာ အေပါင္းနဲ႔ အႏုတ္ တန္ဖိုး ႏွစ္ခုလံုး ထည္႔သြင္းလို႔ရတဲ႔ (signed) types ေတြပဲ ျဖစ္ပါတယ္။ integer types ေတြကို အႏုတ္တန္ဖိုး ထည္႔သြင္းဖို႔ မလိုအပ္တဲ႔ အေျခအေနမ်ိဳးမွာ unsigned ဆိုတဲ႔ keyword ကို အသံုးျပဳျခင္းျဖင္႔ အရြယ္အစား ႏွစ္ဆ ပိုႀကီးလာေစပါတယ္။ true နဲ႔ false တန္ဖိုး ႏွစ္ခုသာ ပါဝင္တဲ႔ Boolean variables ေတြကို သိမ္းဆည္းဖို႔ bool type အသံုးျပဳႏိုင္ပါတယ္။

const ဆိုတဲ႔ keyword ေလးကေတာ႔ variable တစ္ခုကို constant အသြင္ေျပာင္းပစ္ၿပီး ၄င္းထဲမွာ ထည္႔သြင္းထားတဲ႔ တန္ဖိုးကို ေျပာင္းလဲခြင္႔ မေပးေတာ႔ပါဘူး။

Variable တစ္ခုဟာ မတူညီတဲ႔ data types ေတြ ေရာေႏွာေရးထားတဲ႔ expressions ေတြထဲမွာ အလိုအေလ်ာက္ conversion ျပဳလုပ္ခံရေလ႔ ရွိၿပီး အကယ္၍ programmer က သတ္မွတ္ေပးထားတဲ႔ data type သို႔စိတ္ႀကိဳက္ conversion ျပဳလုပ္ခ်င္ရင္ေတာ႔ casting လုပ္ေပးရမွာပါ။

C++ မွာ သံုးေနက် +, -, \*, / ေတြ အျပင္ အၾကြင္းရွာဖို႔အတြက္ remainder operator (%) ကိုပါ ထည္႔သြင္းေပးထားပါတယ္။

Arithmetic assignment operator ေတြျဖစ္တဲ႔ +=, -= စတာေတြဟာ arithmetic operation နဲ႔ assignment operation ေတြကို တစ္ဆက္တည္း လုပ္ေဆာင္ေပးပါတယ္။ (ဥပမာ - operator ညာဘက္က တန္ဖိုးကို ဘယ္ဘက္က variable နဲ႔ ေပါင္း/ႏုတ္/ေျမွာက္/စား ၿပီး ရလာဒ္ကို အဲဒီ variable ထဲသို႔ ျပန္ထည္႔ေပးလိုက္တာ)။ Increment and decrement operators (++,--) ေတြကေတာ႔ variable ထဲကို ၁ ေပါင္းျခင္း သို႔မဟုတ္ ႏုတ္ျခင္း ျပဳလုပ္ေပးမွာ ျဖစ္ပါတယ္။

Preprocessor directives ေတြကေတာ႔ statement ေတြလို မဟုတ္ပါဘူး။ ၄င္းတို႔က computer ကို ညႊန္ၾကားခ်က္မေပးဘဲ compiler ကို ညႊန္ၾကားခ်က္ေပးတာပဲ ျဖစ္ပါတယ္။ #include ဆိုတဲ႔ directive ကေတာ႔ လက္ရွိ source ဖိုင္ ထဲကို အျခား ဖိုင္တစ္ခု ထည္႔သြင္းေပးဖို႔ compiler ကို ညႊန္ၾကားတာပဲ ျဖစ္ပါတယ္။ #define ဆိုတဲ႔ directive ကေတာ႔ ညႊန္ၾကားခ်က္ တစ္ခုကို ေနာက္ညႊန္ၾကားခ်က္ တစ္ခုနဲ႔ အစားထိုးဖို႔ ေစခိုင္းတာပါ။ using ဆိုတဲ႔ directive ကေတာ႔ compiler ကို namespace တစ္ခုရဲ႕ နာမည္ေတြ မွတ္သားထားေစတာပဲ ျဖစ္ပါတယ္။

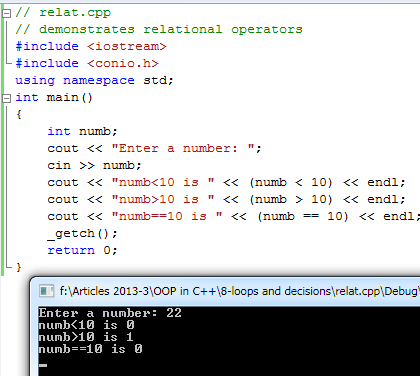
ကၽြန္ေတာ္တို႔ ပရိုဂရမ္ေတြထဲမွာ library function ေတြကို ေခၚယူအသံုးျပဳခဲ႔မယ္ ဆိုရင္ အဲဒီ function ရဲ႕ တကယ္႔ machine-code ေတြက သက္ဆိုင္ရာ library file ေတြထဲမွာ ရွိေနၿပီး အလိုအေလ်ာက္ ခ်ိတ္ဆက္ အသံုးျပဳမွာ ျဖစ္ပါတယ္။ ဒါေပမယ္႔ အဲဒီ function ရဲ႕ declaration ေတြပါဝင္တဲ႔ header file ကိုေတာ႔ ကၽြန္ေတာ္တို႔ source file ထဲမွာ #include ကို အသံုးျပဳၿပီး ထည္႔သြင္းေပးရမွာ ျဖစ္ပါတယ္ခင္ဗ်ာ။

**Relational Operators**

ပရိုဂရမ္ အမ်ားစုသည္ အစမွ အဆံုးအထိ အစဥ္လိုက္ အလုပ္လုပ္သည္႔ ပံုစံမ်ိဳး မရွိၾကပါ။ အသံုးျပဳသူ ဆံုးျဖတ္သည္႔အတိုင္း အေျခအေနေျပာင္းလဲမႈကို လိုက္ေလ်ာညီေထြျဖစ္ေအာင္ ျပဳမူသည္႔ ပံုစံမ်ိဳး ေရးသားေလ႔ ရွိၾကသည္။ ပရိုဂရမ္အတြင္းရွိ တြက္ခ်က္မႈမ်ားေပၚ တူတည္၍ Control flow ကို ပရိုဂရမ္ ၏ တစ္ေနရာမွ အျခားတစ္ေနရာသို႔ jump လုပ္ကာ ေဆာင္ရြက္ေလ႔ ရွိသည္။ ထိုသို႔ jump ျဖစ္ေစႏိုင္ေသာ statements မ်ားကို control statements မ်ားဟု ေခၚသည္။ ၄င္းတို႔ကို loops ႏွင္႔ decisions ဟူ၍ ေခါင္းစဥ္ႏွစ္ခု ခြဲျခား ေလ႔လာႏိုင္ပါသည္။

loop တစ္ခုကို အႀကိမ္မည္မွ် ပတ္မည္ သို႔မဟုတ္ decision တစ္ခုကို မည္သို႔လုပ္ေဆာင္မည္ ဆိုသည္႔ အခ်က္သည္ သက္ဆိုင္ရာ expression မ်ား မွန္သည္ မွားသည္ ဆိုသည္႔ အေပၚ မူတည္ေနေပသည္။ အဆိုပါ expression မ်ား အတြင္း၌ ရွိေသာ တန္ဖိုးႏွစ္ခုကို ႏိႈင္းယွဥ္ရန္ အသံုးျပဳသည္႔ operator မ်ားကို relational operator ဟု ေခၚဆိုၾကသည္။ Loops ႏွင္႔ decisions မ်ားကို ေလ႔လာရာတြင္ ၄င္း relational operator မ်ားအား အသံုးခ်ရသျဖင္႔ ဦးစြာ ရွင္းလင္း တင္ျပေပးပါမည္။

Relational Operators မ်ားသည္ C++ ၏ မည္သည္႔ built-in data type တန္ဖိုး ႏွစ္ခုကိုမဆို ႏိႈင္းယွဥ္ေပးႏိုင္ပါသည္။ ထို႔ျပင္ ေနာက္ပိုင္းေလ႔လာၾကရမည္႔ user-defined classes မ်ားကိုလည္း ႏိႈင္းယွဥ္ေပးႏိုင္စြမ္း ရွိပါသည္။ ထိုသို႔ ႏိႈင္းယွဥ္ရာတြင္ (==) တူညီသည္၊ (>) ႀကီးသည္၊ (<) ငယ္သည္ဟူေသာ ႏိႈင္းယွဥ္ခ်က္မ်ား ျပဳလုပ္ေလ႔ ရွိပါသည္။ ႏိႈင္းယွဥ္ခ်က္ ေဖာ္ျပခ်က္၏ ရလာဒ္မွာ မွန္လွ်င္ true ျဖစ္၍ မွားလ်င္ false ျဖစ္သည္။ ပိုမိုရွင္းလင္း လြယ္ကူစြာ နားလည္ သေဘာေပါက္ေစရန္ relat.cpp ပရိုဂရမ္ အား ေလ႔လာၾကည္႔ၾကပါစို႔။



အထက္ပါ ပရိုဂရမ္တြင္ user ထည္႔သြင္းေပးလိုက္ေသာ တန္ဖိုးႏွင္႔ ၁၀ အား ႏိႈင္းယွဥ္မႈ ၃ ခု ျပဳလုပ္ထားပါသည္။ user မွ 22 အား input အျဖစ္ ထည္႔သြင္းေပးလိုက္ခ်ိန္တြင္ ႏိႈင္းယွဥ္မႈ ၃ ခု၏ ရလာဒ္မ်ားကို အထက္ပါအတိုင္း မွန္လွ်င္ 1 မွားလွ်င္ 0 ျဖင္႔ ေဖာ္ျပထားသည္ကို ေတြ႔ရပါသည္။ ပထမ ႏိႈင္းယွဥ္ခ်က္မွာ numb (22) သည္ 10 ထက္ ငယ္သလား ဟူေသာ ႏိႈင္းယွဥ္ခ်က္ျဖစ္ၿပီး မွားယြင္းသျဖင္႔ ရလာဒ္မွာ 0 ျဖစ္ပါသည္။ ဒုတိယ ႏိႈင္းယွဥ္ခ်က္မွာ numb (22) သည္ 10 ထက္ ႀကီးသလား ဟူေသာ ႏိႈင္းယွဥ္ခ်က္ျဖစ္၍ မွန္ကန္ေသာေၾကာင္႔ ရလာဒ္မွာ 1 ျဖစ္သည္။ ေနာက္ဆံုး ႏိႈ္င္းယွဥ္ခ်က္မွာ numb (22) သည္ 10 ႏွင္႔ ညီသလား ဟူေသာ ႏိႈင္းယွဥ္ခ်က္ျဖစ္ၿပီး မွားယြင္းေသာေၾကာင္႔ ရလာဒ္မွာ 0 ျဖစ္ပါသည္။ ကၽြန္ေတာ္တို႔ အေနျဖင္႔ ရလာဒ္မ်ားကို ture, false မ်ားျဖင္႔ ေမွ်ာ္လင္႔ထားေသာ္လည္း တကယ္တမ္းတြင္ compiler က 1,0 ျဖင္႔သာ အေျဖ ထုတ္ေပးပါသည္။ သို႔ျဖစ္၍ အဆိုပါ ျပႆနာတြင္ compiler အတြက္ 1 သည္ true ျဖစ္၍ 0 သည္ false အျဖစ္ သတ္မွတ္ထားေၾကာင္း ထင္ရွားပါသည္။ အကယ္၍ ကၽြန္ေတာ္တို႔အေနျဖင္႔ bool data type ကို အသံုးျပဳခဲ႔လွ်င္ပင္ အေျဖအား အထက္ပါအတိုင္းသာ ရရွိမည္ ျဖစ္သည္။ အဘယ္ေၾကာင္႔ဆိုေသာ္ C++ စတင္ေပၚေပါက္ကာစက bool data type မရွိေသးေသာေၾကာင္႔ ျဖစ္ပါသည္။ ထို႔ေၾကာင္႔ ယခုအခါ C++ တြင္ အမွားမွန္ကို 0,1 သာမက true,false ႏွင္႔ပါ ေဖာ္ျပႏိုင္ေပသည္။

လက္ေတြ႔တြင္မူ အဆိုပါ ကြဲျပားခ်က္သည္ သိပ္အေရးမပါလွပါ။ ကၽြန္ေတာ္တို႔ အေနျဖင္႔ ture, false ျဖင္႔ ေဖာ္ျပျခင္း၊ မေဖာ္ျပျခင္းထက္ looping တစ္ခု သို႔မဟုတ္ decision တစ္ခု ဘာဆက္လုပ္ရမည္ ဆိုသည္ကို ဆံုးျဖတ္ရန္သာ လိုအပ္ေပသည္။

ေအာက္ပါ ဇယားတြင္ C++ ၏ relational operators မ်ားအားလံုးကို ေဖာ္ျပေပးထားပါသည္။

|  |  |
| --- | --- |
| Operator | Meaning |
| > | Greater than |
| < | Less than |
| == | Equal to |
| != | Not equal to |
| >= | Greater than or equal to |
| <= | Less than or equal to |

အထက္ပါ relational operators မ်ားအား အသံုးျပဳပံုကို ေအာက္တြင္ နမူနာ expression မ်ားျဖင္႔ ေဖာ္ျပထားပါသည္။ ပထမ ႏွစ္ေၾကာင္းမွာ harry ႏွင္႔ jane ဆိုေသာ variable မ်ားထဲသို႔ တန္ဖိုးမ်ား သတ္မွတ္ ထည္႔သြင္းျခင္းျဖစ္ၿပီး က်န္စာေၾကာင္းမ်ားက ၄င္းတန္ဖိုးမ်ားကို relational operators မ်ားအသံုးျပဳ၍ အမ်ိဳးမ်ိဳး ႏိႈင္းယွဥ္ ျပထားျခင္းပင္ ျဖစ္သည္။

jane = 44; //assignment statement

harry = 12; //assignment statement

(jane == harry) //false

(harry <= 12) //true

(jane > harry) //true

(jane >= 44) //true

(harry != 12) // false

(7 < harry) //true

(0) //false (by definition)

(44) //true (since it’s not 0)

မွားေလ႔ရွိေသာ အခ်က္တစ္ခုမွာ equal operator (==) ကို equal signs ႏွစ္ခု သံုးရမည္႔အစား တစ္ခုတည္း သံုးမိျခင္းျဖစ္သည္။ ထိုအမွားမ်ိဳးသည္ compiler က မွားမွန္း မသိျခင္းေၾကာင္႔ ရွာေဖြရန္ ခက္ခဲေလ႔ ရွိသည္။

C++ အေနျဖင္႔ true ကို 1 ျဖင္႔ ကိုယ္စားျပဳေလ႔ ရွိေသာ္လည္း သုညမဟုတ္သည္႔ မည္သည္႔ ဂဏန္း (အႏုတ္တန္ဖိုးအပါအဝင္) ကိုမဆို true အျဖစ္ လက္ခံေလ႔ ရွိသည္။ false အတြက္မူ သုည 0 တစ္ခုတည္းသာ သတ္မွတ္ထားသည္။

ေနာက္အပတ္ေတြမွာေတာ႔ looping ေတြအေၾကာင္းကို ဆက္လက္ ေဆြးေႏြး သြားပါမယ္ခင္ဗ်ာ။ ဤအခန္းဆက္ ေဆာင္းပါးမ်ားဟာ IT ေက်ာင္းသားမ်ားအတြက္ အထူးရည္ရြယ္ၿပီး ေရးသားထားတာ ျဖစ္ပါတယ္။ သိလိုသည္မ်ား ကို aungwh2013@gmail.com ကို ဆက္သြယ္ၿပီး ေမးျမန္းႏိုင္သလို YCC ေက်ာင္းသားမ်ား အတြက္လည္း ေအာက္ေဖာ္ျပပါ လိပ္စာရွိ ေတာ္ဝင္ ကြန္ပ်ဴတာ စင္တာ ၌လည္း ေလ႔လာ စံုစမ္းႏိုင္ပါတယ္ခင္ဗ်ာ။

**အကိုးအကား**

Object-Oriented Programming in C++(4th edition), Robert Lafore, Copyright©2002 by Sams Publishing: ISBN 0-672-32308-7

Dr. ေအာင္ဝင္းထြဋ္ (bluephoenix)

http://www.tech4mm.com

ေတာ္ဝင္ ကြန္ပ်ဴတာ စင္တာ

၁၇၉ စ၊ သုမဂၤလာ၊ ေစ်းေလး အေနာက္ဘက္၊ ျပင္ဦးလြင္ၿမိဳ႕