**OOP in C++ : အပိုင္း (၁၅) - Conditional Operator and Logical Operators**

ဒီအပတ္ သင္ခန္းစာမွာေတာ႔ ဆံုးျဖတ္ခ်က္ေတြခ်ဖို႔အတြက္ သံုးၾကရတဲ႔ Operators ေတြအေၾကာင္းကို ေလ႔လာၾကရေအာင္။

**The Conditional Operator**

Conditional Operator ဟာ အနည္းငယ္ ထူးဆန္းတဲ႔ decision operator တစ္ခုျဖစ္ပါတယ္။ ပရိုဂရမ္ ေရးသားရာမွာ ေတြ႕ၾကံဳရေလ႔ ရွိတဲ႔ - အေျခအေန တစ္ခု မွန္ရင္ variable တစ္ခုထဲ တန္ဖိုး တစ္ခု သတ္မွတ္ေပးၿပီး မွားသြားခဲ႔ရင္ အျခားတန္ဖိုးတစ္ခု သတ္မွတ္ေပးရေလ႔ ရွိတဲ႔ အခါမ်ိဳးမွာ အသံုးျပဳဖို႔အတြက္ ဖန္တီးထားတာပါ။ ဥပမာအားျဖင္႔ ငယ္တဲ႔ တန္ဖိုးရွာတဲ႔ ျပႆနာမ်ိဳးပါ။ အယ္ဖာ နဲ႔ ဘီတာမွာ အယ္ဖာက ငယ္ရင္ min ထဲကို အယ္ဖာ တန္ဖိုး ထည္႔ေပးၿပီး မွားခဲ႔ရင္ (ဘီတာက ငယ္ခဲ႔ရင္) min ထဲကို ဘီတာ တန္ဖိုး ထည္႔ေပးတဲ႔ အေျခအေနမ်ိဳး ျဖစ္ပါတယ္။ အဲဒီ အေျခအေနကို if...else သံုးၿပီး ေအာက္ပါအတိုင္း ေရးလို႔ ရပါတယ္။

if( alpha < beta )

min = alpha;

else

min = beta;

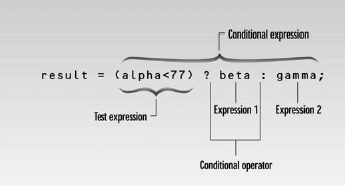
ဒီလို အေျခအေနမ်ိဳးကို မၾကာခဏ ေတြ႕ၾကံဳရတာေၾကာင္႔ C နဲ႔ C++ ကို တီထြင္ခဲ႔သူက ပိုမိုတိုေတာင္းတဲ႔ ေရးသားနည္းကို တီထြင္ရင္း conditional operator ေပၚေပါက္လာရျခင္း ျဖစ္ပါတယ္။ ဒီ operator မွာ သေကၤတ ႏွစ္ခု ပါၿပီး operand သံုးခု ပါဝင္ပါတယ္။ C++ မွာရွိတဲ႔ အျခား operator မ်ား အားလံုးဟာ operand တစ္ခု သို႔မဟုတ္ ႏွစ္ခုကို လုပ္ေဆာင္ေပးပါတယ္။ Conditional operator ကေတာ႔ တစ္ခုတည္းေသာ operand သံုးခုကို လုပ္ေဆာင္ေပးတဲ႔ operator ျဖစ္ပါတယ္။

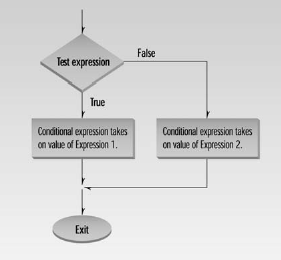
အထက္က ကုဒ္ကို conditional operator သံုးၿပီး ေရးမယ္ဆိုရင္ ေအာက္ပါအတိုင္း ရပါတယ္-

min = (alpha<beta) ? alpha : beta;

အဲဒီ statement ထဲက equal sign ညာဘက္က အျခမ္းျဖစ္တဲ႔ (alpha<beta) ? alpha : beta ဆိုတာ conditional expression ပဲ ျဖစ္ပါတယ္။ အဲဒီထဲမွာ ပါတဲ႔ question mark (?) နဲ႔ colon (:) တို႔ေပါင္းၿပီး conditional operator ျဖစ္လာတာပါ။

Question mark ေရွ႕က (alpha<beta) ဆိုတာကေတာ႔ test expression ပဲျဖစ္ပါတယ္။ အဲဒီ တစ္ခုအပါအဝင္ alpha နဲ႔ beta သံုးခုလံုးဟာ operand ေတြ ျဖစ္ၾကပါတယ္။ တကယ္လို႔ test expression မွန္ခဲ႔မယ္ဆိုရင္ conditional expression တစ္ခုလံုးရဲ႕ တန္ဖိုးဟာ question mark ေနာက္က operand ျဖစ္သြားမွာပါ။ ဒီဥပမာမွာေတာ႔ alpha ျဖစ္ပါတယ္။ Test expression မွားသြားခဲ႔ရင္ေတာ႔ colon ေနာက္က operand တန္ဖိုးျဖစ္သြားမွာပါ။ ဒီဥပမာမွာ beta ျဖစ္ပါတယ္။ Test expression မွာပါတဲ႔ လက္သဲကြင္း အဖြင္႔အပိတ္ကို မထည္႔လဲရပါတယ္။ ဒါေပမယ္႔ အစဥ္အလာ ေရးခဲ႔ၾကတာ ျဖစ္ၿပီး ဖတ္ရႈရ ပိုမိုလြယ္ကူေစပါတယ္။ ေအာက္က ပံုေတြမွာ conditional operator ရဲ႕ syntax နဲ႔ flow chat ကို ျပထားပါတယ္။





Conditional expression က ရလာတဲ႔ တန္ဖိုးကို variable တစ္ခုထဲ ထည္႔သြင္းဖို႔ပဲ ျဖစ္ျဖစ္ ဒါမွမဟုတ္ value တစ္ခု အသံုးျပဳႏိုင္တဲ႔ ေနရာတိုင္း သံုးႏိုင္ပါတယ္။ အထက္ပါ ဥပမာမွာေတာ႔ min ဆိုတဲ႔ variable ထဲကို ထည္႔သြင္းေပးလိုက္တာပါ။ ေနာက္ ဥပမာ တစ္ခုအေနနဲ႔ conditional operator ကိုသံုးၿပီး variable n ရဲ႕ absolute value ကို ရွာတဲ႔ ကုဒ္ေလး ေရးျပပါ႔မယ္။ (ဂဏန္း တစ္ခုရဲ႕ Absolute value ဆိုတာက အဲဒီ ဂဏန္းရဲ႕ အႏုတ္လကၡဏာကို ဖယ္ရွားထားတာပါ။ ဒါေၾကာင္႔ အျမဲတမ္း positive ျဖစ္ေနမွာပါ)

absvalue = n<0 ? -n : n;

တကယ္လို႔ n တန္ဖိုးက သုညထက္ ငယ္ခဲ႔ရင္ ? ေနာက္က -n ကို လုပ္ေဆာင္ၿပီး အႏုတ္ႏွစ္ခု ေတြ႕တဲ႔ အတြက္ အေပါင္းတန္ဖိုး ျဖစ္ေစပါမယ္။ တကယ္လို႔ မငယ္ခဲ႔ရင္ အေပါင္းတန္ဖိုး ျဖစ္ေနၿပီးသားျဖစ္လို႔ ေျပာင္းလဲမႈ မရွိဘဲ n ကိုပဲ assign လုပ္ေပးမွာ ျဖစ္ပါတယ္။

Here’s a program, CONDI.CPP, that uses the conditional operator to print an x every eight spaces in a line of text. You might use this to see where the tab stops are on your screen.

// condi.cpp

// prints ‘x’ every 8 columns

// demonstrates conditional operator

#include <iostream>

#include <conio.h>

using namespace std;

int main()

{

for(int j=0; j<80; j++) //for every column,

{ //ch is ‘x’ if column is

char ch = (j%8) ? ' ' : 'x'; //multiple of 8, and

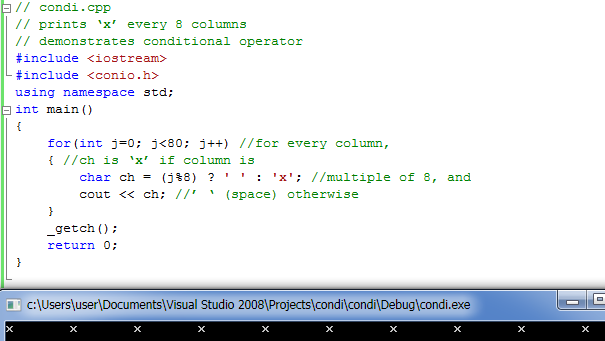
cout << ch; //’ ‘ (space) otherwise

}

\_getch();

return 0;

}



အထက္ပါပံုမွာ space ၇ ခုၿပီးတိုင္း x တစ္ခု ပရင္႔ထုတ္ျပသည္႔ ရလာဒ္ပင္ ျဖစ္သည္။ condi.cpp တြင္ j တန္ဖိုးကို 0 မွ 79 အထိ တစ္ခုခ်င္း for loop ျဖင္႔ တိုးယူလာရင္း conditional operator ကို လုပ္ေဆာင္ပါသည္။ ၄င္းတြင္ ပါဝင္ေသာ test expression (j % 8) က j တန္ဖိုးသည္ ၈ ၏ ဆတိုးကိန္းမ်ား မျဖစ္မခ်င္း (၈ ႏွင္႔စား၍ မျပတ္မခ်င္း) non zero (1-7) ျဖစ္ေနေသာေၾကာင္႔ true ျဖစ္ကာ ? ေနာက္က space ' ' ကို ပရင္႔ထုတ္ေပးေနမည္ ျဖစ္သည္။ ၈ ႏွင္႔ စား၍ ျပတ္ေသာ တန္ဖိုးမ်ားတြင္ အၾကြင္းမွာ သုည ျဖစ္ေနသျဖင္႔ test expression မွာ false ျဖစ္သြားၿပီး x ကို ပရင္႔ထုတ္ေပးျခင္းျဖင္႔ အထက္ပါ ရလာဒ္ ထြက္လာရသည္။ အထက္ပါ ကုဒ္ကို တိုေတာင္းလွၿပီ မထင္ပါႏွင္႔ဦး။ ch ကိုေဖ်ာက္၍ ေအာက္ပါအတိုင္း တစ္ေၾကာင္းတည္း ေပါင္းေရးႏိုင္ပါေသးသည္။

cout << ( (j%8) ? ‘ ‘ : ‘x’ );

C နဲ႔ C++ ပရိုဂရမ္မာ အေတာ္မ်ားမ်ား ဒီလို က်စ္လစ္တဲ႔ ကုဒ္မ်ိဳးကို ႏွစ္ႏွစ္ၿခိဳက္ၿခိဳက္ေရးေလ႔ ရွိပါတယ္။ ကုဒ္တိုတိုေလးနဲ႔ ထိထိေရာက္ေရာက္ ေရးႏိုင္တာ ေပ်ာ္စရာေကာင္းပါတယ္။ ဒါေပမယ္႔ မလိုအပ္တဲ႔အခါမ်ိဳးမွာ အဲလို ကုဒ္မ်ိဳးေတြကို မျဖစ္မေန အခ်ိန္ကုန္ခံေရးေနတိုင္းလဲ အက်ိဳးမရွိ ျဖစ္တတ္ပါတယ္။ ဒါေၾကာင္႔ if...else သံုးၿပီး ကုဒ္နဲနဲ ပိုေရးမလား ဒါမွမဟုတ္ conditional operator ကို သံုးမလားဆိုတာကေတာ႔ ရရွိတဲ႔ အခ်ိန္အေပၚ မူတည္မယ္လို႔ ထင္ျမင္မိပါတယ္။

**Logical Operators**

ေရွ႕သင္ခန္းစာမ်ားမွာ (Conditional Operator မပါ) operator အုပ္စု ႏွစ္ခုကို ေလ႔လာခဲ႔ၾကၿပီးပါၿပီ။ ပထမတစ္ခုကေတာ႔ arithmetic operators (+, -, \*, /, နဲ႔ %) မ်ားျဖစ္ၾကၿပီး ဒုတိယ တစ္ခုကေတာ႔ relational operators (<, >, <=, >=, ==, နဲ႔ !=)မ်ားပဲ ျဖစ္ပါတယ္။

အခု ဆက္လက္ေလ႔လာၾကရမွာကေတာ႔ တတိယအုပ္စုျဖစ္တဲ႔ *logical operators* မ်ားပဲ ျဖစ္ပါတယ္။

အဲဒီ operator မ်ားဟာ (true နဲ႔ false တန္ဖိုး ႏွစ္ခုျဖစ္ႏိုင္တဲ႔ variables) Boolean variables ေတြကို logically ေပါင္းစပ္ေပးႏိုင္စြမ္း ရွိပါတယ္။ ဥပမာအားျဖင္႔ today is a weekday ဟာ မွားလဲ မွားႏိုင္သလဲ မွန္လဲ မွန္ႏိုင္တဲ႔ အတြက္ Boolean value ရွိပါတယ္။ ေနာက္ထပ္ Boolean expression တစ္ခုက Maria took the car ျဖစ္ပါတယ္။ အဲဒီ expression ႏွစ္ခုကို logically ေပါင္းစပ္လို႔ ရပါတယ္။ If today is a weekday, and Maria took the car, then I’ll have to take the bus. ဒီေနရာမွာ သံုးသြားတဲ႔ logical connection ကေတာ႔ and ပဲ ျဖစ္ပါတယ္။ စာေၾကာင္းႏွစ္ခုကို ဆက္စပ္ၿပီးမွ true or false value ထြက္လာေအာင္ ေဆာင္ရြက္ေပးလို႔ပဲ ျဖစ္ပါတယ္။ အခ်က္ႏွစ္ခုလံုးမွန္မွ ဘတ္စ္ကားနဲ႔ သြားရမွာ ျဖစ္ပါတယ္။

**Logical AND Operator**

ကၽြန္ေတာ္တို႔ အေနနဲ႔ Logical AND Operator ကို အသံုးျပဳၿပီး adswitch.cpp ကို ပိုၿပီး စိတ္ဝင္စားစရာ ျဖစ္ေအာင္ ေရးသားၾကည္႔ၾကရေအာင္။ ယခုေရးမယ္႔ advenand.cpp မွာ ရတနာေတြကို (7,11) ဆိုတဲ႔ coordiantes မွာ ဝွက္ထားၿပီး ကစားသူက ရွာေတြ႔မေတြ႔ ၾကည္႔ၾကရေအာင္။

// advenand.cpp

// demonstrates AND logical operator

#include <iostream>

using namespace std;

#include <process.h> //for exit()

#include <conio.h> //for getche()

int main()

{

char dir=’a’;

int x=10, y=10;

while( dir != ‘\r’ )

{

cout << “\nYour location is “ << x << “, “ << y;

cout << “\nEnter direction (n, s, e, w): “;

dir = getche(); //get direction

switch(dir)

{

case ‘n’: y--; break; //update coordinates

case ‘s’: y++; break;

case ‘e’: x++; break;

case ‘w’: x--; break;

}

if( x==7 && y==11 ) //if x is 7 and y is 11

{

cout << “\nYou found the treasure!\n”;

exit(0); //exit from program

}

} //end switch

return 0;

} //end main

ဒီပရိုဂရမ္ တစ္ခုလံုးရဲ႕ လ်ိဳ႕ဝွက္ခ်က္ကေတာ႔ if( x==7 && y==11 ) ဆိုတဲ႔ ကုဒ္ေလးပဲ ျဖစ္ပါတယ္။ Test expression ဟာ x တန္ဖိုး 7 နဲ႔ y တန္ဖိုး 11 ႏွစ္ခုလံုး မွန္မွ၊ မွန္မွာ ျဖစ္ပါတယ္။ အဲဒီ ရလာဒ္ ျဖစ္လာေအာင္ logical AND operator && က relational operator ႏွစ္ခု (x==7) နဲ႔ (y==11) တို႔ကို ေပါင္းစပ္ေပးပါတယ္။ မွတ္သားသင္႔တဲ႔ အခ်က္တစ္ခုက relational expressions ႏွစ္ခုကို လက္သဲကြင္း အဖြင္႔အပိတ္နဲ႔ ကြင္းခတ္ထားဖို႔ မလိုတာပဲ ျဖစ္ပါတယ္။

( (x==7) && (y==11) ) // inner parentheses not necessary

relational operators ေတြဟာ logical operators ေတြထက္ precedence ပိုျမင္႔ပါတယ္။

ကုဒ္ေတြကို run ၾကည္႔ရင္ ေအာက္ပါအတိုင္း ေတြ႔ရမွာပါ-

Your location is 7, 10

Enter direction (n, s, e, w): s

You found the treasure!

C++ မွာ Logical Operator ၃ ခု ရွိပါတယ္-

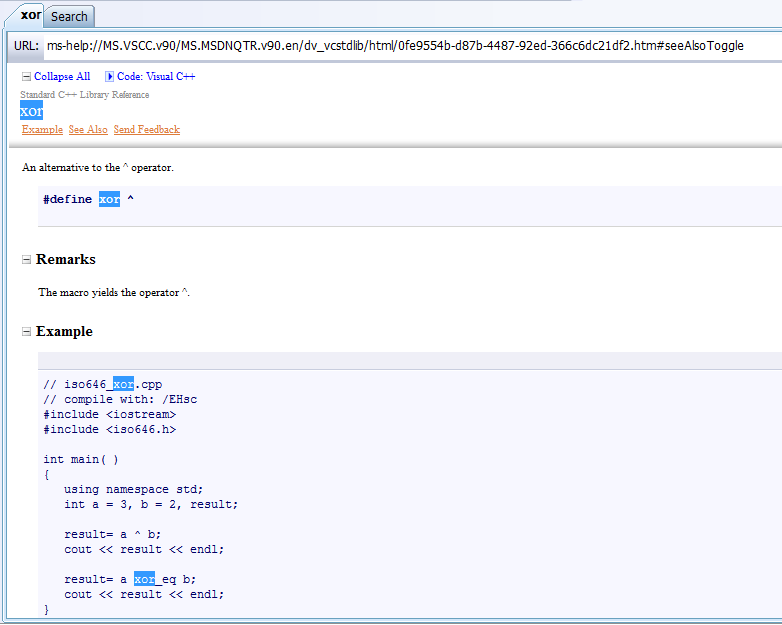
***Operator Effect***

&& Logical AND

|| Logical OR

! Logical NOT

Lafore ရဲ႕ စာအုပ္[1]၊ စာမ်က္ႏွာ ၁၁၆ မွာ ေျပာထားတာက C++ မွာ XOR (exclusive OR) logical operator မရွိဘူးလို႔ ဆိုပါတယ္။ ကၽြန္ေတာ္ကေတာ႔ C မွာတုန္းက သံုးခဲ႔တဲ႔ ^ ကို logical operator အေနနဲ႔ သံုးတာ အဆင္ေျပပါတယ္။ Visual Studio 2008 MSDN မွာေတာ႔ ေအာက္ပါအတိုင္း ႏွစ္မ်ိဳး သံုးျပထားပါတယ္။



ေနာက္အပတ္ေတြက်မွ Logical operators သံုးမ်ိဳးအေၾကာင္းကို ဆက္လက္ ေဆြးေႏြးသြားပါမယ္ခင္ဗ်ာ။ ဤအခန္းဆက္ ေဆာင္းပါးမ်ားဟာ IT ေက်ာင္းသားမ်ားအတြက္ အထူးရည္ရြယ္ၿပီး ေရးသားထားတာ ျဖစ္ပါတယ္။ သင္ရိုးကို ေရးသားျခင္း ျဖစ္တဲ႔အတြက္ တိုက္ရိုက္ ဘာသာျပန္ဆိုျခင္းနည္းကို ေနရာေတာ္ေတာ္မ်ားမ်ားမွာ သံုးထားပါတယ္။ သိလိုသည္မ်ားကို 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)

aungwh2013@gmail.com

http://www.tech4mm.com

ေတာ္ဝင္ ကြန္ပ်ဴတာ စင္တာ

၁၇၉ စ၊ သုမဂၤလာ၊ ေစ်းေလး အေနာက္ဘက္၊ ျပင္ဦးလြင္ၿမိဳ႕