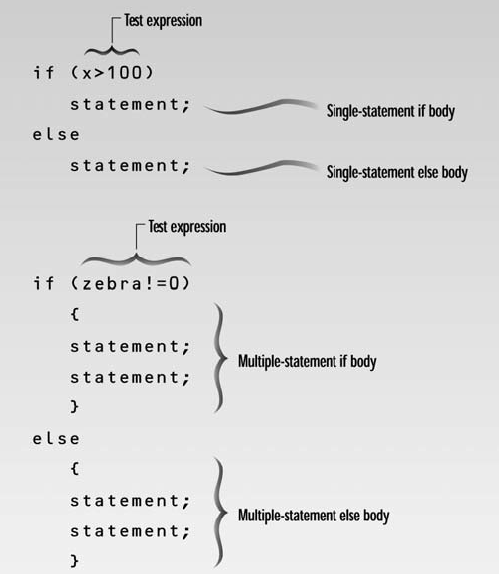
**OOP in C++ : အပိုင္း (၁၂) – Decision by If…Else**

ဒီအပတ္ သင္ခန္းစာမွာေတာ႔ **if…else Statement** အေၾကာင္းေလးေတြကို ဆက္လက္ ေဆြးေႏြးသြားမွာ ျဖစ္ပါတယ္။

**The if...else Statement**

**if** statement ဟာ အေျခအေနတစ္ခု မွန္ကန္ရင္ ဘာလုပ္မယ္ဆိုတာကို ေရးသားဖို႔အတြက္ အသံုးျပဳရတာ ျဖစ္ပါတယ္။ ဒါေပမယ္႔ မွားသြားခဲ႔ရင္ေတာ႔ ဘာတစ္ခုမွ လုပ္ေပးမွာ မဟုတ္ပါဘူး။ အခ်ိဳ႕အေျခအေနေတြမွာ မွန္ရင္ တစ္ခုခုလုပ္ၿပီး မွားသြားရင္ေတာ႔ တျခားတစ္ခု လုပ္ေဆာင္ဖို႔ လိုအပ္တာတတ္ပါတယ္။ အဲဒီလို အေျခအေနမ်ိဳးအတြက္ေတာ႔ if…else statement ကို အသံုးျပဳရမွာ ျဖစ္ပါတယ္။ ေအာက္မွာျပထားသလိုပါပဲ if statement ေနာက္မွာ statement တစ္ခု ဒါမွမဟုတ္ တြန္႔ကြင္းနဲ႔ခတ္ထားတဲ႔ multiple statements ေတြ ရွိပါတယ္။ အဲဒီေနာက္မွာ else ကို ေရးၿပီး သူ႔ေနာက္ကလဲ statement တစ္ခု ဒါမွမဟုတ္ တြန္႔ကြင္းနဲ႔ခတ္ထားတဲ႔ multiple statements ေတြ ထည္႔ေရးေပးရမွာ ျဖစ္ပါတယ္။ if ေနာက္က test expression မွန္ခဲ႔မယ္ ဆိုရင္ အဲဒီေနာက္က statement(s) ေတြ အလုပ္လုပ္မွာ ျဖစ္ပါတယ္။ မွားခဲ႔ရင္ေတာ႔ else ေနာက္က statement(s) ေတြ အလုပ္လုပ္မွာ ျဖစ္ပါတယ္။



အထက္က if example ေလးကို if…else နဲ႔ ျပန္ေရးျပထားပါတယ္။

// ifelse.cpp

// demonstrates IF...ELSE statememt

#include <iostream>

using namespace std;

int main()

{

int x;

cout << “\nEnter a number: “;

cin >> x;

if( x > 100 )

cout << “That number is greater than 100\n”;

else

cout << “That number is not greater than 100\n”;

return 0;

}

ဒီဥပမာေလးမွာ user ထည္႔သြင္းေပးလိုက္တဲ႔ x တန္ဖိုးဟာ 100 ထက္ႀကီးခဲ႔ရင္ cout<< “That number is greater than 100\n”; ဆိုတာကို လုပ္ေဆာင္မွာ ျဖစ္ၿပီး မွားခဲ႔ရင္ေတာ႔ cout << “That number is not greater than 100\n”; ဆိုတာကို လုပ္ေဆာင္ေပးမွာပါ။ နမူနာအေနနဲ႔ 300 နဲ႔ 3 တန္ဖိုးႏွစ္ခု ထည္႔သြင္းေပးလိုက္လို႔ ရလာတဲ႔ ရလာဒ္ေတြကို ေအာက္မွာ ေလ႔လာ ၾကည္႔ႏိုင္ပါတယ္။

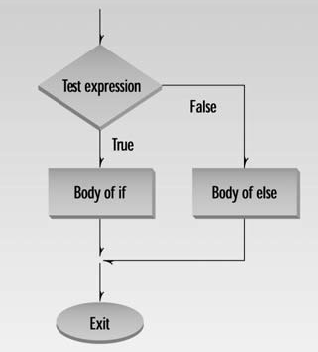
Enter a number: 300

That number is greater than 100

Enter a number: 3

That number is not greater than 100

if…else အလုပ္လုပ္ပံုကို flow chat နဲ႔ ေအာက္ပါအတိုင္း ေလ႔လာႏိုင္ပါတယ္။



**The getche() Library Function**

ေနာက္ထပ္ဥပမာ တစ္ခုအေနနဲ႔ if…else ကို while loop ထဲမွာ ထည္႔သြင္း အသံုးျပဳျပထားပါတယ္။ ေနာက္တစ္ခုက getche() ဆိုတဲ႔ library function အသစ္တစ္ခုကို သံုးျပထားပါတယ္။ အဲဒီ chcount.cpp ပရိုဂရမ္ေလးဟာ user ထည္႔သြင္းေပးလိုက္တဲ႔ စာလံုးေတြနဲ႔ အကၡရာ အေရအတြက္ေတြကို ေရတြက္ေပးမွာပါ။

// chcount.cpp

// counts characters and words typed in

#include <iostream>

using namespace std;

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

int main()

{

int chcount=0; //counts non-space characters

int wdcount=1; //counts spaces between words

char ch = ‘a’; //ensure it isn’t ‘\r’

cout << “Enter a phrase: “;

while( ch != ‘\r’ ) //loop until Enter typed

{

ch = getche(); //read one character

if( ch==’ ‘ ) //if it’s a space

wdcount++; //count a word

else //otherwise,

chcount++; //count a character

} //display results

cout << “\nWords=” << wdcount << endl

<< “Letters=” << (chcount-1) << endl;

return 0;

}

input ထည္႔သြင္းဖို႔အတြက္ cin နဲ႔ >> ကို အသံုးျပဳေလ႔ ရွိပါတယ္။ ဒီနည္းကို သံုးမယ္ဆိုရင္ user က input တစ္ခု ထည္႔သြင္းေပးလိုက္တိုင္း Enter ေခါက္ရမွာျဖစ္ပါတယ္။ ဒါေပမယ္႔ အခုလို ဥပမာ မ်ိဳးမွာ user က Enter မႏွိပ္ရဘဲ အကၡရာတိုင္းကို ဖတ္သြားဖုိ႔ လိုလာပါၿပီ။ ဒါေၾကာင္႔ getche() ဆိုတဲ႔ library function ကို အသံုးျပဳရမွာ ျဖစ္ပါတယ္။ ၄င္းကို အသံုးျပဳဖို႔ conio.h header file ကို ေၾကျငာေပးရမွာ ျဖစ္ပါတယ္။ input argument ထည္႔ေပးဖို႔ မလိုအပ္ဘဲ user ရိုက္ထည္႔လိုက္သမွ် အကၡရာေတြကို ခ်က္ခ်င္း return ျပန္ေပးမွာ ျဖစ္ပါတယ္။ chcount.cpp ထဲမွာေတာ႔ getche() နဲ႔ ဖတ္လို႔ ရလာတဲ႔ တန္ဖိုးေတြကို ch ထဲ ထည္႔ေပးလိုက္မွာျဖစ္ပါတယ္။ ေနာက္တစ္ခုက getche() function ဟာ user ရိုက္ထည္႔လိုက္သမွ် အကၡရာေတြကို ကြန္ပ်ဴတာမ်က္ႏွာျပင္ေပၚကို echo လုပ္ေပးမွာ ျဖစ္ပါတယ္။ (\_getch() function ကေတာ႔ echo ျပန္လုပ္ေပးမွာ မဟုတ္ပါဘူး။ echo ျပန္လုပ္ေပးလို႔လဲ function name ရဲ႕ ေနာက္ဆံုး စကားလံုးမွာ e ထည္႔ထားၿပီး getche ျဖစ္ေနတာပါ။ if…else statement ကေတာ႔ စကားလံုး(word) ေရတြက္ဖို႔ သံုးထားတဲ႔ wdcount ကို space ဆိုတဲ႔ character ေတြ႔တာနဲ႔ တစ္တိုးေပးမွာ ျဖစ္ပါတယ္။ အကၡရာ (character) ေတြ ေရတြက္ဖို႔ သံုးတဲ႔ chcount ကေတာ႔ space ကလြဲခို႔ ဘယ္အကၡရာကိုပဲ ရိုက္ရိုက္ တစ္တိုးေပးမွာ ျဖစ္ပါတယ္။ ဆိုလိုတာက space ကလြဲလို႔ က်န္တဲ႔ အကၡရာေတြကို character လို႔ သတ္မွတ္လိုက္တာပါ။ (ဒီလို စစ္ေဆးတဲ႔ နည္းမွာ အားနည္းခ်က္ေတြ ရွိပါတယ္။ ဒါေပမယ္႔ ဒီေနရာမွာေတာ႔ ဒီဥပမာေလးဟာ လံုေလာက္တယ္လို႔ ယူဆပါတယ္)။ ေအာက္မွာ နမူနာထည္႔သြင္းေပးလိုက္တဲ႔ စာေၾကာင္းေလးကို စကားလံုးနဲ႔ အကၡရာေတြ ေရတြက္ပံု ေလ႔လာၾကည္႔ၾကရေအာင္။

For while and do

Words=4

Letters=13

while statement မွာပါတဲ႔ test expression ကေတာ႔ ch ရဲ႕တန္ဖိုးဟာ ‘\r’ ျဖစ္မျဖစ္ စစ္ေဆးပါတယ္။ တကယ္ေတာ႔ အဲဒီ အကၡရာဟာ ကီးဘုတ္က Enter key ကို ကိုယ္စားျပဳထားတာပါ။ တကယ္လို႔ Enter ေခါက္ခဲ႔ရင္ while loop ကေန ထြက္သြားမွာ ျဖစ္ပါတယ္။

**Assignment Expressions**

chcount.cpp ကို assignment expressions နဲ႔ precedence ေတြရဲ႕ အေရးပါပံုနဲ႔ ကုဒ္လိုင္းအခ်ိဳ႕ကို ခ်ံဳ႕ပစ္ႏိုင္တာကိုျပသဖို႔ အနည္းငယ္ ျပင္ေရးျပခ်င္ပါတယ္။ အဲဒီလို ျပင္ေရးလိုက္ျခင္းအားျဖင္႔ ကုဒ္ေတြကို ဖတ္ရတာ တစ္မ်ိဳး ျဖစ္ေနႏိုင္ေပမယ္႔ C နဲ႔ C++ မွာေတာ႔ ေရးရိုးေရးစဥ္ ေရးနည္းတစ္ခုပဲ ျဖစ္ပါတယ္။ ျပင္ေရးထားတဲ႔ chcnt2.cpp ကိုေလ႔လာၾကည္႔ၾကရေအာင္။

// chcnt2.cpp

// counts characters and words typed in

#include <iostream>

using namespace std;

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

int main()

{

int chcount=0;

int wdcount=1; // space between two words

char ch;

while( (ch=getche()) != ‘\r’ ) // loop until Enter typed

{

if( ch==’ ‘ ) // if it’s a space

wdcount++; // count a word

else // otherwise,

chcount++; // count a character

} // display results

cout << “\nWords=” << wdcount << endl

<< “Letters=” << chcount << endl;

return 0;

}

getche() ကေန return ျပန္ေပးလိက္တဲ႔ တန္ဖိုးကို ch ထဲ ထည္႔ေပးတဲ႔ ကုဒ္ကို အျပင္မွာ သပ္သပ္ မေရးသားေတာ႔ဘဲ while loop ရဲ႕ test expression ထဲမွာ တစ္ခါတည္း ထည္႔သြင္း ေရးသားလိုက္ျခင္းပဲ ျဖစ္ပါတယ္။ ေနာက္ထပ္ၿပီး အဲဒီ တန္ဖိုးကို ‘\r’ ဟုတ္မဟုတ္ စစ္ေဆးေပးပါတယ္။

တကယ္ေတာ႔ (ch=getche()) ဆိုတဲ႔ ကုဒ္ တစ္ခုလံုးက ch ထဲမွာ ရွိတဲ႔ တန္ဖိုးကို ကိုယ္စားျပဳေနတာ ျဖစ္တဲ႔အတြက္ တစ္ခါတည္း မွန္မမွန္ စစ္ေဆးလို႔ ရသြားတာပဲ ျဖစ္ပါတယ္။ C++ မွာ x = y = z = 0; ဆိုၿပီး ေရးသားႏိုင္ပါတယ္။ အဲဒီလို ေရးနည္းမွာ ညာဘက္အစြန္ဆံုး assignment က အရင္ စအလုပ္လုပ္္ၿပီး 0 ကို z ထဲ၊ z တန္ဖိုးကို y ထဲ၊ y တန္ဖိုးကို x ထဲထည္႔မွာ ျဖစ္ပါတယ္။ ဒါေၾကာင္႔ တန္ဖိုးေတြအားလံုးက 0 ျဖစ္သြားပါလိမ္႔မယ္။ (ch=getche()) ဆိုတဲ႔ assignment expression ကို လက္သဲကြင္းနဲ႔ ခတ္ထားတာကို သတိထားမိပါလိမ္႔မယ္။ ဘာလို႔လဲဆိုေတာ႔ assignment operator ဟာ relational operator ထက္ precedence နိမ္႔တဲ႔အတြက္ အရင္အလုပ္လုပ္ေစခ်င္တဲ႔အခါ ကြင္းခတ္ေပးလိုက္ရတာပဲ ျဖစ္ပါတယ္။ တကယ္လို႔ လက္သဲကြင္းသာ မပါခဲ႔ဘူးဆိုရင္ while( ch = (getche() != ‘\r’) ) လို လုပ္ေဆာင္သြားမွာ ျဖစ္ၿပီး ကၽြန္ေတာ္တို႔ မေမွ်ာ္လင္႔ထားတဲ႔ အမွားေတြကို ရင္ဆိုင္ရမွာပါ။

အဲဒီ chcnt2.cpp ထဲမွာပါတဲ႔ while statement ဟာ ေနရာအနည္းငယ္ပဲ ယူေပမယ္႔ အသံုးတည္႔ပါတယ္။ တကယ္ေတာ႔ ၄င္းဟာ test expression တစ္ခုသာမကဘဲ keyboard က ရိုက္ထည္႔လိုက္တဲ႔ အကၡရာ တစ္လံုးကို ch ထဲကိုလည္း ထည္႔ေပးႏိုင္ခဲ႔ပါတယ္။ ဒါေပမယ္႔ ဒီလိုကုဒ္မ်ိဳးကို ပထမဆံုး ျမင္ဘူးကာစမွာေတာ႔ အဓိပၸါယ္ေဖာ္ဖို႔ ခက္ခဲႏိုင္ပါတယ္။

ေနာက္အပတ္ေတြမွာေတာ႔ က်န္ရွိေနေသးတဲ႔ **Nested if...else Statements** အေၾကာင္းကို 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)

http://www.tech4mm.com

ေတာ္ဝင္ ကြန္ပ်ဴတာ စင္တာ

၁၇၉ စ၊ သုမဂၤလာ၊ ေစ်းေလး အေနာက္ဘက္၊ ျပင္ဦးလြင္ၿမိဳ႕