**OOP in C++ : အပိုင္း (၂၀) - Structures (၃)**

**Structures Within Structures**

structures ေတြထဲမွာ ေနာက္ထပ္ structures ေတြ nested လုပ္ၿပီး ထည္႔ႏိုင္ပါတယ္။ နမူနာအေနနဲ႔ englstrc.cpp ကို ျပင္ေရးျပပါ႔မယ္။ ဒီ ဥပမာထဲမွာ အခန္းတစ္ခုရဲ႕ အလ်ားနဲ႔ အနံ အရြယ္အစားေတြကို သိမ္းထားမယ္႔ data structure တစ္ခုကို ဖန္တီးျပပါ႔မယ္။ အဂၤလိပ္ အတိုင္းအတာေတြကို သံုးထားတာျဖစ္လို႔ ေပ နဲ႔ လက္မကိုုပဲ ဆက္သံုးသြားမွာပါ။ ေအာက္မွာ Room ဆိုတဲ႔ structre တစ္ခုကို Distance type variable lenght နဲ႔ width ကို အသံုးျပဳၿပီး တည္ေဆာက္ျပထားပါတယ္။

struct Room

{

Distance length;

Distance width;

}

ေအာက္က englarea.cpp မွာေတာ႔ အခန္းတစ္ခန္းကို ကိုယ္စားျပဳဖို႔ Room structure ကို သံုးထားပါတယ္။

// englarea.cpp

// demonstrates nested structures

#include <iostream>

using namespace std;

////////////////////////////////////////////////////////////////

struct Distance //English distance

{

int feet;

float inches;

};

////////////////////////////////////////////////////////////////

struct Room //rectangular area

{

Distance length; //length of rectangle

Distance width; //width of rectangle

};

////////////////////////////////////////////////////////////////

int main()

{

Room dining; //define a room

dining.length.feet = 13; //assign values to room

dining.length.inches = 6.5;

dining.width.feet = 10;

dining.width.inches = 0.0;

//convert length & width

float l = dining.length.feet + dining.length.inches/12;

float w = dining.width.feet + dining.width.inches/12;

//find area and display it

cout << “Dining room area is “ << l \* w

<< “ square feet\n” ;

return 0;

}

ဒီပရိုဂရမ္ထဲမွာ Room type - dining ဆိုတဲ႔ variable တစ္ခုကို ေအာက္ပါအတိုင္း define ျပဳလုပ္ထားပါတယ္။

Room dining; // variable dining of type Room

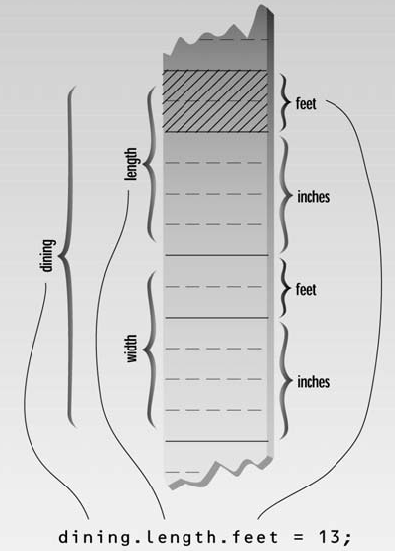
ေနာက္ပိုင္းမွာေတာ႔ အဲဒီ structure ရဲ႕ members ေတြကို တန္ဖိုးေတြ assign ျပဳလုပ္ေပးပါတယ္။

**Accessing Nested Structure Members**

Structure တစ္ခုထဲမွာ အျခား structure တစ္ခုကို ျပဳလုပ္ထားတာေၾကာင္႔ သူတို႔ရဲ႕ members ေတြကို ရယူဖို႔ dot operator ႏွစ္ခုကို အသံုးျပဳရမွာပါ။ ဥပမာ-

dining.length.feet = 13;

အဲဒီ statement မွာဆိုရင္ dining က structure variable ရဲ႕ name ျဖစ္ပါတယ္။ length က အျပင္က structure (Room)ရဲ႕ member name ျဖစ္ၿပီး feet ကေတာ႔ အတြင္းက structure (Distance) ရဲ႕ member name ပါ။ ကုဒ္ရဲ႕ ဆိုလိုခ်က္ အျပည္႔အစံုက dining variable ရဲ႕ length member ထဲက feet member ထဲကို တန္ဖိုး ၁၃ ထည္႔ေပးလိုက္တာပဲ ျဖစ္ပါတယ္။ ေအာက္က ပံု (၂၀-၁) မွာ ေလ႔လာၾကည္႔ပါ။



ပံု(၂၀-၁) nested structures မ်ားအတြင္း dot operator အသံုးျပဳပံု

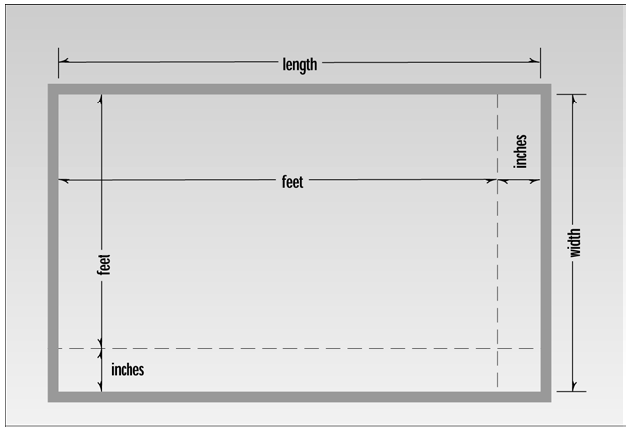
dining ရဲ႕ members ေတြကို တန္ဖိုးေတြ assign လုပ္ေပးၿပီးတာနဲ႔ ပရိုဂရမ္က ပံု(၂၀-၂) မွာ ျပထားသလို အဲဒီအခန္းရဲ႕ ၾကမ္းခင္း ဧရိယာကို တြက္ေပးမွာ ျဖစ္ပါတယ္။ ဧရိယာကို မတြက္ခင္ ပထမဆံုး Distance type ေတြျဖစ္တဲ႔ length နဲ႔ width ေတြကို feet နဲ႔ တိုင္းတာတဲ႔ float variables ေတြ ျဖစ္တဲ႔ l နဲ႔ w အျဖစ္ ေျပာင္းလဲေပးမွာ ျဖစ္ပါတယ္။ အဲဒီလို ေျပာင္းလဲဖို႔အတြက္ inches member တန္ဖိုးေတြကို 12 နဲ႔ စားၿပီး feet member ေတြထဲ ေပါင္းထည္႔ေပးရမွာပါ။

//convert length & width

float l = dining.length.feet + dining.length.inches/12;

float w = dining.width.feet + dining.width.inches/12;

အဲဒီလို ေပါင္းၿပီးသြားတာနဲ႔ ရလာတဲ႔ ရလဒ္ေတြဟာ float type ျဖစ္သြားပါတယ္။ ဧရိယတန္ဖိုး ရွာဖို႔အတြက္ကေတာ႔ l နဲ႔ w ကို ေျမွာက္ေပးလိုက္ရံုပဲ ျဖစ္ပါတယ္။



ပံု(၂၀-၂) အခန္းတစ္ခု၏ အတိုင္းအတာမ်ားျပပံု

**User-Defined Type Conversions**

မွတ္သားရမယ္႔ အခ်က္တစ္ခုက ပရိုဂရမ္ဟာ Distance type အတိုင္းအတာ ႏွစ္ခုကို float type variable l နဲ႔ w အျဖစ္ ေျပာင္းလဲလိုက္တာပဲ ျဖစ္ပါတယ္။ ဒါေၾကာင္႔ Room type structure ထဲမွာ သိမ္းဆည္းထားတဲ႔ (ဒီဥပမာမွာ ေနာက္ထပ္ Distance type structure ႏွစ္ခုနဲ႔ သိမ္းထားတဲ႔) အတိုင္းအတာေတြကို square feet နဲ႔ ကိုယ္စားျပဳတဲ႔ ၾကမ္းခင္းဧရိယာအျဖစ္ floating-point number တစ္ခုတည္းဘဝကို ေျပာင္းလဲေပးခဲ႔ပါတယ္။ ပရိုဂရမ္ရဲ႕ ရလဒ္ေတြကေတာ႔ ေအာက္ပါအတိုင္းပဲ ျဖစ္ပါတယ္ -

Dining room area is 135.416672 square feet

User-defined data types ေတြ အသံုးျပဳထားတဲ႔ ပရိုဂရမ္ေတြထဲမွာ data type ေျပာင္းလဲေပးျခင္းဟာ အေရးႀကီးတဲ႔ အခ်က္တစ္ခု ျဖစ္ပါတယ္။

**Initializing Nested Structures**

အခုလိုမ်ိဳး structure တစ္ခုကို ငံုထားတဲ႔ structure တစ္ခုထဲကို initialize လုပ္ၿပီး တန္ဖိုးေတြကို ဘယ္လို ထည္႔သြင္းၾကမွာလဲ? ေအာက္ပါ statement ေလးကေတာ႔ englarea.cpp ထဲက တန္ဖိုးေတြကို တစ္ခါတည္း initialize လုပ္ၿပီး ထည္႔သြင္းတဲ႔ နည္းလမ္း ျဖစ္ပါတယ္။

Room dining = { {13, 6.5}, {10, 0.0} };

Room structure ထဲမွာ ျမွပ္ႏွံထားတဲ႔ Distance type structure တစ္ခုစီကို သီးျခား initialized လုပ္ေပးဖို႔ လိုပါတယ္။ အဲဒီလို သီးျခားျပဳလုပ္ဖို႔အတြက္ တြန္႔ကြင္းေတြနဲ႔ ေကာ္မာေတြကို အသံုးျပဳရမွာပဲ ျဖစ္ပါတယ္။

ပထမ Distance ကို {13, 6.5} နဲ႔ ဒုတိယ တစ္ခုကိုေတာ႔ {10, 0.0} initialized လုပ္ပါတယ္။ အဲဒီေနာက္ ရလာတဲ႔ Distance value ႏွစ္ခုကို Room variable ထဲကို ထပ္မံ initialize လုပ္ေပးပါတယ္။ တြန္႔ကြင္းနဲ႔ ေကာ္မာေတြပဲ သံုးတာ ျဖစ္ပါတယ္။

**Depth of Nesting**

သီအိုရီအရ structures ေတြထဲမွာ အျခား structures ေတြ အဆင္႔ဆင္႔ အကန္႔အသတ္မရွိ nested လုပ္လို႔ ရပါတယ္။ အေဆာက္အဦေတြကို ဒီဇိုင္းျပလုပ္တဲ႔ ပရိုဂရမ္မ်ိဳးမွာ ေအာက္ပါ ကုဒ္မ်ိဳးေတြကို ျမင္ေတြ႔ရႏိုင္ပါတယ္။

apartment1.laundry\_room.washing\_machine.width.feet

ေနာက္အပတ္ေတြမွာေတာ႔ Card Game ေလးတစ္ခုကို structure အသံုးျပဳၿပီး ေရးသားနည္းမ်ား 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://engineer4myanmar.blogspot.com

ေတာ္ဝင္ ကြန္ပ်ဴတာ စင္တာ

၁၇၉ စ၊ သုမဂၤလာ၊ ေစ်းေလး အေနာက္ဘက္၊ ျပင္ဦးလြင္ၿမိဳ႕