

Microsoft
Windows®

```
1: // This program asks the user for information about a person
2: // class history, and then writes this object to HISTORY.BAT
3: #include <iostream.h>
4:
5: class History
6{
7: protected:
8: char name[20];
9: char degree[3];
10: message;
11: public:
12: void getData(void);
13: {
14: cout << "Enter name : "; cin >> name;
15: cout << "Enter age : "; cin >> age;
16: cout << "Enter degree : "; cin >> degree;
17: }
18: };
19:
20: main( )
21: {
22: history person;           // Create a history
23: person.getData();          // Get data for history
24:
25: ofstream outfile("HISTORY.BAT");
26: outfile.write((char *) &person, sizeof(person));
27: }
```

C



Complete

AUNG MYINT (M.E., AUSTRALIA)

CHAPTER 1

C++ BASICS

1.1.	A Brief History of C and C++	10
1.2.	My First C++ Program	12
1.3.	Identifiers	15
1.4.	Keywords	16
1.5.	Variables	17
1.6.	Constants	24
1.7.	Expressions	30
1.8.	Assignments	31



Complete

CHAPTER 2

FUNCTIONS

2.1.	Create a Simple Function	46
2.2.	Pass the Variables by Value	50
2.3.	Pass Variables by Reference	59
2.4.	Default Function Arguments	63
2.5.	Inline Functions	65
2.6.	Local and Global Variables	65
2.7.	Overloaded Functions	69
2.8.	Recursion	73
2.9.	Passing Structure Variables	77



Complete

CHAPTER 3

PROGRAM FLOW CONTROL

3.1.	The if Statement	82
3.2.	The goto Statement	86
3.3.	The if-else Statement	92
3.4.	Nested if Statement	96
3.5.	The while Statement	101
3.6.	The do-while Statement	108
3.7.	The for Statement	114
3.8.	Nested Loops	120
3.9.	The switch Statement	123
3.10.	The break Statement	127
3.11.	The continue Statement	129



Complete

CHAPTER 4

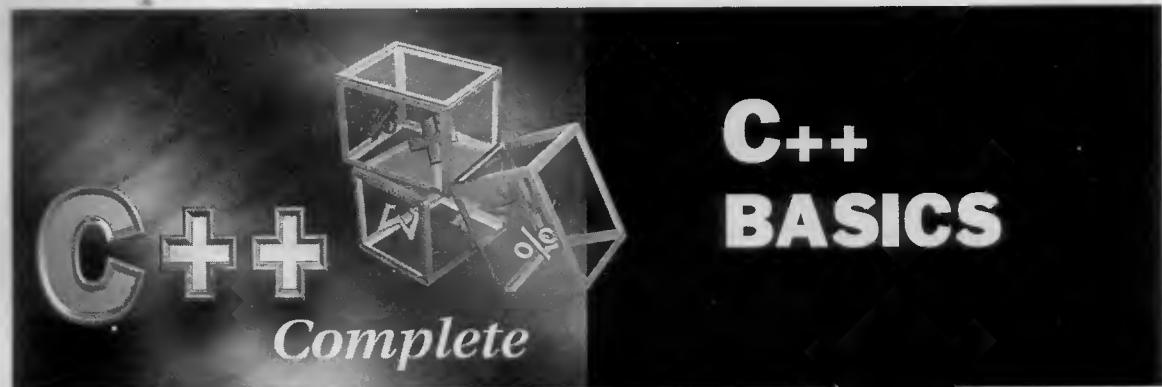
POINTERS AND ADDRESSES

4.1.	Referencing Variables with Pointers	133
4.2.	Pointers and Arrays	136
4.3.	Pointers to Structures	140
4.4.	Pointers as Function Arguments	142
4.5.	Passing Arrays by Pointers	145
4.6.	Pointers and Strings	151
4.7.	Arrays of Pointers to Strings	156
4.8.	Pointers to Pointers	158
4.9.	Memory Allocation	164
4.10.	Using Reference Variables	168



Complete

Chapter 1



C++ BASICS

1979 ခုနှစ် နှစ်ဦးပိုင်းလောက်က အမေရိကန်ပြည်ထောင်စု၊ New Jersey ပြည်နယ်၊ AT & T Bell Laboratories မှာလုပ်ကိုင်နေတဲ့ Bjarne Stroustrup ဆိုသူကွန်ပျူတာပရိုဂရမ်မာတစ်ဦးက C language ကိုအခြေခံပြီး C++ ဆိုတဲ့ object-oriented programming language တစ်ခုကိုရေးသားခဲ့ပါတယ်။ အစက C++ မဟုတ်ပါဘူး။ C with Classes လိုအမည်ပေးခဲ့ပါတယ်။ 1983 ခုနှစ်ရောက်မှာ C++ လိုနာမည်ပြောင်းလိုက် တာပါ။ C++ က C ကိုအဆင့်မြှင့်ထားတဲ့ version ပါပဲ။ C++ source code တွေကို C compiler တွေပေါ်မှာ run နိုင်အောင် Cfront ဆိုတဲ့ interpreter ကိုအသုံးပြုခဲ့တယ်လေ။ တစ်ချို့က C++ ကို C ခဲ့ superset လိုပြောကြပါတယ်။ C language မှာ လုပ်လိုရတာအကုန်လုံးလောက်ကို C++ မှာလည်း လုပ်လိုရတာကိုး။ တစ်ခုတော့ ရှိတယ်။ C++ language မှာအသစ်ထပ်ဖြည့်ထားတဲ့ keyword တွေက C language မှာ မပါဘူးဆိုတော့ အဲဒီ keyword တွေကို C program မှာ identifier တွေအနေနဲ့ အသုံးပြုလိုရတယ်ပေါ့။ အဲဒီလို C program မျိုးကို C++ က compile လုပ်လိုမရပါဘူး။ C compiler ကပဲ compile လုပ်ပြီး run ပေးနိုင်မှာပါပဲ။ ဘာပဲဖြစ်ဖြစ်တဖတ်သူအနေနဲ့ C language ကိုသိနေပြီးသားဆိုရင် C++ ကိုဆက်လက်လေ့လာတဲ့အခါ လွယ်ကူချောမောမှာပါ။

၁.၁ A Brief History of C and C++

C language ကို Dennis Ritchies ဆိုသူ ကွန်ပျော်တာပရီဂရမ်မှတစ်ညီးက AT & T Bell Lab မှာ လုပ်ကိုင်နေရင် 1970 ခုနှစ်မှာ interpreted programming language BCPL ကို မြို့ပြီးရေးသားခဲ့တာဖြစ်ပါတယ်။ ရည်ရွယ်ချက်က C နဲ့ low-level programming ကို Assembly language လိုပဲ လုပ်နိုင်ဖိုပါပဲ။ ဆက်သွယ်နည်းက လူတိုင်းအသုံးပြုလို လွယ်ကူတဲ့ high-level language နဲ့ဆက်သွယ်တာမျိုးပေါ့။ Ritchie ဟာ သူလက်ရှိအသုံးပြုနေတဲ့ UNIX system PDP-11 စက်မှာပဲ C ကိုလက်တွေ့စင်းကြည့်တော့ အောင်မြင်သွားပါတယ်။ ဒီအခါမှာ UNIX developer ဖြစ်တဲ့ Ken Thompson ကလည်း UNIX system ကို C နဲ့ ပြောင်းလဲရေးသားတယ်လေ။ C ဟာ UNIX system မှာ နှစ်အကြောက်းတည်ရှိနေခဲ့ပါတယ်။

တစ်ချိန်မှာ AT & T ကနေ UNIX system ကို တဗ္ဗာသိုလ်တွေမှာအသုံးပြုဖို့အတွက် အလကားလောက်နီးနီးနဲ့လျှော့တော့ C language ဟာ တဗ္ဗာသိုလ် ကျောင်းသားတွေနဲ့ အနီးစပ်ဆုံး language တစ်ခုဖြစ်သွားခဲ့ပါတယ်။ ယင်းအကျိုးဆက်က တဗ္ဗာသိုလ်ကျောင်းသားတွေ ကျောင်းကထွက်ပြီးအလုပ်ပင်သွားတဲ့အချိန်မှာတော့ high level language မှာကြီးစိုးနေတဲ့ COBOL နဲ့ low level language မှာကြီးစိုးနေတဲ့ Assembly language တိုကို သူတို့တတ်သိလာတဲ့ C နဲ့ တိုက်ထုတ်လိုက်ပြီးပေါ့။ အဲဒီအချိန်က microcomputer တွေမှာ CP/M operating system ကလွှမ်းမြို့နေဆဲပါ။ high-level language အနေနဲ့အသုံးများနေတာက BASIC language ဖြစ်ပါတယ်။

1979 ခုနှစ်မှာ Brian Kernighan နဲ့ Dennis Ritchie တို့ပူးတဲ့ရေးသားတဲ့ The C Programming Language စာအုပ်ကို Prentice Hall စာအုပ်တိုက်ကနေထုတ်ဝေခဲ့ပါတယ်။ ဒါပေမယ့် အဲဒီစာအုပ်မှာပါဝင်တဲ့ C language specification တွေကို standard လို့ မသတ်မှတ်နိုင်သေးပါဘူး။ ခေတ်စကားနဲ့ပြောရရင် "Classic C" လိုပဲခေါ်ရမှာပေါ့။ တစ်ချို့ကလည်းဒီစာအုပ်ကို စာရေးသူတွေရဲ့နာမည်အဖျားခွွတ်ထည့်ပြီး K&R C လို့ ခေါ်တာလည်းရှိရဲ့။ တိုက်တိုက်ဆိုင်ဆိုင် IBM ကလည်း main frame တွေထုတ်လုပ်နေရာမှ 1981 ခုနှစ်မှာ PC လောကထဲကိုဝင်လာတယ်လေ။ IBM ကိုတူတဲ့ IBM clone တွေကလည်းပေါ့မှုပေါ့ပေါ့။ Microsoft ကိုယ်တိုင်က သူ့ရဲ့ပထမဆုံးအောင်မြင်လာတဲ့ Windows 3.X operating system ကို C language နဲ့ပဲရေးလိုက်တဲ့အခါမှာ လူအများက C ကို လွှတ်လွှတ်ကျွတ်ကျွတ် လက်ခံသွားတာမဆန်းတော့ပါဘူး။ ဒီအခြေအနေတွေဟာ C language အတွက် ရေကန်အသင့်ကြာအသင့်ဖြစ်သွားပေါ့တယ်။

1983 ခုနှစ်မှာ ANSI က C language definition ကို အရင်သတ်မှတ်ပြီး ISO (International Standards Organization) ကိုဆက်လက်တင်ပြေခဲ့ရာမှာ 1990 ခုနှစ်ကျော် C standard document ကိုရှိက်နှုပ်ဖြန့်ချိနိုင်ခဲ့ပါတယ်။ အဲဒီစာအုပ်ဟာ Standard C ပါပဲ။ international standard C++ document ကို 1998 ခုနှစ်မှာအပြီးသတ်ထုတ်ဝေနိုင်ခဲ့ပါတယ်။ စာဖတ်သူအနေနဲ့ C ၊ C++ တို့ရဲ့တိုးတက်ဖြစ်ပေါ့မှု timeline ကို အလွယ်တကူ လေ့လာလို့ရအောင် ယေား (၀.၁) မှာဖော်ပြထားပါတယ်။ လေ့လာကြည့်ပါ။

အသာ: (၁.၁) **C and C++ Timeline**

Year	Event
1970	UNIX system ကို minicomputer တွေမှာအသုံးပြုနိုင်အောင် Assembly language နဲ့ Ken Thompson ကရေးသားပါတယ်။
1972	C programming language ကို Dennis Ritchie က စတင်ဒီဇိုင်းလုပ်ပြီး Thompson က UNIX system ကို C နဲ့ အသစ်ပြင်ရေးပါတယ်။
1978	Kernighan and Ritchie တို့ နှစ်ဦးစပ်တူရေးသားတဲ့ C Programming Language စာအုပ်ကို Prentice Hall ကရိုက်နှိပ်ဖြန့်ချုပါတယ်။
1979	C language ကို အကြော်ပြီး Bjarne Stroustrup က C with Classes (1st version) ကို အဆင့်မြှင့်ရေးသားပါတယ်။
1980	CP/M 8080 microcomputer operating system တွေမှာ အသုံးပြုနိုင်တဲ့ C compiler တွေ စတင်ထွက်ရှိလာပါတယ်။
1983	C with Classes နာမည်ကို C++ လို့ အမည်ပြောင်းလဲပါတယ်။ C ကို standardize လုပ်ဖို့အတွက် ANSI ကနေစတင်စည်းဝေးပါတယ်။
1985	C++ ကိုပြင်ပမှာအသုံးပြုနိုင်အောင် Cfront interpreter ကို AT&T ကနေဖြန့်ချုပ်ပေးပါတယ်။ Stroustrup ရေးသားတဲ့ The C++ Programming Language စာအုပ်လည်းထွက်ရှိလာပါတယ်။ ဒီချိန်မှာ MS-DOS platform မှာအသုံးပြုနိုင်တဲ့ C compiler တွေလည်းပေါ်လာပါပြီ။

1986	Cfront ကို MS-DOS မှာတင်ပေးပါတယ်။ MS-DOS C compiler တစ်ခုကို Microsoft ကနေဖြန့်ချိခဲ့ပါတယ်။
1987	Borland ကနေ ထုတ်လုပ်တဲ့ Turbo C (for MS-DOS) ဘာ တစ်ခြား version တွေထက်ပိုမိုကောင်းမွန်ပါတယ်။
1989	C++ standardization အတွက် ANSI က စတင်စည်းဝေးပြီဖြစ်ပါတယ်။
1990	C standard document ကို ANSI ကနေ ပုံနှိပ်ထုတ်ဝေခဲ့ပါတယ်။ Borland ကနေ C++ compiler (for MS-DOS) တိဖြန့်ချိပါတယ်။
1991	Microsoft ကနေ C compiler (for MS-DOS) ကို ဖြန့်ချိပါတယ်။
1998	ANSI နဲ့ ISO တိုကနေ C++ standard definition document ကို အပြီးသတ် ပုံနှိပ်ထုတ်ဝေဖြန့်ချိခဲ့ပါတယ်။

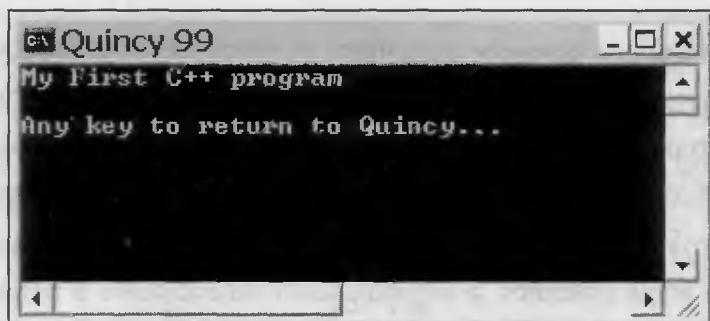
၁၂ My First C++ Program

၁။ ပုံ (၁.၁) မှာဖော်ပြထားတဲ့ Ex101.cpp program ဟာဆိုရင် ကျွန်တော်တို့စတင်ရေးလိုက်တဲ့ C++ program အသေးလေးပါပဲ။ program က သေးပေမယ့် run လိုရပါတယ်။

```
// Listing 1.1: My First C++ program  
  
#include <iostream>  
  
int main()  
{  
    // Display a message on the screen  
    cout << "My First C++ program" << endl;  
  
    return 0;  
}
```

ပုံ (၁၁)

ဒါ program ကို Compile လုပ်ပြီ: Run လိုက်တာနဲ့ ကွန်ပျူတာစကရင်မှာ My First C++ program ဆိုတဲ့စာတော်းမှာ ပေါ်လာတာကိုတွေ့ရပါလိမယ်။ ပုံ (၁၂) ကိုဖြည့်ပါ။



ပုံ (၁၂)

၃။ ဒါ program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- စစချင်းမှာတွေ့ရတဲ့ // Listing 1.1: My First C++ program ဆိုတဲ့ statement ရဲအစက double-slash characters // ကို single-line comment လိုခေါ်ပါတယ်။ အဲဒီနောက်က စာတွေဟာ program အတွက်မှတ်ချက်တစ်ခု ထည့်ရေးထားတဲ့သဘောပါပဲ။ program ထဲမှာ

comment တွေကိုလိုအပ်တဲ့နေရာတွေမှာ ထည့်သွင်းပေးခြင်းအားဖြင့် ဒါ program ကို ဘယ်သူပဲ ဖတ်ဖတ် ရှင်းလင်းလွယ်ကူနေမှာပါ။ program ကို run တဲ့အခါမှာလည်း white space တွေလိုပဲ effect မရှိပါဘူး။ တစ်ကယ်လို comment မှာ စာအများကြီးရေးဖို့လိုရင် comment line တွေကို အခုလို /* */ format နဲ့ပိတ်ပြီးရေးပေးပါ။ ဒါဆိုရင်မှတ်ချက်တွေမှန်း compiler က သိပါတယ်။

- C++ ဟာ free-form language အမျိုးအစားဖြစ်တဲ့အတွက် program ကို compile လုပ်တဲ့အခါမှာ new lines | spaces | tabs | blank lines အစရိတဲ့ white-space character တွေကို အပိုတွေအနေနဲ့ သတ်မှတ်ပြီး compilation မှာ အနောင့်အယူက်မရှိ ချိန်ထားခဲ့မှာ ဖြစ်ပါတယ်။ တစ်ကယ်တော့ program listing မှာ white space တွေကို အသုံးမပြုနိုင်ဘူးဆိုရင် source code တွေဟာ ဖတ်လို ခက်ခဲပါလိမ့်မယ်။ ဥပမာ Listing 1.1 ကို အခုလိုပြောင်းရေးရင် program ကို run လို့ရပေမယ့် program code တွေဟာ ဖတ်လိုမကောင်းတာသောချာပါတယ်။

```
#include <iostream>
int main( ) {           // Display a message on the screen
    cout << "My First C++ program";   return 0; }
```

- #include<iostream> statement ကို preprocessing directive လို့ခေါ်ပါတယ်။ current program မှာ <iostream> header ဖိုင်ထည့်ပေးဖို့ #include directive ကနေ compiler ကိုအမိန့်ပေးနေတာပါ။ console input/output အတွက်အသုံးပြုမယ့် classes | functions နဲ့ global value တွေဟာ <iostream> standard library header ဖိုင်ထဲမှာ အကုန်ရှိနေပါတယ်။ ဒါကြောင့်မို့ cout ကို program မှာအသုံးပြုလိုရတာပါ။ <iostream> မှာ angle bracket < > တွေကိုထည့်ပေးထားတဲ့အဓိပ္ပာယ်က ဒါ header ဖိုင်ကို compiler system ကနေ supply လုပ်ပေးပါလို ဆိုလိုပါတယ်။ တစ်ကယ်လို ကိုယ်ပိုင် header ဖိုင်တွေကို include လုပ်ပေးချင်တယ်ဆိုရင် bracket တွေအား quote (" ") တွေကို အသုံးပြုရပါမယ်။ ဥပမာ #include "myheader.h" ဆိုတာမျိုးပေါ့။
- int main() statement ဟာဆိုရင် main() function ကနေ integer(int) value တစ်ခု ကို return လုပ်ပေးပါလို ကြော်ပေးတာပါ။ main() function ရဲ့ အဝင်မှာ ဘာ argument မှ လက်မခံပါဘူး။

- left brace { သင်္ကာတယာဆိုရင် main() function ထဲက statement block တွေကို စတင်ဖြေရှင်းတော့မယ်လို့ ကြော်ပေးတာပါ။
- cout << "My First C++ program"; ဆိုတဲ့ statement ဟာဆိုရင် cout object ကို အသုံးပြုပြီး My first C++ program ဆိုတဲ့ string constant တစ်ခုကို screen မှာပေါ်လာအောင် compiler ကို display လုပ်ခိုင်းတာပါ။ cout ဟာ BASIC language က PRINT statement နဲ့သဘောအတူတူပါပဲ။ မတူတာက PRINT ဟာ BASIC language ရဲ့အစိတ်အပိုင်းတစ်ခုဖြစ်ပေမယ့် cout ကျတော့ std:: prefix ကနေ compiler ကို (cout ဟာတစ်ခြား identifier တွေနဲ့ မတူကြောင်းကို) အသိပေးပြီး standard input/output library header ဖိုင်ကနေသေချာခွဲထုတ်ဖို့ အမိန့်ပေးတာဖြစ်ပါတယ်။ cout object နဲ့ တွဲထားတဲ့ << stream insertion operator ကို သတိပြုကြည့်ပါ။ stream insertion operator << ဟာ data object တွေကို output stream ထဲမှာထည့်ပေးနိုင်တဲ့အတွက် output operator လို့ခေါ်လည်းရပါတယ်။ ခြားရဲ့ direction ကနေ data object လားရာအရပ်ညွှန်းကို ဖော်ပြန်ပါတယ်။ cin ဆိုရင် input operator >> ကိုအသုံးပြုရမှာပါ။ C++ statement တစ်ခုရေးပြီးတိုင်း semicolon (;) နဲ့ပိတ်ပေးရပါမယ်။ ဒါမှ statement ဆုံးသွားတာကို compiler ကသိမှာပါ။ << endl က next line statement ကိုကူးခိုင်းတာဖြစ်ပါတယ်။ ဒါမှာမဟုတ် << '\n' လို့ရေးလည်းရပါတယ်။
- နောက်ဆုံးနားမှာရေးထားတဲ့ return 0; statement ကဆိုရင် main() function ကိုအလုပ်ရပ်ခိုင်းပြီး constant integer (0) value ကို operating system ဆီ return လုပ်ပေးတာပါပဲ။ right brace (}) သင်္ကာတယာ statement block တွေကို ဖြေရှင်းလိုပြီးသွားကြောင်းကြော်ပေးတာပါ။ ဒါဆိုရင် EX101.cpp program ဟာပြီးသွားပါပြီ။

၁.၃ Identifiers

C++ program တစ်ခုမှာ variables | functions | classes အစရိတ် element မျိုးဆုံးဟာပါဝင်နေပြီး element တိုင်းမှာ နာမည်တစ်ခုစီ ရိုနေကြတယ်လေ။ ဒီနာမည်တွေကို identifier လို့ ခေါ်တာပါပဲ။ လွယ်လွယ်ပြောမယ်ဆိုရင် identifiers ဆိုတာ variable name တွေပေါ့။ ကျွန်တော်တို့ identifier တစ်ခုကိုသတ်မှတ်တဲ့အခါမှာ ဖော်ပြုအချက်အလက်တွေကို လိုက်နာရပါမယ်။

- identifier တစ်ခုမှာ စာလုံးတွေ (letters) । ငကန်းတွေ (digits) နဲ့ underscore (_) character တွေ ပါဝင်လိုပါတယ်။ ဒါပေမယ့် identifier ရဲ့ စီးအကွဲရာဟာ letter သို့မဟုတ် underscore တစ်ခုပဲဖြစ်ရပါမယ်။ underscore (2) ခုတောင် ဖြစ်လို့မရပါဘူး။ ဘာပြုလိုလဲဆိုတော့ အဲဒီ identifier မျိုးတွေကို system အတွက်ပဲ သီးသန်သတ်မှတ်ထားလိုပါ။ C++ program တစ်ခုက လက်သင့်ခံတဲ့ identifier မျိုးတွေဟာ ဥပမာ salary | Amount | TAX | GrossPay | sum_it | TaxRate အစရိုယ်တိဖြစ်ပါတယ်။ လက်သင့်မခံတဲ့ identifier မျိုးတွေကို အောက်မှာဖော်ပြထားပါတယ် । လေ့လာကြည့်ပါ။

4th_July	identifier ရဲ့ ရွှေဆုံးအကွဲရာဟာ ငကန်းဖြစ်လို့မရပါဘူး။
"Amount"	identifier မှာ special character (") တွေ ပါလို့မရပါဘူး။
Kount - 1	အနုတ်လကွာကာလည်း လက်မခံပါဘူး။
num 2	identifier မှာ space ကွက်လပ်ချွန်လို့မရပါဘူး။

- letter တွေကို စာလုံးအကြီး (uppercase letter) သို့မဟုတ် စာလုံးအသေး (lowercase letter) ကြိုက်ရာနဲ့ သတ်မှတ်ပေးလို့ရပါတယ်။ ဒါပေမယ့် C++ ဟာ case-sensitive ဖြစ်တဲ့ အတွက် MyFunc | myFunc သို့မဟုတ် myfunc ဆိုတဲ့ identifier တွေဟာတစ်ခုနဲ့တစ်ခု မတူဘူးဆိုတာကိုသတိပြုပါ။ ဥပမာ area = pi * r * r ဆိုတဲ့ statement တဲ့ **MyFunc = myFunc*myfunc * myfunc** လို့ပြောင်းရေးရင် သဘောအတူတူပါပဲ။
- identifier name ကို ကြိုက်သလောက် နာမည်ရည်ရည်ပေးလို့ ရပါတယ်။ ဒါပေမယ့် ရွှေဆုံးက အကွဲရ (32) လုံးပဲ identifier နဲ့တိုက်ရိုက်အကျိုးသက်ရောက်မှာပါ။ C++ language အတွက် သီးသန်သတ်မှတ်ထားတဲ့ C++ keyword တွေကို identifier တွေအဖြစ် အသုံးပြုလို့မရပါဘူး။

၁.၄ Keywords

C++ program တစ်ခုမှာ identifier name တွေအနေနဲ့ အသုံးပြုခွင့်မရှိတဲ့ standard reserved word တွေကို C++ keyword တွေလို့ခေါ်ပါတယ်။ နာမည်ကွက်တိမတူရင်တော့ သုံးလို့ရပါတယ်။ ထေား (o. j.) မှာ C++ keyword တွေကိုဖော်ပြထားပါတယ် । လေ့လာကြည့်ပါ။

Non-Identifiers

asm	else	operator	throw
auto	enum	private	true
bool	explicit	protected	try
break	extern	public	typedef
case	false	register	typeid
catch	float	reinterpret_cast	typename
char	for	return	union
class	friend	short	unsigned
const	goto	signed	using
const_cast	if	sizeof	virtual
continue	inline	static	void
default	int	static_cast	volatile
delete	long	struct	wchar_t
do	mutable	switch	while
double	namespace	template	
dynamic_cast	new	this	

၁.၅

Variables

variable ဆိတ် program ထဲက data value တွေကို ထားသိမ့်အသုံးပြုတဲ့ storage နေရာတစ်ခုပါပဲ။

variable တိုင်းမှာ type name တစ်ချို့စီရွှေပါတယ်။ type ဆိုတာ variable ရဲ့ format နဲ့ data အမျိုးအစားကို သတ်မှတ်ပေးတာပါ။ variable တစ်ခုရဲ့တန်ဖိုးတွေကို program ထဲမှာ ကြိုက်သလိုပြောင်းလဲလို့ရပေမယ့် သူနဲ့ ပတ်သက်တဲ့ data type ကတေသူမပြောင်းပါဘူး။ type declaration မှာ ကြော်ထားတဲ့အတိုင်း အမြှုပိနေမှုပါပဲ။ C++ မှာပါဝင်တဲ့ data type တွေက bool | char | wchar_t | int | float နဲ့ double တို့ဖြစ်ပါတယ်။ variable type တွေကိုထပ်ခွဲဦးမယ်ဆိုရင် unsigned | long | short | long double တို့ဆိုတာရှိပါသေးတယ်။

The bool Type

၁။ bool variable ဆိုတာ true ဆိုမဟုတ် false ဆိုတဲ့ value (2) မျိုးကိုအသုံးပြုတဲ့ boolean (logical) type အမျိုးအစားပါ။ arithmetic expression တစ်ခုမှာ bool variable ကို အသုံးပြုမယ်ဆိုရင် expression က true ဖြစ်တဲ့အခါမှာ integer value 1 ကို return လုပ်ပေးပါလိမ့်မယ်။ false ဆိုရင် zero ကို return လုပ်မှုပါ။ တစ်ကယ်လို့ integer တစ်ခုကို bool ဖြစ်အောင်ပြောင်းပေးမယ်ဆိုရင် integer 0 ဟာ false ဖြစ်ပြီး integer nonzero က true ဖြစ်မှုပါပဲ။ ပုံ (၁။၃) မှာဖော်ပြထားတဲ့ Ex102.cpp program ဟာဆိုရင် bool variable အသုံးပြုနည်းကိုရေးထားပါတယ် । လေ့လာကြည့်ပါ။

```

// Listing 1.2: Using a bool variable

#include <iostream>

int main()
{
    bool flag; // bool variable

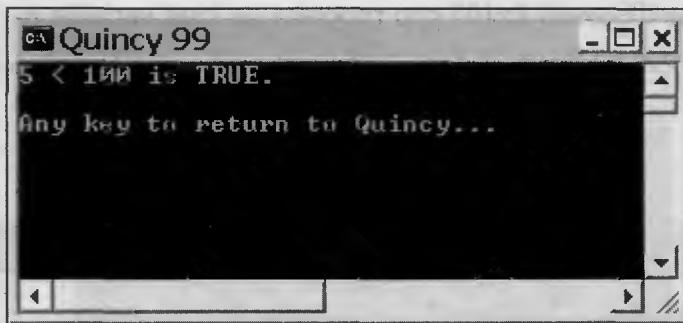
    flag = 5 < 100; // Set a logical value to flag
    if (flag)
        cout << "5 < 100 is TRUE." << endl;

    return 0;
}

```

ပုံ (၁။၃)

program ကို run လိုက်တာနဲ့ကွန်ပျူတာစကရင်မှာ $5 < 100$ is TRUE. ဆိုတဲ့စာကြောင်း ပေါ်လာ
သွေးပါလိမ့်မယ်။ ပုံ (၁၁၄) ကိုဖြည့်ပါ။



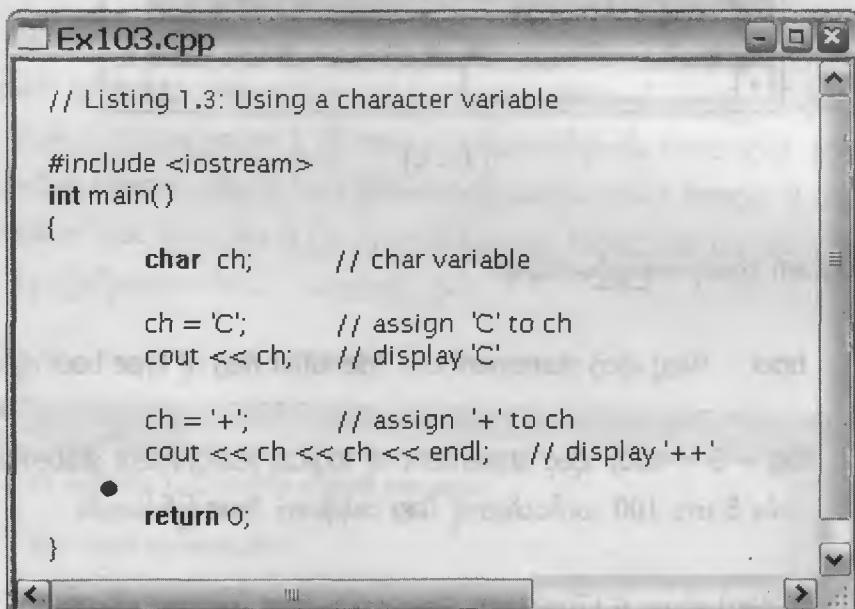
ပုံ (၁၁၄)

၃ program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- `bool flag;` ဆိုတဲ့ statement ဟာ identifier `flag` ကို type `bool` လို့ ကြောာတာပါ။
- `flag = 5 < 100;` ဆိုတဲ့ statement ကို logical assignment statement လိုခေါ်ပါ
တယ်။ 5 ဟာ 100 ထက်ငယ်တာမို့ `flag` တန်ဖိုးဟာ True ဖြစ်ပါတယ်။
- `if (flag)` ဆိုတာ if (True) လားလို့မေးတာပါ။ True ဖြစ်တဲ့အတွက် `cout << "5 < 100
is TRUE."` ဆိုတဲ့ statement ကို execute လုပ်ပြီး console (screen) မှာ $5 < 100$ is
TRUE. ဆိုတဲ့ string constant ကို display လုပ်ပြတာပါ။ တစ်ကယ်လို့ `flag = 5 > 100`
ဆိုရင် False ဖြစ်တဲ့အတွက် if (flag) နောက်က statement ကိုမဖြေရှင်းပဲ ကျော်သွားမှာပါ။
- နောက်ဆုံးမှာရေးထားတဲ့ `return 0;` statement ကိုတွေ့ရင် main() function ရဲ့အလုပ်
ပြီးသွားပြီ့ constant integer (0) value ကို operating system ဆီ return လုပ်ပေးပါ
လိမ့်မယ်။ ပြီးတော့ရင် right brace (}) သက်တကောင်း program ပြီးဆုံးကြောင်းကို
ကြောာပါတယ်။

The char Type

၁။ char variable ဆိုတာ 8-bit bytes ASCII character set ထဲက character value တွက် ဆိုလိပါတယ်။ ဥပမာ ch ဆိုတဲ့ identifier တစ်ခုကို type char နဲ့ declare လုပ်မယ်ဆိုရင် char ch; လိုပေးရင် ရပါတယ်။ ပုံ (၁.၅) မှာဖော်ပြထားတဲ့ Ex103.cpp program မှာ char variable အသုံးပြုနည်းကိုရေးထားပါ တယ်။ Ex103.cpp program ကို run လိုက်တာနဲ့ ကွန်ပျူးတာစကရင်မှာ C++ ဆိုတဲ့စာကြောင်းပေါ်လာမှာပါပဲ။ ပုံ (၁.၅) ကိုကြည့်ပါ။



```
// Listing 1.3: Using a character variable

#include <iostream>
int main()
{
    char ch;      // char variable

    ch = 'C';      // assign 'C' to ch
    cout << ch;    // display 'C'

    ch = '+';      // assign '+' to ch
    cout << ch << ch << endl; // display '++'

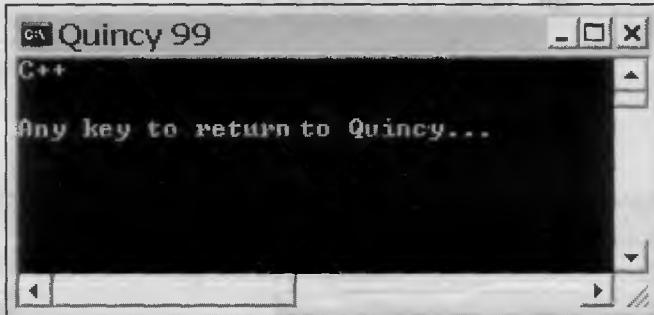
    return 0;
}
```

ပုံ (၁.၅)

၂။ ပုံ (၁.၆) မှာ Ex103.cpp program ကို run ပြထားပါတယ်။ ကွန်ပျူးတာစကရင်မှာ C++ ဆိုတဲ့စာကြောင်းပေါ်လာတာကို တွေ့ရပါလိမ့်မယ်။ program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- ဝေချင်းမှာ identifier ch ကို type char လိုကြပြောပြီး ch ကို character 'C' နဲ့ assign လုပ်ပေးပါတယ်။ ပြီးတော့ရင် ch ရဲ့ value ဖြစ်တဲ့ C စာလုံးကို screen မှာ display လုပ်ဖြပ်လိမ့်မယ်။

နောက်တစ်ခါ ch ကို character constant '+' နဲ့ assign လုပ်ပေးပါတယ်။ ပြီးတော့ရင် cout << ch << ch; ဆိုတဲ့ statement ကနေ C ဘလုံးနောက်မှာ '+' သက်တ (2) ခဲ့ကိုပြီး display လုပ်ပြပါလိမ့်မယ်။ ဒါကြောင့်မို့ကွန်ပူးတာမှာ C++ ဆိုတဲ့စာကြောင်းပေါ်လာတာပါပဲ။ ပုံ (၁. ၆) ကိုဖြည့်ပါ။



ပုံ (၁. ၆)

The wchar_t Type

C++ compiler မှာ wide character class library ရှိနေမယ်ဆိုရင် wcout object ကို အသုံးပြုပြီး 16-bits wide char data type ကနေ character တွေယူပြီး stream output လုပ်ပေးနိုင်ပါတယ်။

The int Type

၁။ int type specifier တွေမှာ plain signed integer ဥပမာ int counter; ဆိုတာမျိုးရှိသလို signed int, unsigned int, long int, short int စသည်ဖြင့် အမျိုးမျိုးရှိနေကြပါတယ်။ int ကို အသုံးပြုမယ်ဆိုရင် range က -32768 to 32767 ဖြစ်ပါတယ်။ short int နဲ့အတူတူပါပဲ။ unsigned int ဆိုရင် range: 0 to 65535 ဖြစ်ပါတယ်။ signed long int ရဲ့ range ဟာ -2147483648 to 2147483647 ဖြစ်ပြီး unsigned long int ရဲ့ range က 0 to 4294967295 ပါ။ ပုံ (၁. ၇) မှာဖော်ပြထားတဲ့ Ex104.cpp program မှာ int variable သုံးနည်းကို ဖော်ပြထားပါတယ်။

```
// Listing 1.4: Using an int variable.

#include <iostream>
int main()
{
    int num; // int variable;

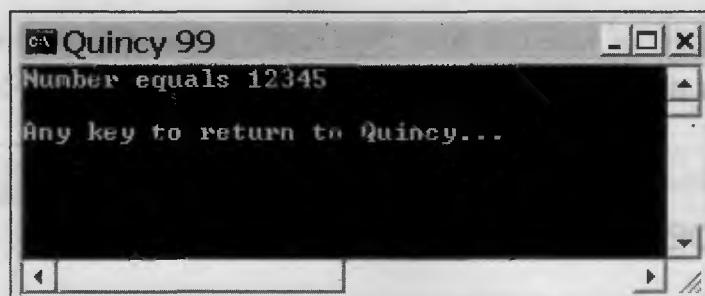
    num = 12345; // assign 12345 to num

    // display the number
    cout << "Number equals " << num << endl;

    return 0;
}
```

ပုံ (၁၃)

၂။ Ex104.cpp program ကို run လိုက်မယ်ဆိုရင် ကွန်ပူးတာစကရင်မှာ Number equals 12345 ထိပေါ်လာပါလိမ့်မယ်။ ပုံ (၁၄) ကိုဖြည့်ပါ။



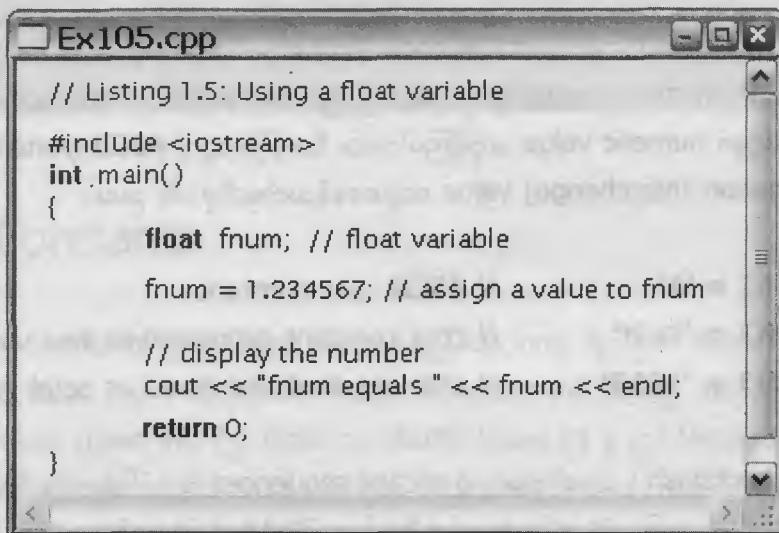
ပုံ (၁၅)

၃။ ဒါ program ကိုလေ့လာကြည့်ရင်

- စေချင်းမှာ identifier num ကို type int လိုကြပြာပါတယ်။ နောက်ပြီး num ကို 12345 ဝက်နှုန်းတန်ဖိုးနဲ့ assign လုပ်ပေးပြီး num ခဲ့ value ကို string constant တစ်ခုနဲ့တွဲပြီးတော့ display လုပ်ပြုခိုင်းတာပါပဲ။

The Floating-Point Numbers

C++ මා ව්‍යවහරක්ෂ:ඡෘෂ්ඨංජිතයා (floating-point number) (3) අශ්‍යාගින්නයා ප්‍රික්ෂීලිතයා|| ඇමි float | double න් long double තුළිපිට|| float බැව range භා 3.4E-38 to 3.4E1038 | double බැව range භා 1.7E-308 to 1.7E308 න් long double බැව range භා 3.4E-4932 to 1.1E4932 තුළි ප්‍රික්ෂීලිතයා කිවත්මුත්වුරු තොග දු ව්‍යෙහභාතුත්වුරු|| compiler ඡෘෂ්ඨංජිතයා ප්‍රික්ෂීලිතයා|| දු (o. 0) මා ප්‍රියාකාශයාත් න් Ex105.cpp program භා float න් ඡෘෂ්ඨංජිතයා ප්‍රික්ෂීලිතයා | ලදාග්‍රිනුවු||

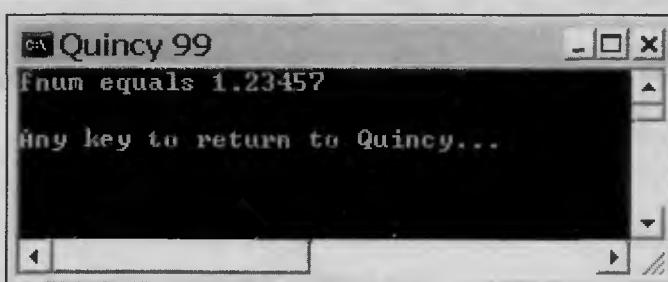


```
// Listing 1.5: Using a float variable
#include <iostream>
int main()
{
    float fnum; // float variable
    fnum = 1.234567; // assign a value to fnum

    // display the number
    cout << "fnum equals " << fnum << endl;
    return 0;
}
```

දු (o. 0)

program ගි run ලදිරිදි ගුණුවුතා තාක්ෂණික මා fnum equals 1.23457 ලදි ප්‍රිඩ්‍රුමයා||
දු (o. 00) ගිග්‍රිනුවු|| default ඡෘෂ්ඨංජිතයා (5) එක්ස්ප්‍රෝ ප්‍රික්ෂීලිතයා||



දු (o. 00)

၁.၆ Constants

C++ program တစ်ခုမှာ တန်ဖိုးမပြောင်းလဲပဲ ကိန်းသေဖြစ်နေတဲ့ value မျိုးတွေကို constants လို ခေါ်ပါတယ်။ constant ဘယ်နှစ်မျိုးရှိလဲဆိုတော့ (5) မျိုးပါ။ (၁) Character constants (၂) Integer constants (၃) Floating-point constants နဲ့ (၄) Address constant တို့ဖြစ်ကြပါတယ်။

Character Constants

Character constants ဆိုတာ single code (' ') တွေနဲ့ပိတ်ထားတဲ့ char value တွေကိုဆိုလိုပါ တယ်။ တစ်ကယ်တော့ character ဟာ letter ဖြစ်ဖြစ် digit ဖြစ်ဖြစ်၊ blank ကိုတောင် character လိုခေါ်လိုရတယ် လေး။ character တွေမှာ numeric value တွေရှိကြပါတယ်။ ဒီတန်ဖိုးတွေကို ASCII (American Standard Code for Intermentation Interchange) value တွေအနေနဲ့သတ်မှတ်မှုပါ။ ဥပမာ

```
ch1 = 'A'           // ASCII char constant  
ch2 = '\x2f'        // char constant expressed as hex value  
ch3 = '\013'        // char constant expressed as octal value
```

ဒီဥပမာမှာ backslash \ သက်တတွေကို escape sequences လို ခေါ်ပါတယ်။ ဒီသက်တပါလာရင် character constant မှာ ထူးခြားတဲ့ ရည်ညွှန်းချက်တစ်ခု ရှိနေပါပြီ။ \a လိုပါလာရင် audible bell character ဖြစ်သလို \n ဆိုရင် output data အတွက် newline character ဖြစ်ပါတယ်။ ယေား (၁.၃) မှာ constant escape sequence တွေကိုဖော်ပြထားပါတယ် । လေ့လာကြည့်ပါ။

ယေား (၁.၃) **Constant Escape Sequences**

Escape Sequence	Description
'	Single quote
"	Double quote
\	Backslash
\0	Null (zero) character

\0nnn	Octal number (nnn)
\a	Audible bell character
\b	Backspace
\f	Formfeed
\n	Newline
\r	Carriage return
\t	Horizontal tab
\v	Vertical tab
\x	Hexadecimal number (nnn)
\?	question mark

Integer Constants

Integer constants ဆိတ် type name တွေဖြစ်တဲ့ long သိမဟုတ် short + signed သိမဟုတ် unsigned fixed value တွေကို ဆိုလိုတာပါ။ integer constant မှာပဲ (3) မျိုး ထပ်ခွဲထားပါသေးတယ်။ (၁) Decimal constants (base 10) (၂) Octal constants (base 8) နဲ့ (၃) Hexadecimal constants (base 16) တို့ပါပဲ။ decimal constant တစ်ခုဟာဆိုရင် သူညောန (9) ထဲက ဂဏန်းတွေနဲ့စပေါင်းဖြစ်နေတာပါ။ constant မှာ ဂဏန်း (2) လုံး သိမဟုတ် (2) လုံးထက် ပိုပါနေမယ်ဆိုရင် ထိပ်ဆုံးဂဏန်းဟာ သူညာဖြစ်လို့မရပါဘူး။ ပေါက်ပြီး constant value ဟာ သတ်မှတ်ထားတဲ့ lower bound နဲ့ upper bound ကြားက တန်ဖိုးတွေပဲ ပါမယ်။ C++ program က လက်သင့်ခံတဲ့ decimal constant တွေက ဥပမာ 0 + 1234 + 32700 တို့ပါပဲ။ အောက်ပါ constant တွေကတော့ decimal constant တွေ မဟုတ်ကြပါဘူး။

23,456	constant မှာ comma (,) ပါလို့မရပါဘူး။
54.68	ဒသောမလည်း ပါလို့မရပါဘူး။
32 591	space ကွက်လပ်ချွန်လို့မရပါဘူး။
0987	စဉ်းအကွွာရာဟာ သူညာဖြစ်လို့မရပါဘူး။
956705	constant value ဟာ upper bound limit ကိုကျော်နေပါတယ်။ ဒါဆိုရင် decimal constant မဖြစ်ပါဘူး။

Octal constant တွက သုညနဲ့ (7) ထဲကိုကန်းတွနဲ့ စုပေါင်းဖြစ်နေတာပါ။ octal constant မှန်းသိအောင်လို့ ထိပ်ဆုံးကို သုညအမြိတ်သုံးပေးရပါမယ်။ ဥပမာ 0775 ဖြစ်ပါတယ်။

Hexadecimal constant တွကျတော့ သုညကနဲ့ (9) ထဲက ကန်းတွနဲ့အတူ (a) ကနဲ့ (f) ထဲက letter တွနဲ့ စုပေါင်းဖြစ်နေတာပါ။ hexadecimal မှန်းသိအောင်လို့ ထိပ်ဆုံးအကွာရာ (2) ခုကို 0x သို့မဟုတ် 0X အမြိတ်သုံးပေးရပါမယ်။ ဥပမာ 0x54fa ဖြစ်ပါတယ်။

Floating-Point Constants

Floating-point constants ဆိုတာ base 10 ကို အခြေခံတဲ့ real number တွေကို ဆိုလိုပါတယ်။ ဒီ constant တွေကို အသေမဂကန်း (floating point) အနေနဲ့ဖြစ်စေ သို့မဟုတ် ထပ်ကိန်း (exponent) ပုံစံနဲ့ဖြစ်စေ သတ်မှတ်လို့ရပါတယ်။ ဥပမာ 1234.56 | 2E-8 | 2.55E+38 တို့ ဖြစ်ပါတယ်။ အောက်ပါ constant တွေကိုတော့ C++ က floating-point constants တွေလို့ မသတ်မှတ်ပါဘူး။

1234	constant မှာ decimal point ဖျောက်နေပါတယ်။ ဒီအားမပေါ်ဘူး။
15E+2.5	ထပ်ကိန်းဟာ ကိန်းပြည့်ကဏ္ဍးဖြစ်ရပါမယ်။
12e 5	space ကွက်လပ်ချိန်လို့မရပါဘူး။

String Constants

String constants ဆိုတာ double quotes (" ") ထဲမှာ ရေးထားတဲ့အကွာရာတွေကို ဆိုလိုတာပါ။ ဒီအကွာရာတွေထဲမှာ blank ပါလည်းရပါတယ်။ အကွာရာအရေအတွက် ဘယ်လောက်ပါရမယ်လို့ မကန့်သတ်ပါဘူး။ အောက်ဖော်ပြုပါ string constant တွေကို C++ program က လက်ခံပါတယ်။

"Hello C++"

"MyanMar"

"25/10/03"

ဤ (၁၀.၁၁) မှာဖော်ပြထားတဲ့ Ex106.cpp program ဟာဆိုရင် string အရှည်ကြီးရေးတာကို string concatenation နည်းနဲ့ဆက်ပေးထားပါတယ်။ လေ့လာကြည့်ပါ။

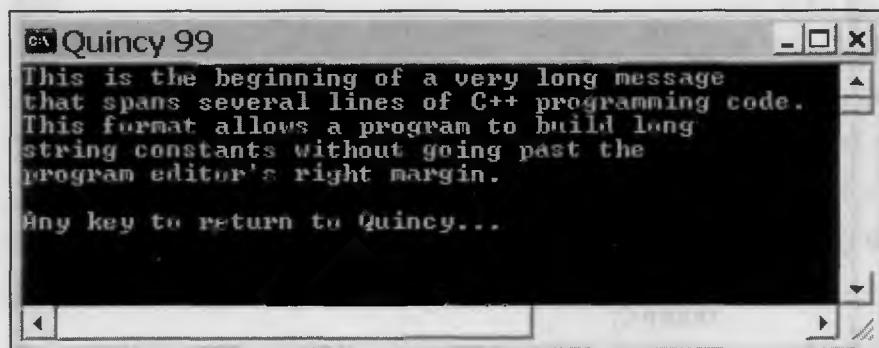
```
// Listing 1.6: Concatenated string constants
#include <iostream>

int main()
{
    cout << "This is the beginning of a very long message\n"
        "that spans several lines of C++ programming code.\n"
        "This format allows a program to build long\n"
        "string constants without going past the\n"
        "program editor's right margin.\n";

    return 0;
}
```

ပုံ (၁၁၁)

- Ex106.cpp program ကို run လိုက်မယ်ဆိုရင် ကွန်ပူးတာမှာ အခုလိပ်လာတာကိုတွေ့ရပါလိမ့်မယ်။
ပုံ (၁၁၂) ကိုဖြည့်ပါ။



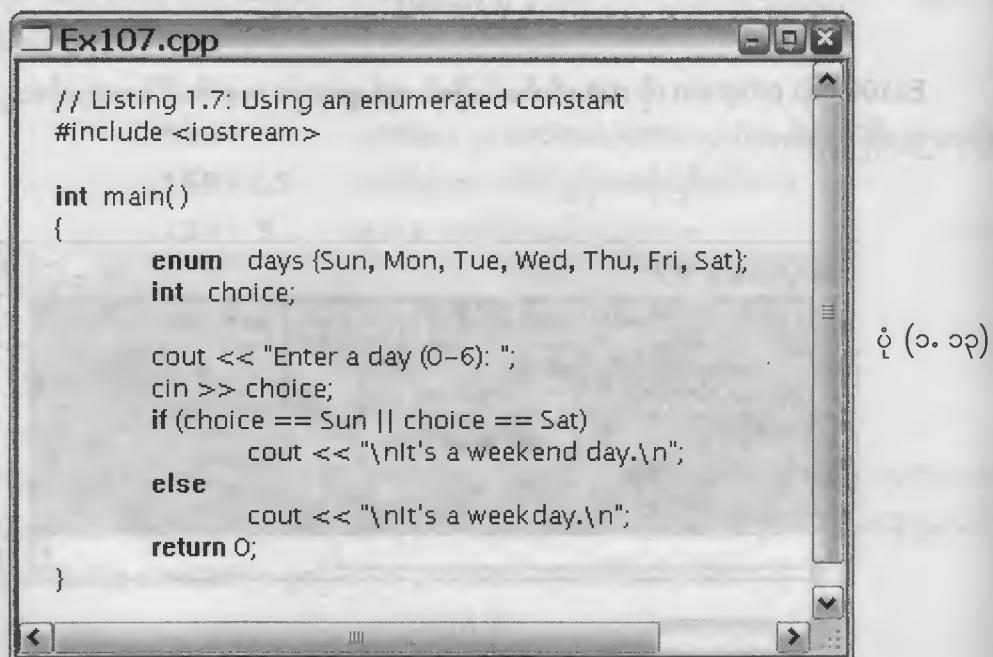
ပုံ (၁၁၂)

- program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- string constant အရှည်တွဲးကို newline escape sequence \n အသုံးပြုဖြီး ခွဲရေးခြင်း
အားဖြင့် စာပိုင်အရှည်တွဲးတစ်ခုကို ဘောင်မကျော်အောင် display လုပ်ပြုခိုင်းလိုရပါတယ်။

Enumerated Constants

၁။ program တစ်ခုမှာ သတ်မှတ်ထားတဲ့ data set တစ်ခုကို type တစ်ခုအနေနဲ့ သတ်မှတ်ပေးချင်ရင် enumerated constant ကိုအသုံးပြုလိုပါတယ်။ ဥပမာ COLOR ကို enumeration တစ်ခုလိုကြပြာပေးပြီး (RED, BLUE, GREEN,WHITE, BLACK) ဆိုတဲ့အရောင် (5) မျိုးနဲ့သတ်မှတ်ပေးချင်ရင် enum COLOR {RED, BLUE, GREEN,WHITE, BLACK}; ဆိုတဲ့ statement ကိုရေးရင်ရပါတယ်။ COLOR ဆိုတာက enumeration type name ပါ။ RED အတွက် symbolic constant ကို 0 လိုသတ်မှတ်ပါတယ်။ ကျွန်တာတွေ ကို 1, 2, 3, 4 လို အစီအစဉ်လိုက် သတ်မှတ်ပေးသွားမှာပါ။ ဒါမှမဟုတ် enum COLOR {RED=50, BLUE, GREEN=500,WHITE, BLACK}; လိုကြပြာပေးမယ်ဆိုရင် BLUE = 51 ; WHITE = 501 နဲ့ BLACK = 502 လို enumeration type က ပြောင်းလဲသတ်မှတ်ပေးသွားမှာပါ။ ပုံ (၁၀.၁၃) မှာဖော်ပြထားတဲ့ Ex107.cpp program ကိုလေ့လာကြည့်ပါ။

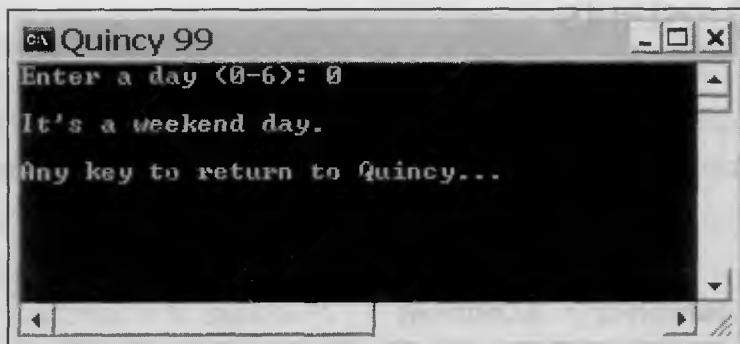


```
// Listing 1.7: Using an enumerated constant
#include <iostream>

int main()
{
    enum days {Sun, Mon, Tue, Wed, Thu, Fri, Sat};
    int choice;

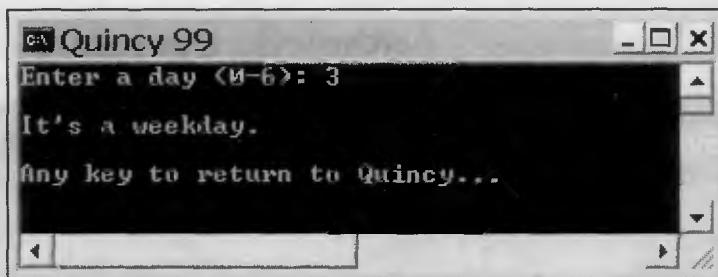
    cout << "Enter a day (0-6): ";
    cin >> choice;
    if (choice == Sun || choice == Sat)
        cout << "\nIt's a weekend day.\n";
    else
        cout << "\nIt's a weekday.\n";
    return 0;
}
```

၂။ အခြင် program ကို run လိုက်ပြီး Enter a day (0-6) : လို prompt တစ်ခုပေါ်လာတဲ့အခါမှာ ၀ ကိုရိုက်ထည့်ပြီး ENTER နှင့်လိုက်တာနဲ့ It's a weekend day. လိုပေါ်လာပါလိမ့်မယ်။ ပုံ (၁၀.၁၄) ကိုကြည့်ပါ။



ဒီ program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- စေချင်: identifier days ကို type enum လိုအကြောဖြီး {Sun, Mon,Tue,Wed,Thu, Fri, Sat } တို့နဲ့ assign လုပ်ပေးပါတယ်။ identifier choice ကို type int လိုအကြောပါတယ်။
- cout << "Enter a day (0-6) : "; ဆိုတဲ့ statement ကနေ choice အတွက် data ထည့်ပေးဖို့ prompt လုပ်ပေးပါတယ်။ cin >> choice; ကနေ chice အတွက် data ကို 0 လိုချိုက်ထည့်ပေးမယ်ဆိုပါစို့။
- ဒါဆိုရင် if (choice == Sun || choice == Sat) statement ကနေ choice ဟာ Sunday (=0) သို့မဟုတ် Saturday (=6) လားဆိုတာကိုစိစစ်ပါတယ်။ 0 ဖြစ်တဲ့အတွက် if () အောက်တည့်တည့်က cout << "\nIt's a weekend day.\n"; ဆိုတဲ့ statement ကို execute လုပ်မှာဖြစ်ပါတယ်။ ဒီတော့ ကွန်ပျူတာ It's a weekend day. လိုပေါ်လာပြီပေါ့။ တစ်ကယ်လို့ choice အတွက် data ကို 3 ထည့်ပေးရင် It's a weekday. လို့ display လုပ်ပြုမှပါ။ ပုံ (၁၁၁၅) ကိုကြည့်ပါ။



the first time in the history of the world, the people of the United States have been called upon to make a choice between two opposite ways of life, between two different philosophies of government. They have been asked, as a nation, to decide whether they will accept the principles which have long been dear to Americans, or whether they will substitute for them other principles which have been accepted by other nations.

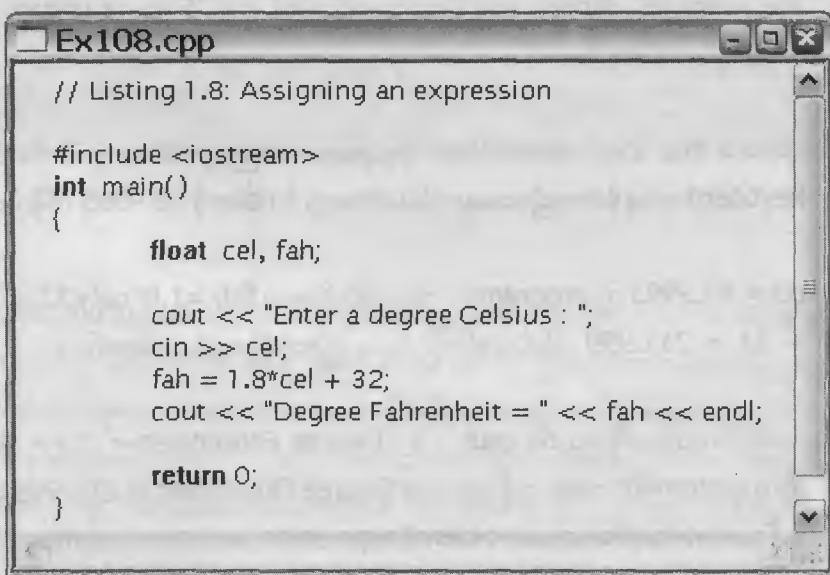
The principles which have long been dear to Americans are those of freedom, of equality, of justice, of individual rights, of the right of self-government, of the right of the people to control their own affairs, of the right of the people to choose their own leaders, of the right of the people to change their leaders when they do not like them, of the right of the people to express their opinions freely, of the right of the people to assemble together, of the right of the people to petition the government for redress of grievances, of the right of the people to bear arms, of the right of the people to have a free press, of the right of the people to have a free church, of the right of the people to have a free school, of the right of the people to have a free government, of the right of the people to have a free country, of the right of the people to have a free world.

The principles which have been accepted by other nations are those of despotism, of

-	Unary minus
*	Multiplication
/	Division
%	Modulus
+	Addition
-	Subtraction

၁.၈ Assignments

၁။ Assignment statement တစ်ခုဟာဆိုရင် expression တစ်ခုကိုဖြေရှင်းလိုက်လို ရရှိလာတဲ့ return value ကို variable တစ်ခုနဲ့ assign လုပ်ပေးမှာပါ။ ပုံ (၁.၁၆) မှာဖော်ပြထားတဲ့ Ex108.cpp program ဟာ Celsius degree တစ်ခုကို $1.8*C+32$; ဆိုတဲ့ expression ကို အသုံးပြုပြီး degree Fahrenheit ကိုဖြောင်းလဲပေးတဲ့ program ဖြစ်ပါတယ်။



```
// Listing 1.8: Assigning an expression

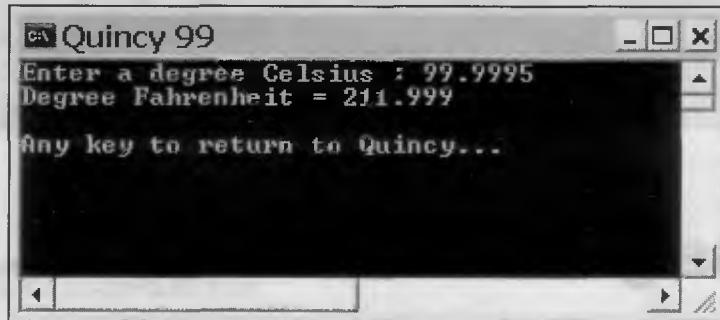
#include <iostream>
int main()
{
    float cel, fah;

    cout << "Enter a degree Celsius : ";
    cin >> cel;
    fah = 1.8*cel + 32;
    cout << "Degree Fahrenheit = " << fah << endl;

    return 0;
}
```

ပုံ (၁.၁၆)

J|| Ex108.cpp program ති run ලදී මයි ස්ථිර ගුණ්මාතා පෙන්වනු ලදී Enter a degree Celsius : ලදී prompt තත්ත්වපේෂීලාපිලිම් මයි 99.9995 ගියුරු තත්ත්වයේ ENTER අනුව ලදී තැන් Degree Fahrenheit = 211.999 ලදී පේෂීලාපිලියි දු (o. o2) ගියුරු තත්ත්වයි



දු (o. o2)

R|| ඩි program ගියුරු පෙන්වනු මයි ස්ථිර නිස්සා

- චතුවාන්: identifier cel න් fah තියු ගියුරු type float ලදී ගියුරු ආවායියි: Enter a degree Celsius: ස්ථිත්තා ගියුරු නිවාද: screen මාපේෂීලා නොවනු cout << "Enter a degree Celsius" ; ගැනීලුද් පෙෂීලා පිටයි
- cin >> cel; ස්ථිත්තා statement ගියුරු නිවාද නිවාද cel නිවාද data නිවාද user නිවාද keyboard ගැනීලුද් පෙෂීලා පිටයි cel නිවාද data නිවාද 99.9995 ගියුරු තත්ත්වයේ පෙෂීලා පිටයි
- cel = 99.9995 ලදී program නිවාද විවාහා පිටයි fah = 1.8*cel+32 = 1.8*99.9995 + 32 = 211.999 ස්ථිත්තා පෙෂීලා පිටයි දු වෙනත් වූයි පිටයි
- පෙෂීලා පිටයි නිවාද cout >> "Degree Fahrenheit = " >> fah >> endl; ස්ථිත්තා statement ගැනීලුද් පෙෂීලා පිටයි Degree Fahrenheit = 211.999 ස්ථිත්තා ගියුරු නිවාද: පේෂීලා නොවනු පෙෂීලා පිටයි << endl expression නිවාද next statement line ගියුරු නිවාද: නිවාද

Logical and Relational Operators

C++ language မှာ ပါဝင်တဲ့ relational and logical operator တွေကို ယေး (၁.၅) မှာ ဖော်ပြထားပါတယ်။ ဒီ operator တွေကိုအသုံးပြုပြီး expression (2) ခုကို နှိမ်းယူပြုခိုင်းမယ်။ နောက်ပြီး true (or false) value ကို return လုပ်ခိုင်းလို့ရပါတယ်။

ယေး (၁.၅) **Logical and Relational Operators**

Symbol Description

&&	Logical AND
	Logical OR
!	Unary NOT
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
==	Equal to
!=	Not equal to

relational and logical operator တွေအသုံးချခနည်းကို ပုံ (၁.၁၈) က C++ program မှာတင်ပြထားပါတယ်။ ဒီ program က ဉာဏ်တစ်ခုရဲ့ အရိယာရှာတဲ့ program ပါ။ ဉာဏ်တစ်ခုရဲ့ အရိယာကိုမရှာခင်မှာ ပေးထားတဲ့ data တွေနဲ့ ဉာဏ်ဖြစ်မဖြစ်ဆုံးတာကို အရင်စစ်ဆေးရပါမယ်။ ဉာဏ်ဖြစ်မှ အရိယာကိုရှာလို့ရမှာပါ။ ဒါဖြင့် ဘယ်လိုစစ်ရမလဲ။ ဉာဏ်တစ်ခုရဲ့ အနားနှစ်ဖက်ပေါင်းခြင်းဟာ တတိယအနားထက် အမြဲတိုးရမယ်မဟုတ်လား။ အဲဒါမဖြစ်ရင် ဉာဏ်မဖြစ်ဘူး။ အခုစစ်ဆေးနည်းမှာ relational and logical operator တွေရဲ့အသုံးကျပုံကို Ex109.cpp program မှာတွေ့ပါလိမ့်မယ်၊ လေ့လာကြည့်ပါ။

```
Ex109.cpp
// Listing 1.9: Using the relational and logical operators

#include <iostream>
int main()
{
    double a,b,c,s,area;

    cout << "Enter three sides of triangle\n";
    cin >> a >> b >> c;
    if ((a+b>c) && (b+c>a) && (c+a>b))
    {
        s = (a+b+c)/2;
        area = sqrt(s*(s-a)*(s-b)*(s-c));
        cout << "\nArea of triangle = " << area << endl;
    }
    else
        cout << "\nTriangle can't be formed";

    return 0;
}
```

ပုံ (၁၁၁)

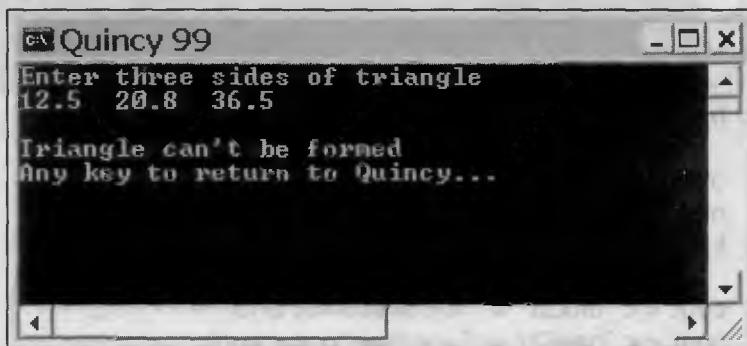
အခြင် program ကို run လိုက်မယ်ဆိုရင် ကွန်ပျူတာစကရင်မှာ Enter three sides of triangle ထို prompt တစ်ခုပေါ်လာပါလိမ့်မယ်။ ၃ <space> ၄ <space> ၅ ကိုရှိက်ထည့်ပြီး ENTER နှင့်လိုက်တာနဲ့ Area of triangle = 6 ဆိုတဲ့ အဖြေကိုတွက်ပေးပါလိမ့်မယ်။ ပုံ (၁၁၂) ကိုကြည့်ပါ။

```
Quincy 99
Enter three sides of triangle
3 4 5
Area of triangle = 6
Any key to return to Quincy...
```

ပုံ (၁၁၂)

၃ program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- စစချင်း program မှာအသုံးပြုမယ့် a | b | c | s | area ဆိုတဲ့ identifier တွေအတွက် data type ကို double လိုကြောပေးပါတယ်။
- Enter three sides of a triangle ဆိုတဲ့ prompt ကို display မှာပေါ်ချင်းပြီး a | b | c တန်ဖိုးတွေကို (space ခြားပြီး) keyboard ကနေရှိက်ထည့်ပေးရပါမယ်။ a | b | c တို့၏ တန်ဖိုးတွေကို a=3 | b=4 နဲ့ c=5 လို့ ရှိက်ထည့်မယ်ဆိုပါစိုး။
- ဒါဆိုရင် ပေးထားတဲ့ data တွေနဲ့တို့ပြစ်မဖြစ်ဆိုတာကို relational operator(>) နဲ့ logical operator(&&) နှစ်ခုကိုအသုံးပြုပြီး စစ်ဆေးပါတယ်။ ဆိုလိုတာက $(a = 3) + (b = 4)$ ဟာ $(c = 5)$ ထက်တိုးမယ်။ $(b = 4) + (c = 5)$ က $(a = 3)$ ထက်တိုးမယ်။ $(c = 5) + (a = 3)$ က $(b = 4)$ ထက်တိုးမယ်ဆိုရင် s နဲ့ area တိုက်ဆက်တွက်ပြီး Area of triangle = 6 လို့ display မှာပေါ်ပေးပါလိမ့်မယ်။ sqrt() function ဟာဆိုရင် အပေါင်းလက္ခဏာကိန်းတစ်ခု ရွှေ့နှုန်းတွေကိုရှာပေးနိုင်တဲ့ math function တစ်ခုဖြစ်ပါတယ်။
- တစ်ကယ်လို့ပေးထားတဲ့ data တွေနဲ့တို့ပြစ်ဘူးဆိုရင် Triangle can't be formed ဆိုတဲ့ သတိပေးချက်ကို ကွန်ပျူတာက ကြောပေးပါလိမ့်မယ်။ ဥပမာ a = 3 | b = 4 နဲ့ c = 9 ဆိုတဲ့ data တွေကိုထည့်ပြီး run ကြည့်ပါ။ အဖြေပေါ်လာပါလိမ့်မယ်။ ပုံ (၁၁၂၀) ကိုကြည့်ပါ။



ပုံ (၁၁၂၀)

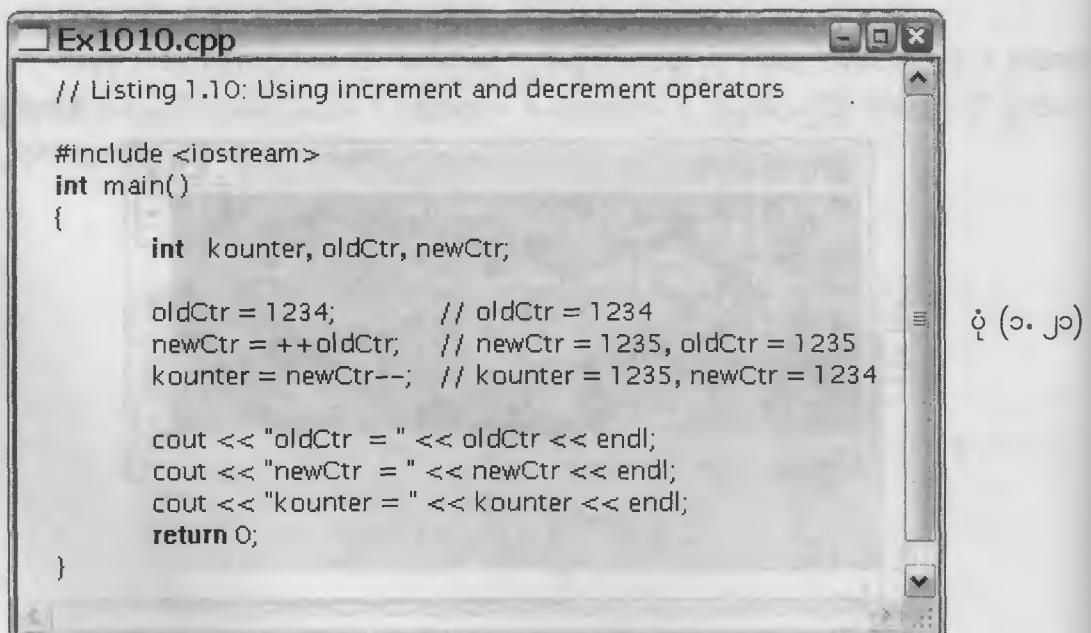
Increment and Decrement Operators

C++ မှာ ပါဝင်တဲ့ Unary operator တွေကို ယေး (၁.၆) မှာဖော်ပြထားပါတယ်။ ကျွန်တော်တို့ ဒီ operator တွေကိုအသုံးပြုပြီး variable တွေရဲ့တန်ဖိုးတွေကို အတိုးအလျော့လှပ်ပေးလို့ရပါတယ်။

ယေး (၁.၆) **Unary Operators**

Symbol	Description
++	Increment operator
--	Decrement operator

၁။ ကျွန်တော်တို့ ပုံ (၁.၂၁) မှာဖော်ပြထားတဲ့ Ex1010.cpp program ကို run လိုက်မယ်ဆိုရင် Unary operator တွေအကြောင်းကိုသိရပါလိမ့်မယ်၊ လေ့လာကြည့်ပါ။



The screenshot shows a code editor window titled "Ex1010.cpp". The code is as follows:

```
// Listing 1.10: Using increment and decrement operators

#include <iostream>
int main()
{
    int kounter, oldCtr, newCtr;

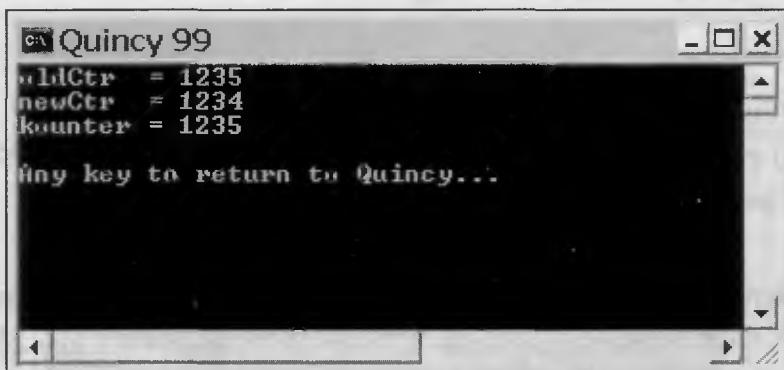
    oldCtr = 1234;           // oldCtr = 1234
    newCtr = ++oldCtr;       // newCtr = 1235, oldCtr = 1235
    kounter = newCtr--;     // kounter = 1235, newCtr = 1234

    cout << "oldCtr = " << oldCtr << endl;
    cout << "newCtr = " << newCtr << endl;
    cout << "kounter = " << kounter << endl;
    return 0;
}
```

ပုံ (၁.၂၁)

program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- စစချင်း program မှာအသုံးပြုမယ့် kounter : oldCtr နဲ့ newCtr ဆိုတဲ့ identifier (3) ခဲ့တဲ့ type int လို့ကြောပါတယ်။ နောက်တစ်ကြောင်းမှာ oldCtr ကို 1234 လို့ assign လုပ်ပေးပါတယ်။
- oldCtr တန်ဖိုးကို increment အရင်လုပ်ပေးပြီး newCtr နဲ့ညီပေးပါတယ်။ ဒါတော့ newCtr = 1235 နဲ့ oldCtr = 1235 ဖြစ်သွားပါပြီ။
- ဒါတစ်ခု kounter တန်ဖိုးကို newCtr = 1235 နဲ့အရင် assign လုပ်ပြီး newCtr တန်ဖိုးကို decrement လုပ်ပါတယ်။ ဒါတော့ newCtr = 1234 ပြန်ဖြစ်သွားပါပြီ။
- နောက်ခုံး statement (3) ခုကနေ oldCtr ၊ newCtr နဲ့ kounter တန်ဖိုးတွေကို တစ်ခုချင်း display လုပ်ပြသွားပါလိမ့်မယ်။ အဖြေကို ဗုံး (o..jj) မှာကြည့်ပါ။



ဗုံး (o..jj)

Compound Assignment Operators

C++ မှာ ပါဝင်တဲ့ += ၊ -= ၊ /= ၊ %= ၊ <=< အစရိတဲ့ Compound assignment operator တွေကို ယော် (o..j) မှာဖော်ပြထားပါတယ်။ ဒါ operatorတွေအသုံးပြုပြီး statement တွေကို short form ဖြစ်အောင်ပြောင်းရေးလို့ရပါတယ်။ ဥပမာ sum= sum + counts; ကို sum += counts; လို့ရေးလို့ရပါတယ်။

အသာ (၁၁၃) Compound Assignment Operators

Symbol	Description
$+=$	Addition assignment
$-=$	Subtraction assignment
$*=$	Multiplication assignment
$/=$	Division assignment
$\%=$	Modulus assignment
$<<=$	Shift left assignment
$>>=$	Shift right assignment
$\&=$	Bitwise AND assignment
$ =$	Bitwise OR assignment
$^=$	Bitwise exclusive OR assignment

၁။ ပုံ (၁၁၃) မှာဖော်ပြထားတဲ့ program ဟာဆိုရင် multiplication assignment operator ကိုအပြဖိုး ပေးထားတဲ့ကိန်းတစ်ခု၊ factorial ကိုရှာပေးတဲ့ program ပါပဲ၊ လေ့လာကြည့်ပါ။

၂။ ဒီ program ကို run လိုက်မယ်ဆိုရင် ကွန်ပျူးတာမှာ Enter a number : လို့ prompt တစ်ခုပေါ်လိမ့်မယ်။ ကွန်တော်တိုက keyboard ကနေ ဝက်နဲ့ 10 ကိုရိုက်ထည့်ပေးပြီး ENTER key နှင့်လိုက်တာနဲ့ တန်ဖိုးကို တွက်ပေးလိုက်ပါပြီ။ ပုံ (၁၁၄) ကိုကြည့်ပါ။

၃။ Ex1011.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

■ စစချင်း program မှာအသုံးပြုမယ့် identifiers i နဲ့ num တို့ကို type int လိုက်ကြဖို့ fac ကို long int လိုက်ကြဖို့ပါတယ်။ i က counter ပါ။ num က ကွန်တော်တို့ factors ရှာပေးချင်တဲ့ကိန်းပါ။ fac က num! ကို store လုပ်မယ့် identifier ဖြစ်ပါတယ်။

Ex1011.cpp

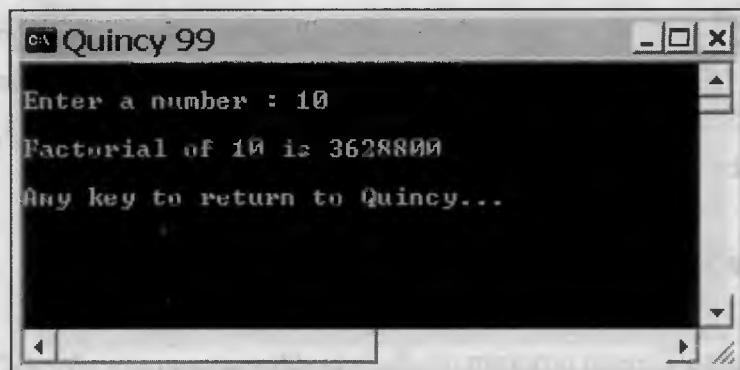
```
// Listing 1.11: Using a compound operator

#include <iostream>
int main()
{
    int    i, num;
    long   int fac;

    cout << "\nEnter a number : ";
    cin >> num;
    fac = 1;
    for (i=2; i <= num; i++)
        fac *= i;
    cout << "\nFactorial of " << num << " is " << fac << endl;

    return 0;
}
```

ଓ (০. জু)



ଓ (০. জু)

- cout << "\nEnter a number : "; হ্যাতে statement কফি prompt কর্তৃতে পেলাবে: ১০
keyboard কফি 10 কৰিগৰ কল্পনা পিষিএল। তিহ্যের্ণ num = 10 লে cin >> num; কফি assign কৰিবে: লিঙ্ক পিষিএল।

- for loop ထဲမှာ counter ကို 2 ကနေ 10 အထိပတ်ပေးပြီး num! ကိုအခုလိုတွက်ပေးပြီး

```

fac *= i ► fac = fac * i
2! = 1 * 2 = 2           i = 2
3! = 2 * 3 = 6           i = 3
4! = 6 * 4 = 24          i = 4
5! = 24 * 5 = 120         i = 5
6! = 120 * 6 = 720        i = 6
7! = 720 * 7 = 5040       i = 7
8! = 5040 * 8 = 40320      i = 8
9! = 40320 * 9 = 362880     i = 9
10! = 362880 *10 = 3628800    i = 10

```

Conditional Operators

- ၁။ conditional operator တစ်ခုဟာဆိုရင် program ထဲက ညီမှုခြင်းရဲ့လာဘက်က expression စံ test လုပ်ပြီး true ဖြစ်ခဲ့ရင် ? ရဲ့နောက်က expression တန်ဖိုးကို ညီမှုခြင်းရဲ့ဘယ်ဘက်က identifier က assign လုပ်ပေးပါလိမ့်မယ်။ false ဆိုရင် : နောက်က expression တန်ဖိုးနှင့်ညီပေးမှုပါ။ ဒီဥစ္စာရဲ့ format စံ အခုလို အလွယ်မှုတေသားလိုရပါတယ်။

variable = <test> ? expr1 : expr2;

- ၂။ ပုံ (၁.၂၅) မှာဖော်ပြထားတဲ့ program ဟာဆိုရင် conditional operator ကိုအသုံးပြုပြီး ပေးထားတဲ့ကိုနဲ့
(3) ခုထဲက အကြော်းဆုံးကိန်းကိုရှာပေးတဲ့ program ပါပဲ၊ လေ့လာကြည့်ပါ။

- ၃။ ဒီ program ကို run လိုက်မယ်ဆိုရင် ကွန်ပျူတာမှာ Enter three numbers : လို prompt တစ်ခုပေါ်လာမှုပါ။ keyboard ကနေ 150.56 , -500 နဲ့ 2500 တိုကိုရှိက်ထည့်ပေးပြီး ENTER key နှိပ်လိုက်တာနဲ့ Largest number is 2500 ကို ကွန်ပျူတာမှာ display လုပ်ပြပေးပါလိမ့်မယ်။ ပုံ (၁.၂၆) ကိုကြည့်ပါ။

```
// Listing 1.12: Using the conditional operator

#include <iostream>
int main()
{
    float num1, num2, num3, largst;

    cout << "\nEnter three numbers\n";
    cin >> num1 >> num2 >> num3;

    largst = num1 > num2 ? num1 : num2;
    largst = largst > num3 ? largst : num3;
    cout << "\nLargest number is " << largst << endl;
    return 0;
}
```

ଓ (০. জে)

```
Quincy 99
Enter three numbers
150.56
-500
2500
Largest number is 2500
Any key to return to Quincy...
```

ଓ (০. জে)

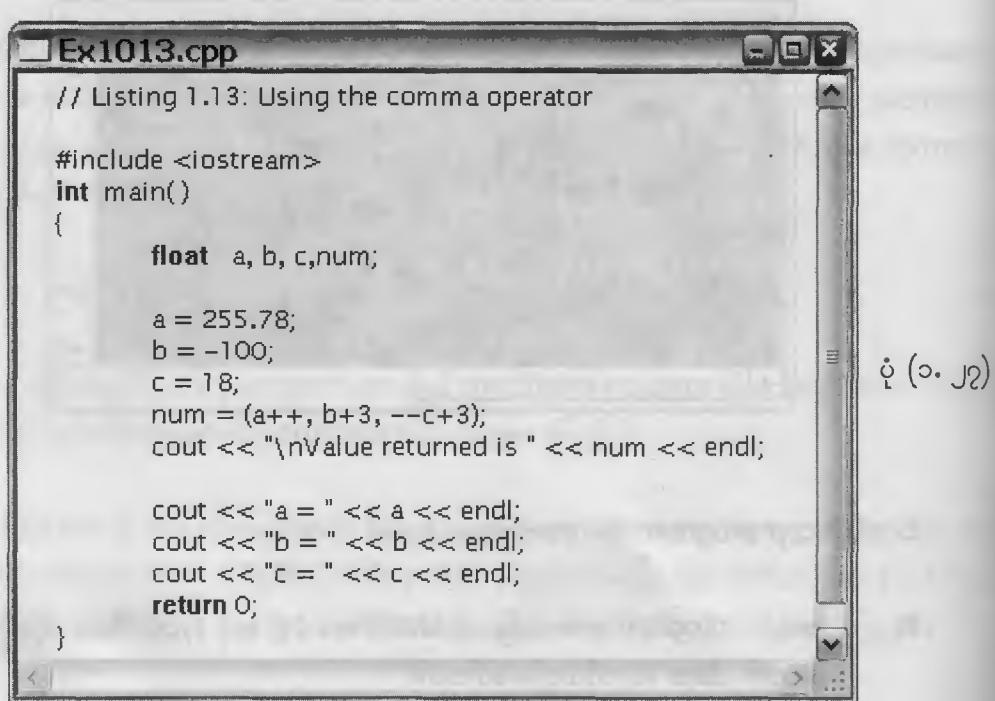
Ex1012.cpp program কৰিলে লাভ্যন্ত ময়দার্দ

- সূচী: program মুজা অবস্থা: প্রিমিয়ার identifiers (4) একটি type float লিঙ্গেটিপ্রিম: যুক্তি আত্মক data টেক্সট তান্ত্রিকভাবে প্রিমিয়ার।

- `largst = num1 > num2 ? num1 : num2 ;` ဆိတ် statement ကနေ `largst = 150.56 > -500 ? 150.56 : -500 = <true> ? 150.56 : -500 = 150.56` လို့ assign လုပ်ပေးမှာပါ။
- အခြာစ်ခါမှာတော့ `largst = largst > num3 ? largst : num3 ;` ဆိတ် statement ကနေ `largst = 150.56 > 2500 ? 150.56 : 2500 = <false> ? 150.56 : 2500 = 2500` လို့ assign လုပ်ပေးမှာပါ။ ဒါကြောင့်မို့ ကွန်ပျူးတာမှာ Largest number is 2500 ကို display လုပ်ပြတာပါပဲ။

Comma Operators

ၦ။ C နဲ့ C++ program တွေမှာ expression တွေကို comma (,) တွေနဲ့ခြားပေးလိုရပါတယ်။ ဥပမာ `var1 = (expr1, expr2, expr3);` လိုပေးမယ်ဆိုရင် expression တစ်ခုချင်းရဲ့ တန်ဖိုးတွေကိုတွက်ပေးသွားပြီး ညာအစွန်ဆုံးက expression ရဲ့တန်ဖိုးကို var1 နဲ့ assign လုပ်ပေးသွားမှာဖြစ်ပါတယ်။ ပုံ (၁။၂၃) မှာဖော်ပြထားတဲ့



```
// Listing 1.13: Using the comma operator

#include <iostream>
int main()
{
    float a, b, c, num;

    a = 255.78;
    b = -100;
    c = 18;
    num = (a++, b+3, --c+3);
    cout << "\nValue returned is " << num << endl;

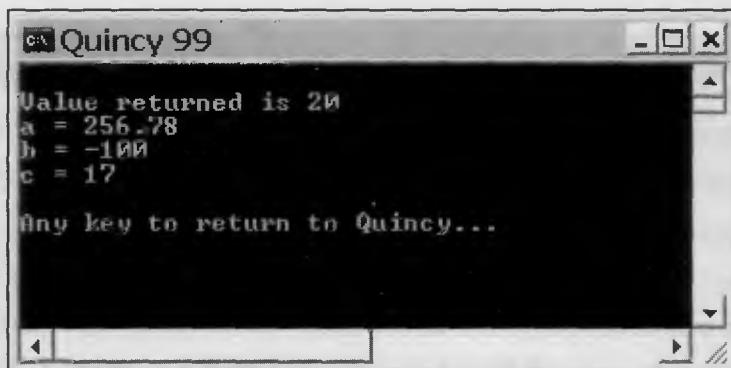
    cout << "a = " << a << endl;
    cout << "b = " << b << endl;
    cout << "c = " << c << endl;
    return 0;
}
```

ပုံ (၁။၂၃)

program ဘာဆိုရင် comma operator ကိုအသုံးပြုပြီး ပေးထားတဲ့ကိန်း (3) ခုရဲ့တန်ဖိုးတွေကို အသစ်ပြင်ပေးတဲ့ program ပါပဲ။ လေ့လာကြည့်ပါ။

Ex1013.cpp program ကို run လိုက်မယ်ဆိုရင် ကွန်ပျူတာမှာ ပုံ (၁၂၂) မှာပြထားတဲ့အတိုင်းပေါ်လာ မှာပါပဲ။ ဒါ ပြုခဲ့ပါ။ ဒါ ပြုခဲ့ပါ။

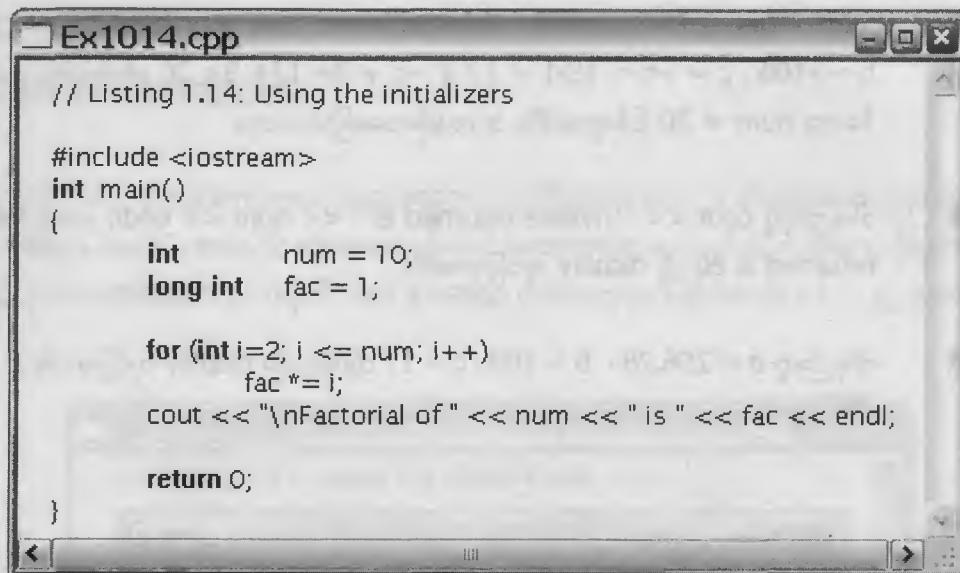
- စေချင်း program မှာအသုံးပြုမယ့် identifiers (4) ခုကို type float လိုကြော်ပြီး သူတို့ အတွက် data တွေကို a = 255.78 , b = -100 , c = 18 လို့ assign လုပ်ပေးပါတယ်။
- num = (a++, b+3, --c+3); ဆိုတဲ့ statement ကနေ a = 255.78 + 1 = 256.78 , b = -100 , c = --c = 18 - 1 = 17 နဲ့ --c + 3 = 17 + 3 = 20 တို့ကိုတွက်ပေးမှာပါ။ ဒါတော့ num = 20 ဖြစ်သွားပါပြီ။ b တန်ဖိုးကမပြောင်းပါဘူး။
- ဒါကြောင့်မို့ cout << "Value returned is " << num << endl; ကနေ Value returned is 20 လို့ display လုပ်ပြတာပါပဲ။
- ထိန်ည်းတဲ့ a = 256.78 , b = 100 , c = 17 တို့ကိုလည်း display လုပ်ပြမှာပါ။ ပုံ (၁၂၃) ကိုကြည့်ပါ။



ပုံ (၁၂၃)

Initializers

ကျွန်တော်ဟို ရှေ့ပိုင်းမှာတူန်းက program မှာပါတဲ့ variable တွေကို type declare လုပ်ပြီးမှ သူတို့ initial value တွေကို assign လုပ်ပေးခဲ့ပါတယ်။ စင်စစ် type declaration နဲ့အတူ data initialization ကို လည်းလည်လို့ရပါတယ်။ ပုံ (၁၂၃) မှာဖော်ပြုခဲ့တဲ့ Ex1011.cpp program ကို ပုံ (၁၂၄) အတိုင်းရအောင်ပြင်ရေးပြီး run လိုက်မယ်ဆိုရင် ပုံ (၁၂၅) ကအဖြေမျိုးပဲရမှာပါ။ ဒါဆိုရင် C++ နဲ့ပတ်သက်တဲ့အခြားတွေကို စာဖတ်သူအနေနဲ့ စုစုပေါင်လင်လေး သိရှိထွားပြီလို့ထင်ပါတယ်။



```
// Listing 1.14: Using the initializers

#include <iostream>
int main()
{
    int      num = 10;
    long int fac = 1;

    for (int i=2; i <= num; i++)
        fac *= i;
    cout << "\nFactorial of " << num << " is " << fac << endl;

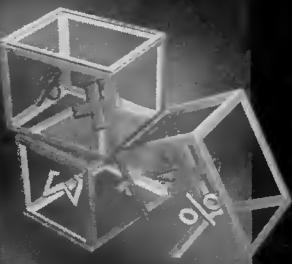
    return 0;
}
```

ပုံ (၁၃၂)

Chapter 2

FUNCTIONS

C++ Complete



Complete

C++ function တွေဟာ C++ program တစ်ခုကိုတည်ဆောက်ရာမှာ အရေးပါပြီးအခြေခံဘုတေသနတွေပါပဲ။ function တစ်ခုချင်းဟာ executable program code တွေကိုလက်ခံထားပါ၏။ C++ program တစ်ခုမှာ အနည်းဆုံး function တစ်ခုတော့ ပါရပါမယ်။ အဲဒါဟာ program အတွက် အပိုအတွက်ပေါက်နဲ့ ထွက်ပေါက်တွေပါဝင်တဲ့ main() function ပါပဲ။ main() function ကနေတစ်ခြား lower-level function တွေကို cal ခေါ်ပြီး အဒီ function တွေထက် code တွေကို execute လုပ်မှာပါ။ အဲဒီချိန်မှာ calling function ဟာအလုပ်ရပ်ထားဖြီး call ခေါ်ခဲ့ရတဲ့ function က return ပြန်လာမယ့် value တစ်ခုကိုစောင့်တယ်လေ။ main() function ကနေ call ခေါ်သလိုပဲ lower-level function တစ်ခုကာနေ နောက်တစ်ခုကို call ခေါ်တာလည်းရှုပါတယ်။ main() function ကိုတော့ call ပြန်ခေါ်လို့မရပါဘူး။ function တစ်ခုဟာ call ခေါ်ခဲ့ရတဲ့အချိန်မှာ အပြင်က argument တွေ pass ဝင်လာတာကိုလက်ခံနိုင်သလို ပြန်အတွက်မှာ value တစ်ခုကို return လုပ်ပေးနိုင်ပါတယ်။ function တွေကိုအသုံးပြုလို့ အကျိုးရှိရဲ့လားမေးရင် အကျိုးအများကြီးရှိတယ်လို့ ပြုပါမယ်။ ဥပမာ main() function လိုမျိုး calling function တွေမှာ ထပ်ခါတစ်လဲ လုပ်ဆောင်ရမယ့်

အလုပ်တွေကို calling function ထဲမှာ ရှုပ်အောင်ရေးစရာမလိုတော့ပါဘူး။ new function တစ်ခု သပ်သပ်ရေးပြီ အဲဒီကိုဘွားလိုက်ပြန်လိုက် လုပ်ခိုင်းလိုရန်ပြီမဟုတ်လား။ ဒါဆိုရင် calling function တွေမှာ code တွေရှင်းလင် သွားသလို C++ programming technique ဟာလည်း အစွမ်းထက်လာမှာပါပဲ။

J.2 Create a Simple Function

၁။ function တစ်ခုမှာ အမိကပါဝင်တဲ့အပိုင်း (2) ခုဟာ function header နဲ့ statement body ထို့ဖြစ်ပါတယ်။ function header ကိုထပ်မံစိစစ်ကြည့်ရင် အပိုင်း (3) ခုပါဝင်နေတာကိုတွေ့ရမှာပါ။ (၁) return type (၂) function name နဲ့ (၃) parameter list တို့ဖြစ်ကြပါတယ်။ return value ဟာ C++ data type တွေထက် တစ်ခုခုဖြစ်နိုင်သလို data type တွေကို point လုပ်နေတဲ့ pointer တစ်ခုလည်းဖြစ်နိုင်ပါတယ်။ ဒါမှာမဟုတ် structure၊ array တွေကို point လုပ်နေတဲ့ pointer မျိုးလည်းဖြစ်လိုရပါတယ်။ function name ကတော့ identifier တွေကို နာမည်ပေးနည်းနဲ့အတူတူပါပဲ။ parameter list ဟာ blank ဖြစ်နိုင်သလို variable တွေအများကြီးရှိနေလိုလည်းရပါတယ်။ ဒီ variable တွေထက်က calling function က argument တွေကိုကော်ပို့ထည့်မှာပါ။ statement body ကို program တစ်ခုရေးတဲ့အတိုင်းရေးရပါမယ်။ local variable declaration တွေ၊ executable code တွေပါဝင်ပါဝါမယ်။ ကျွန်တော်တို့ function တစ်ခုကို call မခေါ်ခဲ့ function parameter type တွေနဲ့ return value အမျိုးအစားတွေဟာ ဘာတွေပါလို compiler ကို အရင်အသိပေးရပါမယ်။ အဲဒီလို function declaration ကို prototype လိုခေါ်ပါတယ်။ ကောင်းပြီ၊ လွယ်ကူတဲ့ function တစ်ခု define လုပ်ပြီး main() function ကနေ call ခေါ်ကြည့်ရအောင်။ ပုံ (J. ၁) က Ex201.cpp program လေ့လာကြည့်ပါ။

၂။ Ex201.cpp program ကို run လိုက်မယ်ဆိုရင် ကွန်ပျူတာမှာ ပုံ (J. J) မှာပြထားတဲ့အတိုင်းပေါ်မှာပါပဲ။ ဒီ program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- စစချင်းမှာ main() function ရဲ့ထိပ်ပိုင်းမှာ func နာမည်နဲ့ functionprototype ကို ပြောပါတယ်။ return type က void ဖြစ်ပြီး parameter list မှာ variable ဘာမှုမရှိပါဘူး။ prototype declaration နောက်ဆုံးမှာ semicolon (;) နဲ့ပို့ပေးရပါမယ်။
- main() function ထဲမှာ အရင်ဆုံးလုပ်တာက STEP I: Inside the main() function ဆိုတဲ့အကြောင်းကို display လုပ်တာပါပဲ။ ဒါတာ ပုံ (J. J) က ပထမစာကြောင်းပေါ့။

Ex201.cpp

```
// Listing 2.1: Create a simple function

#include <iostream>
void func(); // declare the function prototype

int main()
{
    cout << "\nSTEP I: Inside the main() function\n";

    // call to the function
    func();
    cout << "\nSTEP III: Back in the main() function again\n\n";

    return 0;
}

void func()
{
    // function body
    cout << "\nSTEP II: Now inside the func() function\n";
}
```

♂ (J. J)

Quincy 99

```
STEP I: Inside the main() function
STEP II: Now inside the func() function
STEP III: Back in the main() function again

Any key to return to Quincy...
```

♂ (J. J)

- func(); ကနေ function call လိုက်တဲ့အတွက် void func() ထဲကိုဝင်လာပါဖြို့။ calling function မှာ argument ထည့်ပေးမထားပါဘူး။ ဒီတော့ func() function ထဲကို တိုက်ရှိကြရောက်သွားပြီး STEP II: Now inside the func() function ဆိုတဲ့စာကြောင်းကို display လုပ်ပြုမှာပါ။ ဒါဟာ ပုံ (၂။၂) က ဒုတိယစာကြောင်းပေါ့။ function body ထဲမှာလုပ်စရာကုန် သွားရင် main() function ကိုပြန်လာမှာပါ။
- အခုတစ်ပါ main() function ထဲက func(); အောက်မှာရှိတဲ့ code statement ကိုဆက်ဖြုံးဖြေရှင်းပါလိမ့်မယ်။ ဒီတော့ STEP III: Back in the main() function ဆိုတဲ့စာကြောင်းကို display လုပ်ပြုးမှာပါ။ ဒါဟာ ပုံ (၂။၂) က တတိယစာကြောင်းပေါ့။ ဒါဆိုရင် Ex201.cpp program ကာ ပြီးသွားပါဖြို့။
- တစ်ကယ်လို့ func() function တစ်ခုလုံးကို main() function အပေါ်မှာကြိုးရေးထားရင် function prototype declaration မပါဝင်၍ program run လိုရပါတယ်။ ပုံ (၂။၃) ကိုဖြည့်၏

```
// Listing 2.1A: Create a simple function

#include <iostream>
void func()
{
    // function body
    cout << "\nSTEP II: Now inside the func() function\n";
}

int main()
{
    cout << "\nSTEP I: Inside the main() function\n";
    // call to the function
    func();
    cout << "\nSTEP III: Back in the main() function again\n\n";

    return 0;
}
```

ပုံ (၂။၃)

Printing a Line of Asterisks

- ဒီတစ်ခါတင်ပြုမှာက starline() ဆိတဲ့ function (void) တစ်ခုကို call ခေါ်ပြီး program output asterisk(*) တွေပေါ်လာအောင်လုပ်ခိုင်းမှာပါ။ ပုံ (၂၁၄၇) မှာရေးထားတဲ့ Ex202.cpp program ၏ code ဒုက္ခလေးလာကြည့်ပါ။

```
// Listing 2.2: This program demonstrates a simple function
// that prints a line of 40 asterisks.

#include <iostream>
void starline();
int main()
{
    cout << "Data Type           Range" << endl;
    starline();
    cout << "char    -128 to 127" << endl
        << "int     -32,768 to 32,767" << endl
        << "double  -2,147,483,648 to 2,147,483,647" << endl;
    starline();

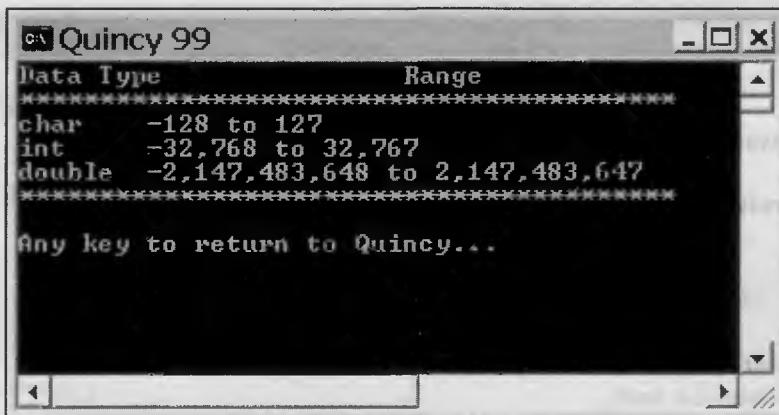
    return 0;
}

void starline()
{
    for (int i=0; i<=40; i++) cout << "*";
    cout << endl;
}
```

ပုံ (၂၁၄၇)

- Ex202.cpp program ကို run လိုက်မယ်ဆိုရင်ကွန်ပျူတာမှာ ပုံ (၂၁၄၇) မှာပြထားတဲ့အတိုင်းပေါ်လာမှာပါပဲ။ ဒီ program ကိုလေးလာကြည့်မယ်ဆိုရင်

- စစချင်း main() function ရဲအပေါ်မှာ starline နာမည့်နဲ့ function prototype တစ်ခု၏ ကြော်ပါတယ်။ return type က void ပါ။ parameter list မှာ ဘာမှုမရှိပါဘူး။ နောက်၌ Data Type <2 Tabs> Range ဆိုတဲ့ heading ကိုရေးခိုင်းပြီး starline() function ၏ call ခေါ်ပါတယ်။
- starline() function ထဲမှာ for loop ကိုအသုံးပြုပြီး ကြယ်ပွင့်အခါ (40) ကို print လုပ်ခိုင်ပါတယ်။ counter i ကို for loop ထဲမှာပဲ data type နဲ့တွဲပြုးကြောယာပါတယ်။ asterisk (40) ခုကို print လုပ်ပြီးရင် main() ကိုပြန်လာပါပြီ။ အခုတစ်ခါ char <2 Tabs> -128 to 127 ကိုရှိကြပြီး နောက်တစ်ကြောင်းကိုကူးပါတယ်။ endl နောက်မှာ semicolon(;) မရေးအပေါ်ကအတိုင်းဆက်ရေးရင် cout ကို ဆက်ပြီးအသုံးပြုမယ်ဆိုတဲ့သဘောပါပဲ။ ဒီတော့ int <2 Tabs> -32,768 to 32,767 ကိုဆက်ရှိက်မှာပါ။ ဒီအတိုင်းပဲ တတိယအကြောင်းကိုလည့် print လုပ်သွားမှာပါ။
- ဒုတိယအကြိမ် starline() function ကို call ခေါ်ပြီးတော့ print လုပ်ခိုင်းမယ်ဆိုရင် output မှာ asterisk (40) ခုဟာ ထပ်ပေါ်လာမှာပါပဲ။ ပုံ (J. ၅) ကိုကြည့်ပါ။



ပုံ (J. ၅)

J.J Pass the Variables by Value

၁။ calling program တစ်ခုကနေ function တစ်ခုကို call ခေါ်လိုက်တဲ့အခါမှာ argument တစ်ချို့ကို called function ထဲက variable တွေထက် pass လုပ်ပေးပါတယ်။ data pass လုပ်နည်းတွေအမှားကြီးရှိပါ

ဘယ်။ အမိန့်နည်း (2) ခုက Passing by value နဲ့ Passing by reference နည်းတွေဖြစ်ပါတယ်။ ပုံ (၂၆) ဘရေးထားတဲ့ Ex203.cpp program မှာ constant value တွေ pass လုပ်နည်းကို တင်ပြထားပါတယ်။

```
// Listing 2.3: Calculate the area of a triangle

#include <iostream>
float calArea(float, float);

int main()
{
    float area = calArea(10,20);
    cout << "\nBase = " << 10;
    cout << "\nHeight = " << 20;
    cout << "\nArea of the triangle = " << area << endl;
    return 0;
}

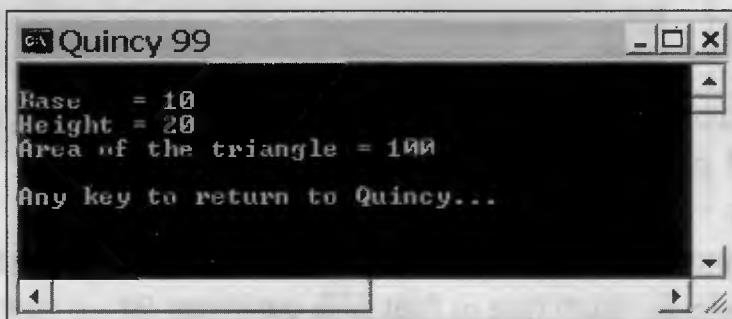
float calArea(float b, float h)
{
    float area = 0.5*b*h;
    return area;
}
```

ပုံ (၂၆)

JII Ex203.cpp program ကို run လိုက်မယ်ဆိုရင် ကွန်ပျူးတာမှာ ပုံ (၂၇) မှာပြထားတဲ့အတိုင်းပေါ်လာမှုပါပဲ။ ဒီ program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- program စစ်ချင်း main() function ရဲ့အပေါ်မှာ calArea နာမည်နဲ့ prototype တို့ကြော်ပါတယ်။ return type သဲ float ဖြစ်ပြီး parameter list မှာ float data(2) ခုကို pass လုပ်ခိုင်းမှုပါ။
- main() function ထဲမှာ အရင်ဆုံးလုပ်တာက identifier area ကို float လို့ declare လုပ်ပြီး calArea() function ကို call ခေါ်ပါတယ်။ pass လုပ်ပေးမယ့် argument တွေက constant value တွေပါပဲ။ 10 က ဖြို့ဂုံးတစ်ခုရဲ့ base ဖြစ်ပြီး 20 က height ဆိုပါမို့။

- pass လုပ်ယူသွားတဲ့ constant argument (2) ခုတဲ့ function calArea() ထဲက b နဲ့ h ဆိုတဲ့ variable တွေထဲမှာ အသီးသီး copy ထည့်ပေးလိုက်ပါတယ်။ ဒီတော့ b = 10 နဲ့ h = 20 ဖြစ်သွားပါပြီ။ ဒီနေရာမှာ သတိထားရမှာက prototype declaration နဲ့ function header တို့ဟာတူညီပါမယ်။ အခုလို data pass လုပ်တာကို passing by value လိုပါတယ်။
- အခုခိုရင် function body ထဲရောက်နေပါပြီ။ local variable ဖြစ်တဲ့ area ခဲ့ data type ကို area လိုအပ်ကြမြာဖြီး ဖြိုဂံချိယာကို $(base * height)/2$ ဆိုတဲ့ပုံသေနည်းနဲ့တွက်ပါတယ်။ float area = 0.5 * b * h; ဆိုတဲ့ statement ဘာ အဲဒီအတွက် ရေးထားတာပဲ area ကိုတွက်ပြီးသွားရင် return area; ကနေ function return value အဖြစ် area တဲ့ return လုပ်ပေးပါလိမ့်မယ်။ ဒီတော့ area ခဲ့ data type နဲ့ return value data type တို့ဟာလည်း တူရမှာပေါ့။
- main() function ကိုပြန်ရောက်လာတဲ့အခါမှာ area= calArea(10,20)= return value ထဲ့ assign လုပ်ပေးမှာဖြစ်ပါတယ်။ main() function ထဲက local variable: area ထဲမှာ ဖြိုဂံချိယာတန်ဖိုးက ရောက်နေပါပြီ။ နောက်ဆုံးအနေနဲ့ cout << expression ကိုအသုံးပြု၍ base + height နဲ့ area တို့ရဲ့အဖွဲ့ကိုကွန်ပြု၍တာမှာ display လုပ်ပေးလိုက်ပြုဖြစ်ပါတယ်။ ပုံ (J. 2) ကိုကြည့်ပါ။



ပုံ (J. 2)

- Ex204.cpp ဟာဆိုရင် Ex203.cpp ကို အသစ်ပြင်ရေးထားတာပါ။ variable တွေကို pass လုပ်နည်းနဲ့ program ကိုပြင်ရေးထားပါတယ်။ ပုံ (J. 1) ကိုကြည့်ပါ။ ဒီ program ကိုလေ့လာကြည့်မယ်ဆိုရင်

အပေါ်ဆုံး statement ဟာ base , height , area တိုကို global variable ထွေလို့ declare လုပ်ဖြီး prototype (3) ခုကို define လုပ်ပေးထားပါတယ်။ main() ထဲမှာပထမဆုံး getData() function ကို call ခေါ်ပါတယ်။ getData() function ထဲမှာ base နဲ့ height အတွက် တန်ဖိုးတွေကိုရှိက်ထည့်

```
// Listing 2.4: Passing variables by value
#include <iostream>

float base, height, area;
void getData();
void calArea();
void printArea();

int main()
{
    getData();
    calArea();
    printArea();

    return 0;
}

void getData()
{
    cout << "\nEnter base and height\n";
    cin >> base >> height;
}

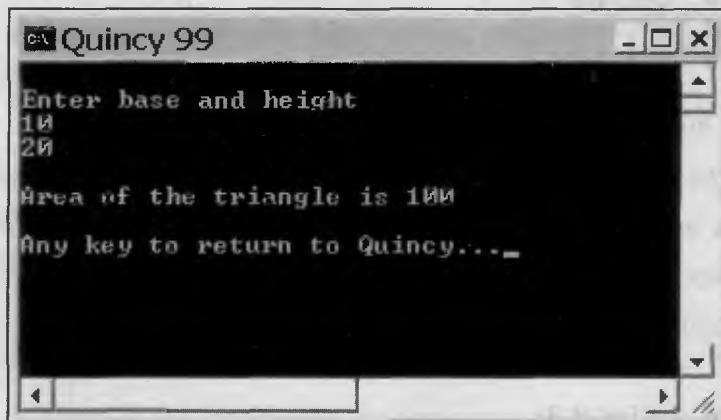
void calArea()
{
    area = 0.5*base*height;
}

void printArea()
{
    cout << "\nArea of the triangle is " << area << endl;
}
```

ဗျူး (J. ၁)

ပြီးရင် main() ကိုပြန်ရောက်လာပြီး calArea() function ကို call ခေါ်မှာပါ။ calArea() function ထဲရောက်လာတဲ့အခါကျရင် ဖြိုဂံရိယာကို ပုံသေနည်းနဲ့တွက်ယူပါမယ်။ base နဲ့ height တန်ဖိုးတွက သိနေပြီးမို့ function ထဲမှာပြောပြုစရာမလိုတော့ပါဘူး။ area တွက်ပြီးရင် main() ကိုပြန်ရောက်လာပြီး printArea() ကို call ဆက်ခေါ်ပါတယ်။ printArea() function က area တန်ဖိုးကို print လုပ်ပြပါလိမ့်မယ်။

JII Ex204.cpp program ကို run လိုက်မယ်ဆိုရင် ကွန်ပျော်ဘာမှာ ပဲ (J. ၉) မှာပြထားတဲ့အတိုင်းပေါ်လာမှာပါ။ အဖြေက Ex203.cpp program နဲ့အတူတူပါပဲ။



ပဲ (J. ၉)

Creating a Self-defined Square Root Function

၁။ အနုပည်ပြုသူ၏ program တာဆိုရင် sqroot() ဆိုတဲ့ 10 ကို self-defined function တစ်ခုကို အသုပ္ပါဒ် အသုပ္ပါဒ်ခဲ့ရဲ့ နှစ်ထပ်ကိန်းရင်းကိုရောပေးမှာပါ။ function အလုပ်လုပ်ပုံက calling function ကနေ constant (၂) ကို called function ထဲ pass လုပ်ပေးလိုက်ပြီး အဲဒီမှာတွက်တာချက်တာလုပ်ပြီးတာနဲ့ အဖြေကိုလည်း print ထုတေသနအဲပါတယ်။ အဖြန်မှာ return value မယူပဲ main() ကိုပြန်လာတာပါ။ ဒီသဘောတရားအတွက် Ex205.cpp program ကိုရေးထားတာဖြစ်ပါတယ်။ ပဲ (J. ၁၀) မှာရေးထားတဲ့ Ex205.cpp program က အသုပ္ပါဒ် လေ့လာပြည့်ပါ။

```

// Listing 2.5: This program displays the square root of
// a number passed from a program to the function.

#include <iostream>
void sqroot (float , float );

int main()
{
    sqroot (625,10000);

    return 0;
}

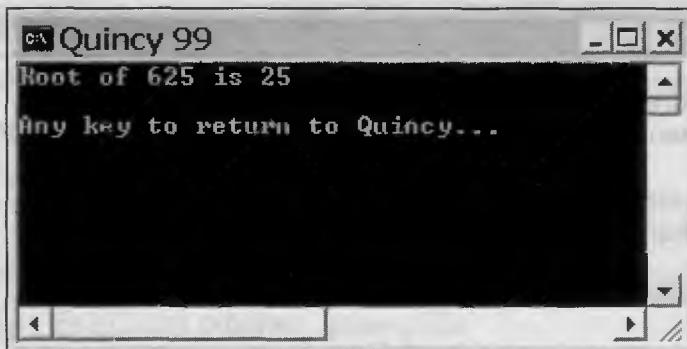
void sqroot (float xn, float xr)
{
    for (int i=1; i<=50; ++i)
        xr = (xr+xn/xr)/2;
    cout << "Root of " << xn << " is " << xr << endl;
}

```

ပုံ (၂၁)

၂။ Ex205.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- program စာခင်: main() function ရဲ့အပေါ်မှာ sqroot နာမည့်နဲ့ prototype ကိုဖော်ပြုပါတယ်။ return type က void ဖြစ်ပြီး parameter list မှာ float data(2) ခုကို pass လုပ်ခိုင်းမှာပါ။ ကောင်းပြီး main() function မှာ sqroot (625, 10000) ကို call ခေါ်ပြီး 625 နဲ့ 10000 ကို sqroot() function ထဲ pass လုပ်လိုက်ပါတယ်။ passing by value နည်းပေပါ။
- sqroot() function header မှာ xn = 625 နဲ့ xr = 10000 ဆိုပြီး constant တွေ pass လုပ်ဝင်သွားပါပြီ။ function body ထဲမှာ xn ရဲ့ square root တန်ဖိုးကိုရှာပါတယ်။ xr က ကျွန်ုတ်တို့ assume လုပ်တဲ့ root တန်ဖိုးပါ။ root ရှာတဲ့နည်းက xr = (xr+xn/xr)/2 ဆိုတဲ့ statement ကိုထပ်ခါတစ်လဲပြန်တွက်ခိုင်းတာပါပဲ။ for loop ကိုအကြိမ် (50) ပတ်ပြီးသွားရင်အဖြေရပါပြီ။ output result ကို ပုံ (၂၁) မှာဖော်ပြထားပါတယ်။



ဤ (J. oo)

၃။ ဒါ program မှာ ကွန်ပျော်တာကအဖြစ် ဖြည့်ဖြည့်ချင်း မှန်အောင်ခန့်မှန်းတွက်ချက်လာတာကို မြင်ချင်တယ်
ဆိရင် ဤ (J. oo) မှာပြထားတဲ့အတိုင်းပြင်ရေးပြီး run ကြည့်ပါ။

```
Ex205A.cpp

// Listing 2.5A: This program displays the square root of
// a number passed from a program to the function.

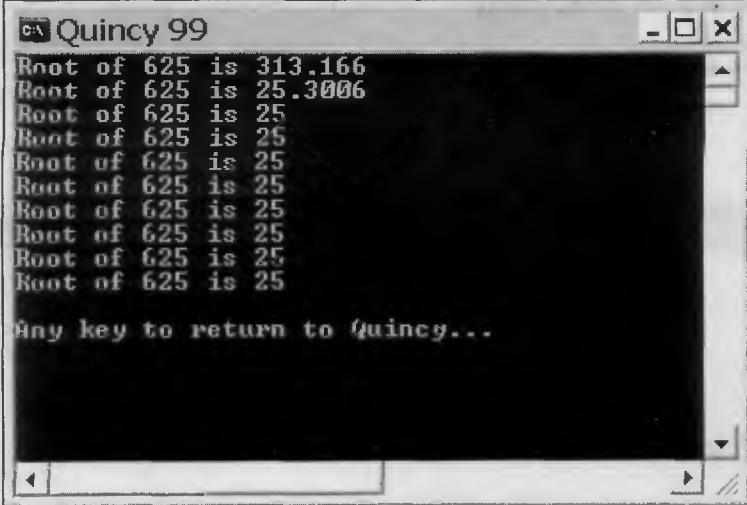
#include <iostream>
void sqroot (float, float);

int main()
{
    sqroot (625,10000);
    return 0;
}

void sqroot (float xn, float xr)
{
    for (int i=1; i<=50; i++)
    {
        xr = (xr+xn/xr)/2;
        if (i % 5 == 0)
            cout << "Root of " << xn << " is " << xr << endl;
    }
}
```

ဤ (J. oo)

- ၄။ ထူးခြားတာကိုဖွေ့တယ်မဟုတ်လား။ for loop ကိုအကြိမ် (50) ပတ်တွက်ခိုင်းပြီး ခန့်မှန်းအဖြေဖွေ့ကို
 (5) ကြိမ်တွက်ပြီးတိုင်း တစ်ကြိမ်နှင့်နဲ့ print လုပ်ခိုင်းထားတဲ့အတွက် ပုံ (၂၁၃) ကလိုမျိုး display မှာပေါ်နေတာ
 ပြစ်ပါတယ်။ စာဖတ်သူကိုယ်တိုင် program trace လုပ်ကြည့်ပါ။



```
Root of 625 is 313.166
Root of 625 is 25.3006
Root of 625 is 25

Any key to return to Quincy...
```

ပုံ (၂၁၃)

- ၅။ ဒီ program မှာ root ရှာချင်တဲ့ကန်းနဲ့ assumed value တို့ကို keyboard ကနေ အရှင်ထည့်ပေး
 နိုင်မယ်ဆိုရင် ဒီ program ကပိုကောင်းလာမှာပါ။ xn နဲ့ xr တို့ရဲ့တန်ဖိုးတွေကို variable တွေအနေနဲ့ pass လုပ်
 ပေးကြည့်ရအောင်။ ပုံ (၂၁၄) မှာရေးထားတဲ့ program code တွေကို ဘယ်လိုပြင်ရေးထားလဲဆိုတာကို သတိပြု
 ကြည့်စေချင်ပါတယ်။

- sqroot() function ရဲ့ return type ကို float လိုပြင်ရေးထားပါတယ်။ main() ထဲမှာ
 xn နဲ့ xr တန်ဖိုးတွေကို keyboard ကနေ ရှိနိုင်ထည့်ပေးနိုင်အောင်ပြောင်းရေးထားပါတယ်။
 အဖြေကို cout << နဲ့ sqroot(xn xr) တို့ကိုတွဲရေးပြီး print ထုတ်ခိုင်းထားပါတယ်။
- sqroot() function ထဲမှာတော့ root ရှာတဲ့နည်းသက်သက်တို့ပဲရေးထားပါတယ်။ တွက်ပြီးရင်
 အဖြေကို xr = sqroot(xn, xr) အနေနဲ့ return လုပ်ပေးမှာပါ။

- ၆။ ဒီ program ကို run လိုက်မယ်ဆိုရင် ပုံ (၂၁၅) မှာပြထားတဲ့အတိုင်း တွက်သွားလိုရပါတယ်။

Ex205B.cpp

```
// Listing 2.5B: This program displays the square root of
// a number passed from a program to the function.

#include <iostream>
float sqroot (float, float);

int main()
{
    float xn, xr;

    cout << "Enter a positive real number\n";
    cin >> xn;
    cout << "Assume its root\n";
    cin >> xr;
    cout << "Root of " << xn << " is " << sqroot(xn,xr) << endl;
    return 0;
}

float sqroot(float xn, float xr)
{
    for (int i=1; i<=50; i++)
        xr = (xr+xn/xr)/2;
    return xr;
}
```

Ø (J. o9)

Quincy 99

```
Enter a positive real number
123456789
Assume its root
111
Root of 1.23457e+08 is 11111.1
Any key to return to Quincy...
```

Ø (J. o9)

Pass Variables by Reference

C++ program တစ်ခုမှာ calling function ထဲက argument တစ်ခုကို called function ဆိတိ pass လုပ်တဲ့အခါမှာ argument-to-argument ကော်ပိလုပ်ခိုင်းတာမျိုးမဟုတ်ပဲ calling argument မှာပြောင်းလဲ ဘဏ္ဍာကို address မှာ တိုက်ရှိကြပြင်သွားအောင် argument-to-address pass လုပ်နည်းကို Passing by Reference လိုပေါ်ပါတယ်။ Ex203.cpp program ကို ဒီနည်းနဲ့ပြောင်းရေးကြည့်ရအောင်။ ပုံ (J. ၁၆) မှာရေးထားတဲ့ Ex206.cpp program ကိုလေ့လာကြည့်ပါ။

```
// Listing 2.6: Passing variables by reference

#include <iostream>
void calArea(float, float, float& );
void printArea(float);

int main()
{
    float base, height, area;

    cout << "\nEnter base and height\n";
    cin >> base >> height;
    calArea(base, height, area);
    printArea(area);
    return 0;
}

void calArea(float b, float h, float &x)
{
    x = 0.5*b*h;
}

void printArea(float a)
{
    cout << "\nArea of the triangle is " << a << endl;
}
```

ပုံ (J. ၁၆)

- အပေါ်ဆုံး statement (2) ခုဟာ function prototype declaration တွေပါ။ Ex203.cpp မှာကြော်နည်းနဲ့ မတူတော့ပါဘူး။ called function က return type တွေကို void တွေချည်းလုပ်ထားပါတယ်။ ဒီတော့ တွက်လိုချက်လိုရတဲ့အဖြေတွေကို function return value အနေနဲ့ return မလုပ်တာတော့သေချာပါပြီ။ calArea() function ထဲကနောက်ဆုံး argument ကို သတိပြုကြည့်ပါ။ ဒီဥစ္စာရွှေးရှိး variable မဟုတ်ပါဘူး။ calling function က argument ကို function မှာ address တစ်ခုနဲ့ pass လုပ်ပေးထားတယ်လေ။ ဒီတော့ address နေရာမှာ တစ်ခုခုပြင်လိုက်မယ်ဆိုရင် အဲဒီ address နဲ့ဆိုင်တဲ့ variable ရဲ့တန်ဖိုးဟာ program ရဲ့ဘယ်နေရာမှာပဲရှိရှိ ပြင်ပြီးသားဖြစ်သွားပါပြီ။
- main function ကိုပြန်ရောက်လာတဲ့အခါမှာ area တန်ဖိုးကို ကွန်ပျူးတာကသိနေပြီဖြစ်တာမျိုး assign လုပ်ပေးမှာဖြစ်ပါတယ်။ နောက်ဆုံးအနေနဲ့ printArea() function ကို call ခေါ်ပြီး function ထဲမှာ area တန်ဖိုးကို print လုပ်ပေးမှာပါပဲ။

Lowercase to Uppercase Character Conversion Program

ဗု (၂၁၇) မှာပြထားတဲ့ Ex207.cpp program ကို လေ့လာကြည့်မယ်ဆိုလိုရှိရင် function ကို call ခေါ်ကတည်းက data ယူသွားတာကို တွေ့ရပါတယ်။ data က lowercase letter 'e' ပါ။ lower_to_upper ဆိုတဲ့ function တဲ့မှာ passing by reference နည်းနဲ့ uppercase ရဲ့ address မှာ letter 'E' ဖြစ်အောင်ပြင်ပေးလိုက်ပါတယ်။ ပြောင်းတဲ့နည်းကတော့ $c2 = (c1 >='a' \&& c1 <= 'z') ? ('A'+c1-'a') : c1;$ statement ကို အသုံးပြုတာပါ။ ကျွန်တော်တိုက $c1 = 'e'$ အနေနဲ့ pass လုပ်ပေးလိုက်တဲ့အခါမှာ ($c1 >='a' \&& c1 <= 'z'$) expression နဲ့ test လုပ်လိုက်တဲ့အခါကျေရင် TRUE ဖြစ်တဲ့အတွက် $c2$ ကို ' A ' + $c1 - 'a'$ နဲ့ assign လုပ်ပေးပါလိမ့်မယ်။ char ' A ' ရဲ့ ASCII hex value က 41 ပါ။ $c1 = 'e'$ အတွက်က 65 ပါ။ ' a ' အတွက်က 61 ပါ။ ဒီတော့ $c2 = 'A' + c1 - 'a' = 41 + 65 - 61 = 45$ ကိုရပါတယ်။ ASCII hex value = 45 တာ ' E ' ဖြစ်တဲ့အတွက် character variable (upper) တာ ' E ' ဖြစ်သွားပါပြီ။ တစ်ကယ်လို့ $c1 = 'E'$ လိုထည့်ပေးမယ်ဆိုရင် $c2 = (c1 >='a' \&& c1 <= 'z') ? ('A'+c1-'a') : c1;$ statement မှာ $c2 = c1$ လို့ assign လုပ်ပေးမှာပါပဲ။ ဒီတော့ $c2 = c1 = 'E'$ ဖြစ်သွားပြီလေ။ ဗု (၂၁၈) မှာ program ကို run ပြထားပါတယ်။

Ex207.cpp

```
// Listing 2.7: This program reads a lowercase character,  
// converts it to uppercase and then writes out  
// the uppercase equivalent.  
  
#include <iostream>  
  
void lower_to_upper(char c1, char& c2)  
{  
    c2 = (c1 >='a' && c1 <= 'z') ? (A'+c1-'a') : c1;  
}  
  
int main()  
{  
    char lower, upper;  
  
    cout << "Enter a lowercase character\n";  
    cin >> lower;  
    lower_to_upper(lower, upper);  
    cout << "\nThe uppercase equivalent is " << upper << endl;  
    return 0;  
}
```

Quincy 99

```
Enter a lowercase character  
e  
The uppercase equivalent is E  
Any key to return to Quincy...
```

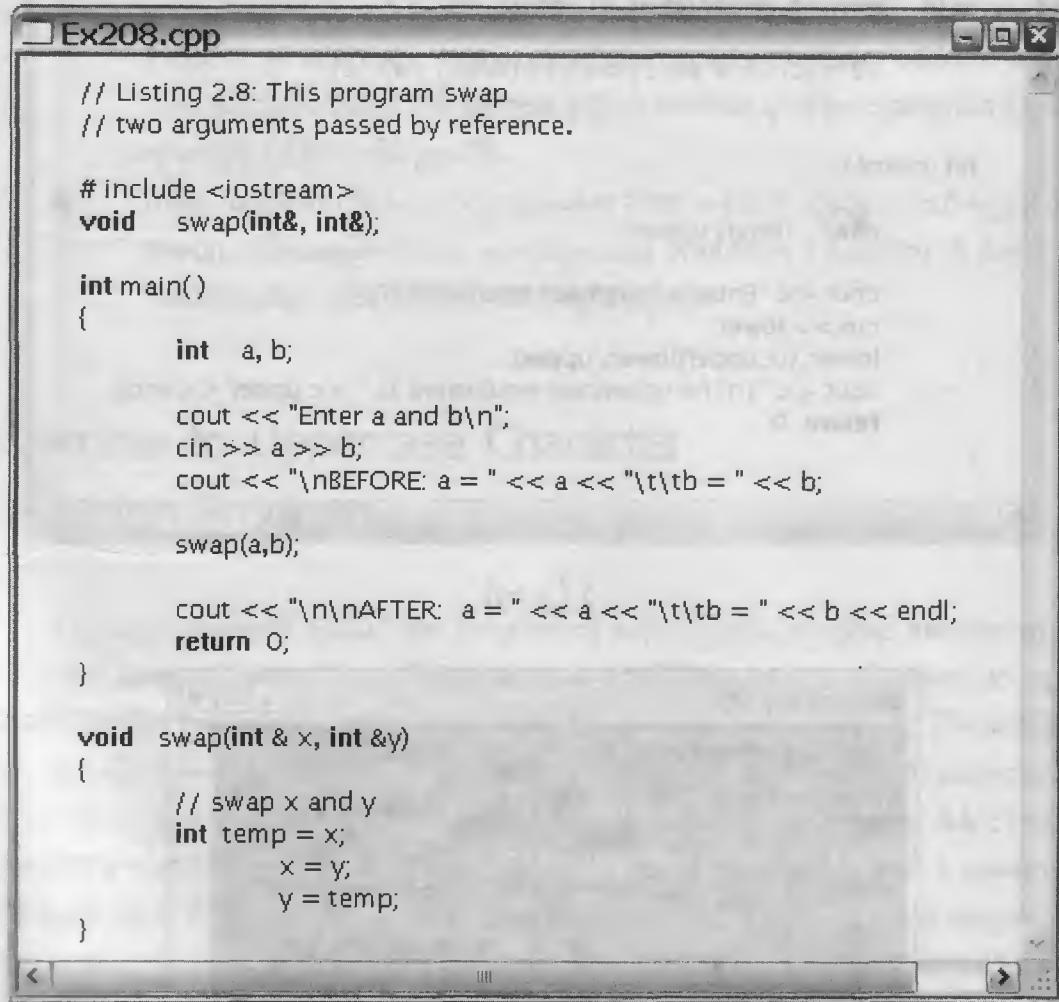
Ø (J. 02)

Go

Functions

Swap Variables by Reference

၁။ ဗုဒ္ဓ (၂၉) မှာပြထားတဲ့ Ex208.cpp program ဟာဆိုရင် passing by reference နည်းကိုအသုံးပြု၍ variable a နဲ့ b တို့၏ value တွေကိုနေရာချင်းလဲပေးပါတယ် । လေ့လာကြည့်ပါ။



```
// Listing 2.8: This program swap
// two arguments passed by reference.

#include <iostream>
void swap(int&, int&);

int main()
{
    int a, b;

    cout << "Enter a and b\n";
    cin >> a >> b;
    cout << "\nBEFORE: a = " << a << "\t\tb = " << b;

    swap(a,b);

    cout << "\n\nAFTER: a = " << a << "\t\tb = " << b << endl;
    return 0;
}

void swap(int &x, int &y)
{
    // swap x and y
    int temp = x;
    x = y;
    y = temp;
}
```

ဗုဒ္ဓ (၂၉)

Ex208.cpp program ကို ဗုဒ္ဓ (၂၉) မှာ run ပြထားပါတယ်။

```

Quincy 99
Enter a and b
15
56
BEFORE: a = 15          b = 56
AFTER:  a = 56          b = 15
Any key to return to Quincy...

```

ဃ (J. jo)

Default Function Arguments

function prototype တစ်ခုကို declare လုပ်တဲ့အခါမှာ parameter list မှာ ပါဝင်တဲ့ argument ထက ကြိုက်တဲ့အရေအတွက်ကို default value တွေအဖြစ်ထားပေးလို့ရပါတယ်။ function call ခေါ်လိုက်လို့ထားတဲ့ argument တွေကိုတွေ့ရင် compiler ကနေ default value တွေကိုအစားထည့်ပေးမှုပါ။ ဗုဒ္ဓရ (J. jo) မှာဖော်ပြထားတဲ့ program မှာဆိုရင် function prototype ထက argument တွေအတွက် default value တွေကို declare လုပ်ပြီး အသုံးပြုသွားပုံကို တင်ပြထားပါတယ်။

ဒါ program ကို လေ့လာကြည့်မယ်ဆိုရင်

- စေချင်: function prototype volume() ကို declare လုပ်တဲ့အခါမှာ argument တွေ အတွက် default value တွေကို 2 । 3 । 4 လို့ အသုံးသီး သတ်မှတ်ပေးပါတယ်။ ပထမဆုံး call ခေါ်တဲ့ volume() function မှာ argument လုံးဝမပါတဲ့အတွက် default value (3) ခု လုံးကို အသုံးပြုပြီး volume ကို $l*w*h=2*3*4=24$ လို့ တွေ့က်ပေးပါလိမ့်မယ်။
- ဒုတိယအကြိမ် call ခေါ်တဲ့အခါမှာ parameter list မှာ argument တစ်လုံးပဲ ပါတဲ့အတွက် ပထမဆုံး default value ကို new argument နဲ့ အစားပြောင်းပေးမှုပါ။ ဒီတော့ $volume = 5*3*4 = 60$ ဖြစ်မှုပါ။ ဒီနည်းအတိုင်းကျန်တဲ့ function တွေကို call ခေါ်ပြီးရင် volume ဟာ $5*7*4 = 140$ နဲ့ $5*7*9 = 315$ တို့ ဖြစ်ပါလိမ့်မယ်။ ဗုဒ္ဓရ (J. JJ) မှာ EX209.cpp ကို run ပြထားပါတယ် । လေ့လာကြည့်ပါ။

```
Ex209.cpp

// Listing 2.9: Using functions with default arguments

#include <iostream>
void volume(float=2, float=3, float=4);

int main()
{
    volume();      // all three arguments default
    volume(5);    // provide the first argument
    // provide the first and second arguments
    volume(5,7);
    volume(5,7,9); // provide all three arguments
    return 0;
}

void volume(float l, float w, float h)
{
    float vol = l*w*h;
    cout << "\nVolume = " << vol << endl;
}
```

Ø (J. Jo)

```
Quincy 99

Volume = 24
Volume = 60
Volume = 140
Volume = 315
Any key to return to Quincy...
```

Ø (J. JJ)

၂၅ Inline Functions

ကျွန်တော်တိ function တစ်ခုကို define လုပ်လိုက်တာနဲ့ ပုံမှန်ဆိုရင် compiler ကနေ အဲဒီ function execute လုပ်ဖို့အတွက် instruction အစုတစ်ခုကို create လုပ်ပေးပြီး memory မှာ သိမ်းထားလိုက်ပါမြို့။ function ကို call ခေါ်တိုင်း calling function ကနေ called function ရှိရာကို ခုန်ကျော်သွားပြီး တွက်တာချက်တာ လုပ်မှာပါ။ ပြီးရင် calling function ကိုပြန်လာမယ်ပေါ့။ function ကို (10) ကြိမ် call ခေါ်ရင် (10) ကြိမ်တိတော်တုန်ခေါက်ပြန် သွားနေမှာပါပဲ။ call ခေါ်တဲ့ function က သိပ်ငယ်နေတယ်ဆိုရင် execution time ထက် traverse time ကပိုနေပါလိမ့်မယ်။ function ကလည်းသိပ်ငယ်ပြီး call ခေါ်တဲ့အကြိမ်ကနည်းမယ်ဆိုရင် inline function ကိုကြော်လှုံးပြီး call ခေါ်တာဟာပုံမြန်မှာပါ။ inline function တစ်ခုကို create လုပ်ချင်ရင် keyword inline ကို function declaration ရွှေ့အုံမှာထည့်ပေးရပါမယ်။ inline function လို့ကြော်ထားရင် compiler ဘာတကယ် function လို့ create လုပ်မပေးတော့ပဲ function call ခေါ်တဲ့အခါကျေမှ inline function မှာရေးထားတဲ့ code statement တွေကို calling function မှာ တိုက်ရှိရက်ကူးထည့်ပေးပြီး အလုပ်ဖြစ်အောင်လုပ်ခိုင်းရှိတယ်။ ဒါတော့ inline function မှာ code statement တစ်ခုခုနှစ်ခုလောက်ပဲ ပါတာမျိုးမှ အဆင်ပြောပေါ့။ program ကိုဒီတိုင်းကြည့်ရင် inline function တွေဟာ ပုံမှန် function တွေကို call ခေါ်သလိုပါပဲလေ။

၂.၆ Local and Global Variables

local variables ဆိုတာ main() function နဲ့ တစ်ခြား function တွေရဲ့ body ထဲမှာ declare လုပ်ထားတဲ့ variable name တွေပါပဲ။ သူတို့ဟာ မိမိရောက်ရှိနေတဲ့ function တစ်ခုချင်းနဲ့ပဲဆိုင်ပါတယ်။ function တစ်ခုထက် variable တစ်ခုဟာ နောက် function ထဲက variable နဲ့နာမည်ချင်းတူနေအောင် program မှာ အနောင့်အယှဉ်မရှိပါဘူး။ သူတို့တွေရဲ့ scope ဘာ မိမိရောက်ရှိနေတဲ့ function နဲ့ပဲဆိုင်တာကိုး။ နောက်တစ်ခုက function header တွေထက် parameter list တွေကို အဲဒီ function အတွက် local variable တွေလို့သတ်မှတ်ပါရပါတယ်။ ဒီ variable တွေကို function body ထဲမှာထပ်ပြီး declare လုပ်စရာမလိုပါဘူး။ global variable တွေကို main() function အပြင်ဘက် program ရဲ့ထိပ်အုံမှာ declare လုပ်ရပါမယ်။ variable scope ဘုတေသာ့ program တစ်ခုလုံးနဲ့ဆိုင်ပါတယ်။ ကောင်းပြီး၊ ပုံ (၂၁၂၃) မှာဖော်ပြထားတဲ့ EX2010.cpp program ကိုလေ့လာကြည့်ရအောင်။

```

// Listing 2.10: This program creates a function power()
// to compute the value of integer raised to the yth power.

#include <iostream>
#include <cmath>
long int power(int, int);

int main( )
{
    long int x,y,temp;

    cout << "Enter x and y\n";
    cin >> x >> y;
    cout.precision(10);
    temp = power(x,y); // function call
    cout << x << " to the power of "
        << y << " = " << temp;
    cout << endl << "\nUsing math function pow():"
        << pow(x,y) << endl;
    cout << endl;
    return 0;
}

long int power(int a, int b) // function header
{
    long int temp = 1;
    for (b > 0; b--)
        temp *= a;

    return temp;
}

```

ဦး (၂။၂၃)

၂။၁၁ Ex2010.cpp program ကို လေ့လာကြည့်မယ်ဆိုရင်

- temp ဆိတ္တဲ့ variable name တစ်ခုဟာ main function မှာရော । power() function မှာရော (2) မျိုးလုံးမှာ ပါနေပါတယ်။ ဒါပေမယ့်သူတို့ချင်းဟာ ဘာမှုမဆိုင်ပါဘူး။ သူတို့ရောက် function တစ်ခုချင်းစိန့်ပဲဆိုင်ပါတယ်။ main() function တဲ့မှာ local variable (3) - long int လို့ကြော်ပြောပါတယ်။ ဒီထဲမှာ temp ဆိတ္တဲ့ variable လည်းပါတာကိုသတိပြု၏

- $x \neq y$ အတွက် data တွေကိုထည့်ပေးပြီးရင် power() function ကို call ခေါ်ပါတယ်။ calling argument တွေက $x \neq y$ ဖြစ်ပါတယ်။ power() function ထဲ ရောက်လာတဲ့အခါ $x \neq y$ တို့ရဲ့တန်ဖိုးတွေကို function header က local variable တွေဖြစ်တဲ့ $a \neq b$ မှာအစား ပြောင်းထည့်ပေးပါတယ်။
- power() function body ထဲက temp ဟာ main() function ထဲက temp နဲ့နာမည်ချင်း တူပေါ်မယ့် ဘာမှာမသက်ဆိုင်ပါဘူး။ function ထဲမှာတွက်ချက်ပြီးလို့ temp ကို return လုပ်ပေး တာဟာ power() function ရဲ့တန်ဖိုးကို return လုပ်ပေးတာနဲ့အတူတူပါပဲ။ main() ကို ပြန်ရောက်တဲ့အခါ local variable ဖြစ်တဲ့ temp ထဲကို power() function ရဲ့ value ကို အခုလို့ temp ► power(x,y) ► tempပြောင်းထည့်ပေးပါတယ်။
- နောက်ဆုံးမှာ အဖြေ (2) ခုကိုနှိမ်ငြိမ်ပြထားပါတယ်။ ပထမတစ်ခုက ကျွန်တော်တို့ program ရေါ်ပြီး တွက်လို့ရတဲ့အဖြေပါ။ နောက်အဖြေကတော့ math library function ဖြစ်တဲ့ pow() function ကိုအသုံးပြုပြီး တွက်လို့ရတာပါ။ ဒါလောက်ဆိုရင် local variable တွေရဲ့အမိပို့ယိုင်းပြီးလို့ထင်ပါတယ်။ ပုံ (J. J9) မှာ Ex2010.cpp ကို run ပြထားပါတယ်။

```

Quincy 99
Enter x and y
5
6
5 to the power of 6 = 15625
Using math function pow() : 15625
Any key to return to Quincy...

```

ပုံ (J. J9)

- Ex2010.cpp ကိုနောက်တစ်မျိုးရေးကြည့်ပါမယ်။ ပုံ (J. J9) မှာပြထားတဲ့ Ex2010A.cpp program မှာ global variable ကို အသုံးပြုပြီးပြင်ရေးထားပါတယ်။ Ex2010A.cpp ဟာပထမ program နဲ့တူသလိုထင်ရပေါ် ရေးပုံရေးနည်း အနည်းငယ်ကွာခြားသွားပါတယ်။ temp ကို global variable အနေနဲ့ကြော်ထားတဲ့အတွက် main() ထဲက temp နဲ့ power() ထဲက temp တို့ဟာ တစ်ခုတည်းသော variable ဖြစ်နေပါပြီ။ function body ထဲမှာတွက်ချက်လို့ရတဲ့

temp value ကို return လုပ်စရာမလိုပဲနဲ့ main() ကိုပြန်ရောက်သွားအောင်လုပ်ပေးပါလို့
မယ်။ ဒါကိုကြည့်ခြင်းအားဖြင့် global variable တွေဟာ function တွေကြား ကူးကလာပြန်ဖြေ
data transfer လုပ်ပေးနိုင်တဲ့ အစွမ်းရှုတဲ့သဘောပေါ့။ စာဖတ်သူကိုယ်တိုင် ဒီ program (2)
ခုကို နှိပ်တော်လေ့လာကြည့်စေချင်ပါတယ်။ program ကို run မထိဆိုရင် ပထမအဖြမ်းပဲရမှာပါ။

```
// Listing 2.10A: This program demonstrates
// the use of global variables.

#include <iostream>
void power (int, int);
long int temp;           // define global variable

int main()
{
    long int x, y;

    cout << "Enter x and y\n";
    cin >> x >> y;
    cout.precision(10);
    power (x,y);
    cout << x << " to the power of " << y << " = "
        << temp << endl;
    return 0;
}

void power (int a, int b)
{
    temp =1;
    for ( b > 0; b--)      temp *= a;
}
```

ဗု (J-JJ)

Overloaded Functions

C++ program တစ်ခုမှာ function အများကြီးကို function name တစ်ခုတည်းပေးပြီး create လုပ်ခုပါတယ်။ အဲဒါကို function overloading လိုပေါ်ပါတယ်။ function name တူပောင်း parameter list ဘွဲ့တူတော့ဘူးပေါ့။ argument type ၊ argument number တွေလည်းကွဲပြားမှာပါ။ ဥပမာ C library မှာ ရင် absolute value ရှာတဲ့ function သုံးမျိုးလောက်ရှိပါတယ်။ integer အတွက်ဆိုရင် abs() ၊ long integer အတွက်ဆိုရင် labs() ၊ floating-point value အတွက်ဆိုရင် fabs() ဆိုပြီးခဲ့ထားပါတယ်။ ဒီဦ့စွာ သုတေသနမှတ်တော်ဘူးလား၊ အများများနိုင်ပါတယ်။ ဒီအခက်အခဲကို C++ မှာ function overloading နည်း အသုံးပြုးမဖြစ်နိုင်ပါတယ်။ function overloading ကို function polymorphism လိုလည်းခေါ်ပါတယ်။ polymorphism ထာက many ၊ morph က form ပါ။ ဒါကြောင့်မို့ ပုံစံအမျိုးမျိုးရှိတဲ့ function တစ်ခုကို polymorphic function လိုပေါ်တာပဲဖြစ်ပါတယ်။

ဗု (၂၆) မှာပြထားတဲ့ Ex2011.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- program စစ်ချင်းမှာ PI = 3.141593 နဲ့ radius ၊ height ၊ volume တို့ကို global float variable တွေလိုပေါ်ပါတယ်။ calVol() ကို overloaded function တွေလုပ်ပေးထားပါတယ်။ main() function ထဲဝင်လာလာချင်း radius နဲ့ height တို့အတွက် data တွေကို keyboard ကနေရှိကြတယ်ရမှာပါ။ ပြီးတော့ရင် ပထမ calVol(radius) ကို call ခေါ်ပြီး void calVol(float r) function ထဲမှာ volume = 4*PI*r*r*r/3 ဆိုတဲ့ formula အသုံးပြုပြီး sphere volume ကိုတွက်ယူပါတယ်။ volume ကို global လုပ်ထားတာမို့ return လုပ်ပေးစရာမလိုပါဘူး။ main() မှာ volume တန်ဖိုးကို အလိုလိုသိနေပါပြီ။
- နောက်တစ်ခါ call ခေါ်တဲ့ calVol() function မှာ argument (2) ခုပါပါတယ်။ radius နဲ့ height ကို argument အနေနဲ့ pass လုပ်ပေးလိုက်ရင် calVol(float r, float h) ထဲဝင်သွားပြီး volume= PI*r*r*h ဆိုတဲ့ formula အသုံးပြုပြီး cylinder volume ကိုတွက်ယူပါတယ်။
- ဒီ program ကို run လိုက်မယ်ဆိုရင် ဗု (၂၁၂) မှာပြထားတဲ့အတိုင်းမြင်ရမှာပါ။ radius = 5.5 နဲ့ height = 7.5 ကိုထည့်ပေးကြည့်ပါ။ sphere နဲ့ cylinder volume တွေကိုတွက်ပေးလိုက်ပါပြီ။

Ex2011.cpp

```
// Listing 2.11: Using overloaded functions

#include <iostream>
const float PI = 3.141593;
float radius, height, volume;
void calVol(float);
void calVol(float, float);

int main()
{
    cout << "\nEnter radius : ";
    cin >> radius;
    cout << "Enter height : ";
    cin >> height;

    calVol(radius);
    cout << "\nVolume of sphere is " << volume << endl;

    calVol(radius, height);
    cout << "\nVolume of cylinder is " << volume << endl;
    return 0;
}

void calVol(float r)
{
    volume = 4*PI*r*r*r/3;
}

void calVol(float r, float h)
{
    volume = PI*r*r*h;
}
```

Q (J. J6)

Quincy 99

```
Enter radius : 5.5
Enter height : 7.5

Volume of sphere is 696.91
Volume of cylinder is 712.749

Any key to return to Quincy...
```

Q (J. J2)

Overloading Function myDate()

ဗုဒ္ဓ (၂၁၂) မှာပြထားတဲ့ Ex2012.cpp program ဟာဆိုရင် string type date နဲ့ integer type date ရက်စွဲ (2) မျိုးကို overloaded myDate() function တွေကိုအသုံးပြုပြီး display လုပ်ခိုင်းတာဖြစ်ပါတယ်။ call ခေါ်ပုံခေါ်နည်းကို လေ့လာကြည့်ပါ။

```
// Listing 2.12: This program shows function
// myDate() overloading

#include <iostream>
void myDate (char *str);      // date as string
void myDate(int, int, int);   // date as numbers

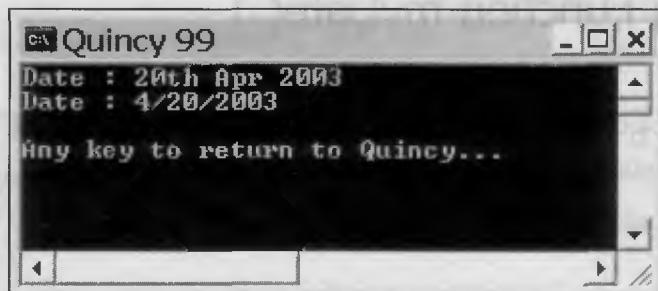
int main()
{
    myDate("20th Apr 2003");
    myDate(4,20,2003);
    return 0;
}

void myDate (char *str)
{
    cout << "Date : " << str << endl;
}

void myDate (int month, int day, int year)
{
    cout << "Date : " << month << "/" << day << "/"
        << year << endl;
}
```

ဗုဒ္ဓ (၂၁၂)

၂။ ဒါ program ကို run လိုက်မယ်ဆိုရင် ဗုဒ္ဓ (၂၁၂) မှာပြထားတဲ့အတိုင်း Date: 20th Apr 2003 နဲ့ Date: 4/20/2003 ဆိုတဲ့ရက်စွဲ (2) မျိုးကို display လုပ်ပြနေပါပြီ။



ও (J. Jc)

Overloading with Default Arguments

॥ Ex2013.cpp program ক্ষেত্রে প্রদর্শিত repeat_ch() function prototype এ default argument দ্বারা প্রক্রিয়াজনক করা হয়েছে। এই ক্ষেত্রে এই ফাংশনের নামটি পুরো ক্ষেত্রে প্রদর্শিত হয়ে থাকে। অন্য ক্ষেত্রে এই ফাংশনের নামটি পুরো ক্ষেত্রে প্রদর্শিত হয়ে থাকে।

```
// Listing 2.13: This program demonstrates function
// overloading with default arguments.

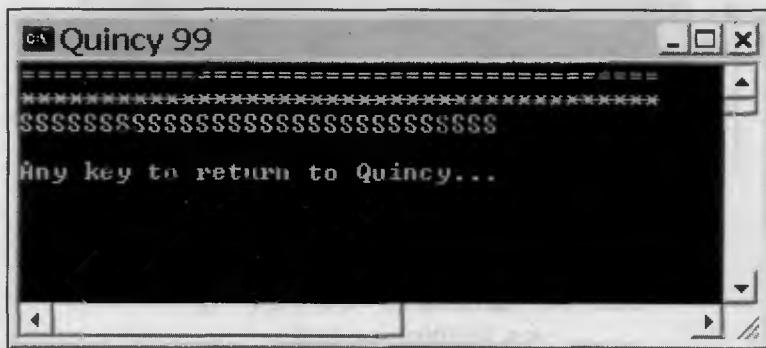
#include <iostream>
void repeat_ch(char = '=', int = 40);

int main( )
{
    repeat_ch( );
    repeat_ch('*');
    repeat_ch('S', 30);
    return 0;
}

void repeat_ch(char ch, int n)
{
    for (int i=0; i<n; i++) cout << ch;
    cout << endl;
}
```

ও (J. ১০)

ဒါ program ကို run လိုက်မယ်ဆိုရင် ပုံ (၂၁) မှာပြထားတဲ့အတိုင်း character line (3) မျိုးကို display လုပ်ပြတာ တွေ့ရမှုပါ။



ပုံ (၂၁)

J.1 Recursion

I. recursion ဆိုတာ function တစ်ခုကနေ same function ထဲကို ထပ်ခါတစ်လဲ call ပြန်ခေါ်ပြီးတော့ သတ်မှတ်ထားတဲ့ condition တစ်ခု မရောက်သေးမချင်း looping ပတ်နေတာပါပဲ။ recursive program တစ်ခုကို run တဲ့အခါမှာ function call တွေဟာချက်ချင်း execute လုပ်ပြီးအမြဲတ်မပေးပါဘူး။ အဖြေတွေကို (တိုင် တစ်တိုင်မှာ အပေါက်ပါတဲ့သံပြားအဝိုင်းတွေကို ထည့်စွဲထားပေးသလိုမျိုး) ပြောင်းပြန်ထပ်စီသွားပါတယ်။ သတ်မှတ်ထားတဲ့ condition တစ်ခုကိုရောက်လာလို့ recursion ရပ်သွားတော့မှ အဖြေတွေကို (တိုင်ကနေ သံပြားဝိုင်းတွေ တစ်ခုချင်းထုတ်ပေးသလို) ပြောင်းပြန်အစီအစဉ်နဲ့ ကွန်ပူးတာကထုတ်ပေးသွားမှာပါပဲ။ အဲဒါဟာ recursion နည်းပါ။ ကောင်းပြီ။ ငက်န်းတစ်ခုရဲ့ factorial ကို recursive နည်းနဲ့ရှာကြည့်ရအောင်။ ပုံ (၂၂) မှာရေးထားတဲ့ code statement တွေကိုကြည့်ပါ။

II. ဒီပူးစွာကို ဘယ်လိုရှင်းသွားတယ်ဆိုတာ ကျွန်တော် အမိအစဉ်ချုပြီးတော့ရေးပြပါမယ်။

$$n! = n \times (n - 1)!$$

$$(n - 1)! = (n - 1) \times (n - 2)!$$

$$(n - 2)! = (n - 2) \times (n - 3)!$$

```
Ex2014.cpp

// Listing 2.14: This program calculates the factorial of
// a positive integer using recursive method.

#include <iostream>
long double facto(int);

int main()
{
    int n;

    cout << "Enter a number : ";
    cin >> n;
    cout << "Factorial of " << n << " is "
        << facto(n) << endl;
    return 0;
}

long double facto(int i)
{
    if (i <= 1)
        return 1;
    else
        return (i * facto(i-1));
}
```

```
....  
4 ! = 4 x 3 !  
3 ! = 3 x 2 !  
2 ! = 2 x 1 !
```

սաւ function call မှာ $n!$ ကို $n \times (n-1)!$ လိုသတ်မှတ်ပြီး execute မလုပ်သေးဘဲ တိုင်မှာစွင်ပါတယ်။ ဒီဟာတိုင်ရဲအောက်ဆုံးမှာ ရှိနေမှာပါပဲ။ ပြီးတော့ ဒုတိယအကြိမ် function call ကို ခေါ်တဲ့အခါ argument က $(n-1)$ ဖြစ်သွားပါပြီ။ ဒီတော့ $(n-1)!$ ကိုလည်း $(n-1) \times (n-2)!$ လိုသတ်မှတ်ပြီးတော့ execute မလုပ်သေးပဲ တိုင်မှာပဲထပ်စွာပြန်ပါတယ်။ ဒီလိုနည်းနဲ့ recursion ကိုဆက်လုပ်သွားတာဟာ argument အနှစ်တန်ဖိုးဖြစ်သွားတာနဲ့ process ဟာရပ်သွားပြီးတော့ actual value တွေကို reverse order နဲ့ကွန်ပျုံတွက်ထုတ်ပေးပြီလေ။ ပုံ (J. ၃၃) မှာ program run ပြထားတာကိုဖြည့်ပါ။

```

Quincy 99
Enter a number : 100
Factorial of 100 is 9.33262e+157
Any key to return to Quincy...

```

ද (J. ၃၃)

The Tower of Hanoi

၁။ Ex2015.cpp program ကလည်း recursive program တစ်ခုဖြစ်ပါတယ်။ အလယ်မှာအပေါက်ပါပြီး ဘဏ္ဍားအသေးမတူတဲ့ အပိုင်းပြားတွေကို ထိုင် (3) ထိုင်မှာစွဲပေးတဲ့အခါ ဘယ်ထိုင်မှာပဲစွဲပဲစွဲ အသေးအပြားက ဘဏ္ဍားအပြားပေါ်မှာ ရောက်နေအောင်စွဲပေးချင်ရင် တစ်ဆင့်ပြီးတစ်ဆင့် မမှားအောင်လုပ်သွားရမယ့်လမ်းကြောင်းကို ဒီ program ကတွက်ထုတ်ပေးမှာပါ။ ပထမထိုင်ကို LEFT၊ ဒုတိယထိုင်ကို CENTER နဲ့ တတိယထိုင်ကို RIGHT လိုခေါ်ပါမယ်။ အခုနေ disk (3) ခုကိုထိုင်မှာစွဲပေးချင်ရင် ද (J. ၃၇) မှာပြထားတဲ့အစီအစဉ်အတိုင်းလုပ်ရပါမယ်။

```

Quincy 99
Welcome to the TOWERS OF HANOI
How many disks ? 3
Move disk 1 from L to R
Move disk 2 from L to C
Move disk 1 from R to C
Move disk 3 from L to R
Move disk 1 from C to L
Move disk 2 from C to R
Move disk 1 from L to R
Any key to return to Quincy...

```

ද (J. ၃၇)

၂။ ဒါ program ရဲ့ source code တွေကို ပုံ (J· ၃၅) မှာရေးပြထားပါတယ်။ စာဖတ်သူကိုယ်တိုင် traceလုပ်ကြည့်ဖို့ အလုပ်ချွန်ခဲ့ပါမယ်။ စိတ်ရှည်ရှည်ထားပြီး program လမ်းကြောင်းကိုရှာကြည့်ပါ၊ ရပါတယ်။

```
// Listing 2.15: This program is a game used to transfer
// the disks from the leftmost pole to the rightmost pole,
// without ever placing a larger disk on top of a smaller
// disk. Only one disk may be moved at a time.

#include <iostream>
void transfer(int, char, char, char);

int main()
{
    int nDisk;

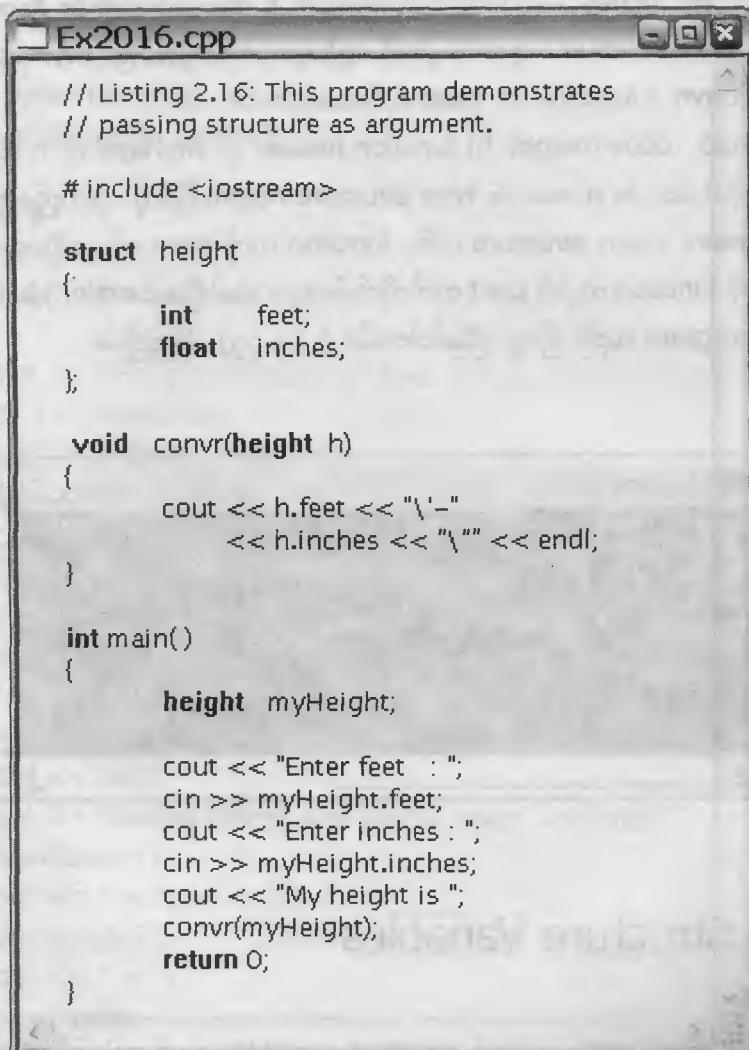
    cout << "Welcome to the TOWERS OF HANOI\n\n";
    cout << "How many disks ? ";
    cin >> nDisk;
    cout << endl;
    transfer(nDisk, 'L', 'R', 'C');
    return 0;
}

void transfer(int n, char from, char to, char temp)
{
    if (n > 0)
    {
        transfer(n-1, from, temp, to);
        cout << "Move disk " << n << " from " << from;
        cout << " to " << to << endl;
        transfer(n-1, temp, to, from);
    }
}
```

ပုံ (J· ၃၅)

Passing Structure Variables

function ထဲမှာ constant တွေ၊ variable တွေကိုတစ်ခုချင်း pass လုပ်လိုရသလို data type အုပ်စု အနေဖြင့်တဲ့ structure လိမ့်မျိုးကိုလည်း pass လုပ်ပေးလိုပါတယ်။ ဗု (J. ၃၆) မှာပြထားတဲ့ Ex2016.cpp program ဟာဆိုရင် structure တစ်ခုကို function ထဲရောက်အောင် pass လုပ်ပေးနိုင်ပါတယ်၊ လေးလာကြည့်ပါ။



```

// Listing 2.16: This program demonstrates
// passing structure as argument.

#include <iostream>

struct height
{
    int    feet;
    float inches;
};

void convr(height h)
{
    cout << h.feet << "-" 
        << h.inches << "" << endl;
}

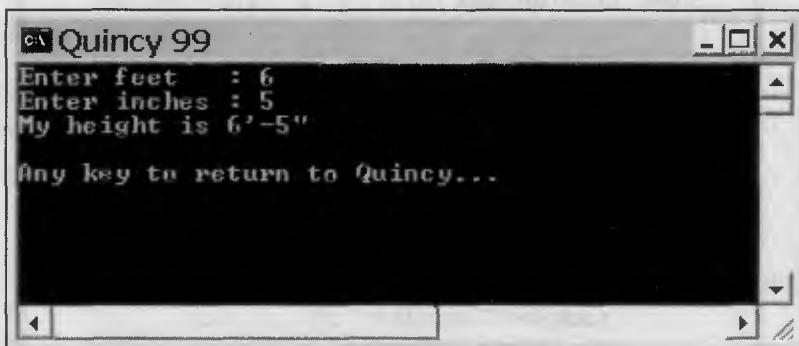
int main()
{
    height myHeight;

    cout << "Enter feet : ";
    cin >> myHeight.feet;
    cout << "Enter inches : ";
    cin >> myHeight.inches;
    cout << "My height is ";
    convr(myHeight);
    return 0;
}

```

ဗု (J. ၃၆)

- စစချင်း struct ဆိုတဲ့ keyword ကိုအသုံးပြုပြီး height ဆိုတဲ့ structure name ရှိတဲ့ type တစ်ခုကိုသတ်မှတ်ပေးပါတယ်။ အဲဒီထဲမှာတဲ့ structure member (2) ခုက feet နဲ့ inches ပါ။ main() ထဲမှာ myHeight ဆိုတဲ့ type structure height အသစ်တစ်ခုကို create လုပ်ပါတယ်။
- myHeight ရဲ့ feet အရှည်ကိုတောင်းတဲ့အတွက် keyboard ကနေ 6 ကိုရိုက်ထည့်ပေးမယ် ဆိုပါမိ။ inches အရှည်ကိုတောင်းတဲ့အအခါ 5 ကိုထည့်ပေးပါမယ်။ ဒီတော့ myHeight ရဲ့ member value တွေတစ်ခုချင်းကို ကွန်ပူးတာကသိသွားပါပြီ။ myHeight value တွေကို convr() function ထဲ pass လုပ်ပေးရအောင်။
- void convr(height h) function header မှာ myHeight ကို h ထဲ ကော်ပီကူးထည့်ပေးလိုက်ပါတယ်။ h ကလည်း type structure height ဖြစ်လို့ ကူးထည့်ပေးလို့ရတာပါ။ ဒါဆိုရင် main() ထဲက structure တစ်ခု function ထဲကို pass ဝင်သွားပြီလေ။ structure value ကို function ထဲမှာပဲ print ထုတ်လိုက်ပါတယ်။ အဲဒါ ပြီးရင် main() function ကို ပြန်လာဖြီး program လည်း ပြီးသွားပြီဖြစ်ပါတယ်။ ပုံ (၂၁၃၃) ကိုကြည့်ပါ။



ပုံ (၂၁၃၃)

Returning Structure Variables

- ၁။ Ex2017.cpp program ဟာဆိုရင် structure variable တွေကိုအသုံးပြုပြီး length (2) ခုကိုပေးမှာပါ။ ဒီ program ကိုလေ့လာကြည့်မယ်ဆိုရင်

```
// Listing 2.17: This program demonstrates returning a structure

#include <iostream>

struct Length
{
    int feet;
    float inches;
};

void convr(Length);
Length add(Length, Length);

int main( )
{
    Length piece1,piece2,total;

    cout << "For piece1\n\tEnter feet : ";
    cin >> piece1.feet;
    cout << "\nEnter inches : ";
    cin >> piece1.inches;

    cout << endl;
    cout << "For piece2\n\tEnter feet : " cin >> piece2.feet;
    cout << "\nEnter inches : " cin >> piece2.inches;

    total = add(piece1, piece2);
    cout << endl;
    cout << "Adding piece1 and piece2 gives" << endl;
    convr(piece1);
    cout << " + ";
    convr(piece2);
    cout << " = ";
    convr(total);
    cout << endl;
    return 0;
}
```

```

Length add(Length x, Length y)
{
    Length z;
    z.inches = x.inches + y.inches;
    z.feet = 0;
    if (x.inches >= 12.0)
    {
        z.inches -= 12.0;
        z.feet++;
    }
    z.feet += x.feet + y.feet;

    return z;
}

void convr(Length x)
{
    cout << x.feet << "-" << x.inches << "\n";
}

```

- main() ထဲမှာ piece1 | piece2 | total တိုကို structure type Length ဖြစ်အောင် create လုပ်ပါတယ်။ Length ရဲ့ declaration ကို program ထိပ်ဆုံးနားမှာရေးထားပါတယ်။ piece1 နဲ့ piece2 တို့ရဲ့ member value တွကိုရှိက်ထည့်ပေးပြီး add() function ကို call ခေါ်ပါတယ်။ argument တွက piece1 နဲ့ piece2 ဆိုတဲ့ structure တွေပါပဲ။
- add() function ထဲကို ရောက်လာတဲ့အခါမှာ piece1 ကို x | piece2 ကို y နဲ့ထပ် assign လုပ်ပေးတဲ့အတွက် x.inches ဟာ piece1.inches နဲ့အတူတူပါပဲ။ ဒီတော့ z.inches = x.inches + y.inches = 7+11= 18 ဖြစ်ပါလိမ့်မယ်။ z.feet ကို 0 လို့ initialize လုပ်ထားပါတယ်။ z.inches=18 ဟာ 12 ထက်ကြီးလား၊ ညီလားလို့မေးပါတယ်။ ဒါ <test> ဟာ True ဖြစ်တဲ့အတွက် if block ထဲဝင်ပြီး z.inches -=12 ဆိုမဟုတ် z.inches = z.inches -12 = 18-12 = 6 လို့တွက်ယူပါတယ်။ 1 ပေ ပိုတွက်လာတဲ့အတွက် z.feet ထဲမှာပေါင်းထည့်ပေးရမှာပေါ့။ ဒါကြောင့်မို့ z.feet ++ လို့ရေးလိုက်တာပါ။ အခုဆိုရင် z.feet = z.feet + 1 = 0+1 = 1 ဖြစ်သွားပြီ။

ශොක්න්තේ z.feet = z.feet + x.feet + y.feet සිංහැ මෙය ගෙන 1+25+55 = 81 ක්ද තුළ යුතු තාවක් ප්‍රමුණ පිටයි। ඇම ස්කීර්දන structure z තුළා z.feet = 81 න් සහ z.inches=6 ක්ද රැක්වූ මහුත්ලයා යි। z තුළ return ලද්ද පෙස දිග්‍රියා පිටු ඇත්තා මෙයි main() තුළ රෙළා තුළ ඇමුවා total = add(piece1, piece2) නෙක් ප්‍රමුණ තුළ පිටු වියේ total member තුළා total.feet = 81 න් සහ total.inches = 6 ඇවියි ප්‍රමුණ නෙක් නෙක් නෙක් convr(total) ගෙන ඇප්‍රේෂ්‍රී තුළ පෙස පිළිගුණු යි। එය (J. 21) තුළ පිටු ඇතුළත් යි।

```

Quincy 99
For piece1
    Enter feet : 25
    Enter inches : 7

For piece2
    Enter feet : 55
    Enter inches : 11

Adding piece1 and piece2 gives
25'-7" + 55'-11" = 81'-6"

Any key to return to Quincy...

```

දෑ (J. 21)

Chapter 3



C++ programming မှာ program တစ်ခုရဲ့ program flow ကို control လုပ်ပေးနိုင်တာ ဟာဆိုရင် C++ control statement တွေပါပဲ။ C++ မှာအသုံးများဆုံး statement တွေကတော့

break	do	if
case	else	return
continue	for	switch
default	goto	

တိဖြစ်ကြပါတယ်။ ဒီအခန်းမှာ အရင်ဆုံးတင်ပြမှာက selection test တွေလုပ်လို့ရတဲ့ if statement ပါပဲ။

၃.၁ The if Statement

၀။ if statement ဆိုတာ conditional statement တစ်ခုဖြစ်ပါတယ်။ if statement ကိုသုံးတဲ့အခါမှာ if နောက်ကကွင်းထဲမှာရှိတဲ့ <test expr> မှန်လားများလားဆိုတာကို အရင်စိစစ်ရပါတယ်။ အများအားဖြင့် <test

expr> තොග relational expression මූද්‍රාවකින් පිතයේ if statement නැංවා සඳහා පිතයේ

if (<test expr>) <statement1>;

මිනු ඇත්තු ඇත්තු **<test expr>** කිහිපියෙන් ලිඛිත නොවා නොවා nonzero එක් අනුමත TRUE පිතයේ if statement නැංවා සඳහා **<statement1>** නිස් පිතයේ execute ලදී පිතයේ if statement නැංවා සඳහා **<test expr>** නැංවා zero (FALSE) නැංවා **<statement1>** නිස් පිතයේ execute ලදී **next line** නිස් පිතයේ if statement නැංවා **<test expr>** නැංවා nonzero එක් අනුමත TRUE පිතයේ if statement block නිස් **brace{ }** නැංවා පිතයේ if statement block නැංවා **single statement** නැංවා execute ලදී **if statement** නැංවා Ex301.cpp program නිස් (2.0) මා පිතයේ if statement | ලදාලාගැනීම් පිතයේ

```
// Listing 3.1: This program illustrates the use of if statement.
#include <iostream>

int main()
{
    short int num;
    cout << "Enter a number\n";
    cin >> num;

    if (num < 0) cout << "The number is -ive.\n";
    if (num > 0) cout << "The number is +ive.\n";
    if (num == 0) cout << "The number is zero.\n";
    return 0;
}
```

දු (2.0)

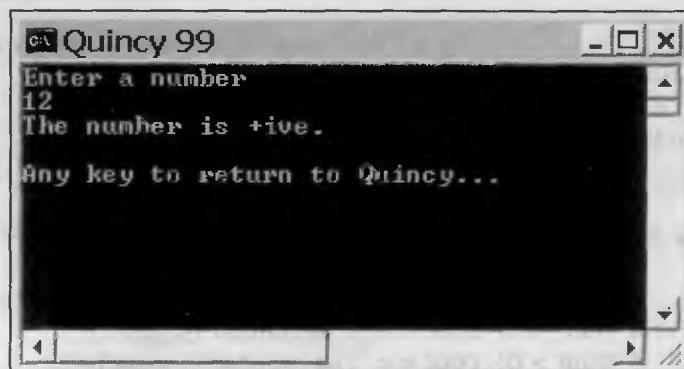
JII Ex301.cpp program නිශ්චිත පිතයේ

- program නැංවා num නිස් short integer variable නැංවා පිතයේ num

ရဲတန်ဖိုးဟာ -32,768 ကနေ 32,767 အတွင်းမှာရှိနေရပါမယ်။ Enter a number ဆိုတဲ့ prompt ပေါ်လာတဲ့အခါမှာ num အတွက် data ကို 12 ရှိက်ထည့်ပြီး ENTER key နှင့်မ ဆိုရင် num = 12 ဖြစ်သွားပါပြီ။

ပထမဆုံး if statement နဲ့တွေ့ပါတယ်။ num က သူည်ထက်ငယ်လားတဲ့ ဘယ်ငယ်မလဲ zero (FALSE) ကိုထုတ်ပေးလိုက်တဲ့အတွက် ကွန်ပျူးတာက နောက်တစ်ကြောင်းကိုဆင်းသွားပါ။ ဒုတိယ if statement မှာ num က သူည်ထက်ကြီးလား မေးပါတယ်။ num = 12 က သူည်ထက်ကြီးတာမှန်ပါတယ်။ ဒီတော့ကွန်ပျူးတာက <test expression> နောက်မှာရှိတဲ့ The number is +ive. ဆိုတဲ့စာကြောင်းကို ကွန်ပျူးတာမှာပေါ်လာအောင်လုပ်ပေးပါတယ်။ ဖြိုးတော့ရင် နောက်တစ်ကြောင်းကိုဆက်ဆင်းလာမှာပါ။

နောက်ဆုံး if statement ကတော့ num ကို သူည်နဲ့ညီလားလို့မေးပါတယ်။ မညီပါဘူး။ FALSE ဖြစ်တဲ့အတွက် ကွန်ပျူးတာက next line ကို ဆက်ဆင်းသွားပါတယ်။ closing brace (}) တွေ့တဲ့အခါ program ပြီးသွားပါပြီ။ ပုံ (၃။၂) မှာ program ကို run ပြထားပါတယ်။



ပုံ (၃။၂)

Conditionally Executing a Program Block

Ex301.cpp program ကိုပြပြင်ဖြီး နောက်တစ်မီးရေးလို့ရတာက num ဟာ positive value မှာမျှ မရှိဘူး။ အမြဲကို display လုပ်ပြပါလိမ့်မယ်။ မဟုတ်ရင် You chose not to compute မှာ message ကို display လုပ်ပြခိုင်းထားပါတယ်။

```
Ex302.cpp

// Listing 3.2: This program cubes a number if it is
// not a zero or negative value.

#include <iostream>
#include <cmath>

int main()
{
    int num;
    cout << "Enter a number\n";
    cin >> num;
    if (num)
    {
        cout << num << " cubed is " << pow(num,3) << endl;
        return 0;
    }
    cout << "You chose not to compute\n";
    return 0;
}
```

ပုံ (၃။ ၃)

၂။ ပုံ (၃။ ၃) မှာရေးထားတဲ့ Ex302.cpp program ကို run ရင် အခုလိုမြင်ရမှာပါ။ ပုံ (၃။ ၄) ကိုဖြည့်ပါ။

```
Quincy 99
Enter a number
12
12 cubed is 1728
Any key to return to Quincy...
```

ပုံ (၃။ ၄)

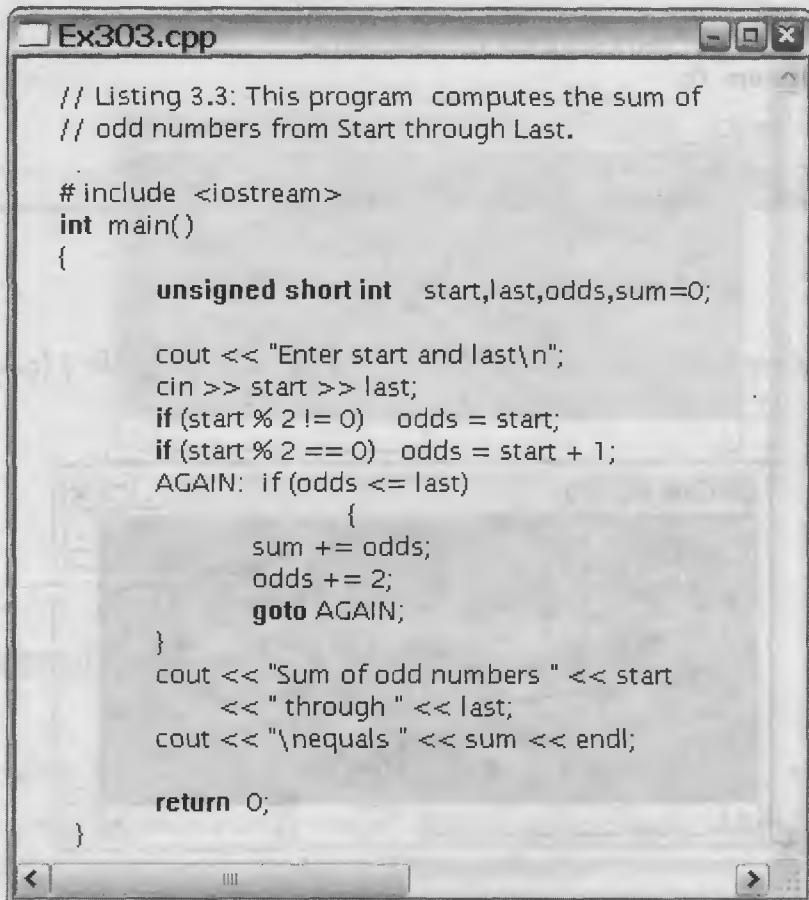
၃.၂ The goto Statement

၁။ goto ဆိတာ program တစ်ခုရဲ့ ပုံမှန်သွားနေတဲ့လမ်းကြောင်းကနေ တစ်ခြားနေရာကို လွှတ်လွတ်လပ်လပ်ခုန်ကျော်သွားချင်တယ်ဆိုရင် အသုံးပြုရမယ့် statement ပါပဲ။ goto ရဲပုံစံက ဒီလိုပါ။

<label> : target statement ;

goto <label> ;

label ဆိတာ ကွဲနိပ္ပါတာသွားချင်တဲ့နေရာဒေသကို ကျွန်ုတ်တို့ အမှတ်သညာပြုထားတဲ့ local label



```
// Listing 3.3: This program computes the sum of
// odd numbers from Start through Last.

#include <iostream>
int main()
{
    unsigned short int start, last, odds, sum = 0;

    cout << "Enter start and last\n";
    cin >> start >> last;
    if (start % 2 != 0)    odds = start;
    if (start % 2 == 0)    odds = start + 1;
    AGAIN: if (odds <= last)
    {
        sum += odds;
        odds += 2;
        goto AGAIN;
    }
    cout << "Sum of odd numbers " << start
        << "through " << last;
    cout << "\nequals " << sum << endl;

    return 0;
}
```

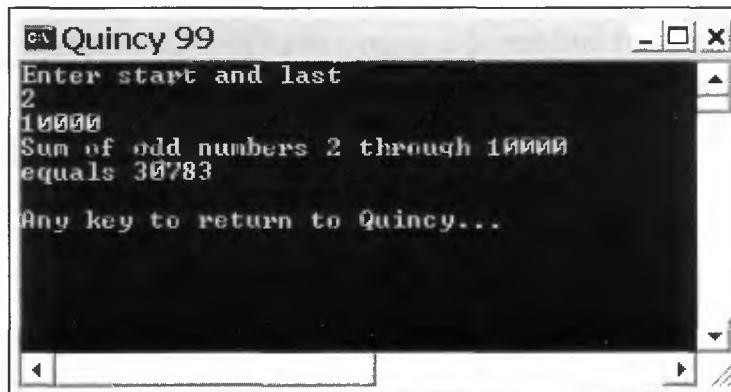
ပုံ (၃၁၂)

name ပါ။ target statement မှာ label ကို colon(:) နဲ့တွေ့ပြီးရေးပေါ်မယ်။ goto statement ကိုသုံးပြုဖိုကို ပုံ (၃.၅) က Ex303.cpp မှာဖော်ပြထားပါတယ်။ လေ့လာကြည့်ပါ။

ဒါ program ဟာ start ဆိုတဲ့ စဉ်းကဏ္န်းတစ်ခုကနေ last ဆိုတဲ့ နောက်ဆုံးကဏ္န်းတစ်ခုအတွင်းက စုံကဏ္န်းတွေအားလုံးကို ပေါင်းပေးမှာပါ။ start နဲ့ last တို့ဟာ ဖြစ်ချင်ရာကဏ္န်းတွေဖြစ်နိုင်ပါတယ်။ ဒါ program အလုပ်လုပ်သွားပုံကို လေ့လာကြည့်ရအောင်။

- program မှာအသုံးပြုမယ့် variable (4) ခုကို unsigned short integer တွေလို့ declare လုပ်ပေးပါတယ်။ ဒါတော့ variable တွေရဲ့တန်ဖိုးတွေဟာ 0 ကနေ 65,535 အတွင်းမှာပဲရှုရပါ မယ်။ ကျွန်ုတ်တို့တွေက်ချင်တာက မဂကဏ္န်းတွေအားလုံးရဲ့ပေါင်းလဲကို sum ထဲမှာ အဖြေတွေက် ထုတ်မှာပါ။ ဘာမှာတွေက်တာချက်တာမစသေးခင် sum = 0 လို့ initialize လုပ်ထားပါတယ်။
- Enter start and last ဆိုတဲ့ prompt ကိုပေါ်ခြင်းပါတယ်။ cin >> start >> last ဆိုတဲ့ statement ကြောင့် start နဲ့ last တို့အတွက် data တွေကို keyboard ကနေရရှိထည့်ပေး ရပါမယ်။ 2 <ကွဲက်လပ်> 9 သို့မဟုတ် 2 <ENTERr> 9 ကိုရရှိထည့်ပြီး ENTER ပုံတိလိုက် မယ်ဆိုရင် start= 2၊ last= 9 ဖြစ်သွားပါပြီ။
- အရေးကြီးဆုံးက စဉ်းကဏ္န်း start ဟာစုံလား၊ မလားဆိုတာကို အရင်စိစစ်ရပါမယ်။ စစ်နည်းက start ကို 2 နဲ့စားကြည့်မယ်၊ မပြတ်ဘူးဆိုရင် start ကမဂကဏ္န်း။ ပြတ်တယ်ဆိုရင် စုံဖြစ်မှာပါ။ start % 2 (start ကို 2 နဲ့စားရင် ကျွန်ုတ်အကြောင်း) က သုညနဲ့ညီတာပေါ့။ ဒါတော့ start % 2 != 0 ဆိုတဲ့ <test expression> ဟာ FALSE ဖြစ်သွားပါပြီ။ ဒါ expression ရဲ့နောက်က odds = start ဆိုတဲ့ statement ကိုကွဲနိပုံတာက မဖြေရှင်းတော့ပါဘူး။ နောက်တစ်ကြောင်း ကိုဆက်ဆင်းသွားမှာပါ။
- ဒုတိယ if ကျတော့ <test expression> က TRUE ဖြစ်သွားပါတယ်။ ဒါကြောင့် odds = start+1 = 2+1 = 3 လို့တွေက်ယူလိုက်ပါတယ်။ ဆိုလိုတဲ့အမိပါယ်က စဉ်းကဏ္န်းဟာ 2 ဖြစ် ပေမယ့် 3 ကိုပဲ စဉ်းကဏ္န်းလို့ ကျွန်ုတ်ပုံတာကပြောင်းယူလိုက်တာပါ။ odds နဲ့ last ကိုယျှဉ်လိုက်လို့ ငယ်တယ် သို့မဟုတ် ညီတယ်ဆိုရင် if နောက်က block statement ကို execute လုပ်ပါလိမ့် မယ်။ အခုလည်းပဲ odds = 3 က last = 9 ထက်ငယ်တာမို့ <test expr> က TRUE ပေါ့။ if block ထဲကိုဝင်လာပါပြီ။ sum += odds ဆိုတာကို sum = 0+3 = 3 လို့ ကွဲနိပုံတာက တွေက်လိုက်ပါတယ်။ အမိပါယ်က sum ထဲကို ပထမဆုံးမဂကဏ္န်းတစ်လုံး ထည့်လိုက်တာပါ။ 3 ပြီးရင် ဒုတိယမဂကဏ္န်းဟာ 5 မဟုတ်လား။ အဲဒီဥစ္စာရဖို့အတွက် odds += 2 ကိုအသုံးပြုပြီး odds = odds+2 = 3+2 = 5 ဆိုတဲ့ ဒုတိယမဂကဏ္န်းကိုတွေက်ယူပါတယ်။

- goto statement ဟာ AGAIN ဆိတဲ့ label name ရှိတဲ့နေရာကို ပြန်သွားခိုင်းနေတာပါ။ အခုခိုရင်ကွန်ပျူးတာဟာ block statement အတွင်းမှာ ထပ်ခါတစ်လဲပတ်နေမှာဖြစ်ပါတယ်။ တစ်ချိန်မှာ odds = 11 ဖြစ်သွားပါလိမ့်မယ်။ ဒါဆိုရင် odds = 11 က last = 9 ထက်ကြီးသွားတာကြောင့် <test expression> ဟာ TRUE ဖြစ်သွားပါဖြီ။ if block ထကိုထပ်မဝင်တော့ပဲ့အောက်ကိုဆင်းလာဖြီလေ။
- နောက်ဆုံးမှာ Sum of odd numbers 2 through 9 equals 24 ဆိတဲ့ output တော်ကြောင်းကို (2) ကြောင်းချွဲ့ပြီး အဖြေထုတ်ပေးပါလိမ့်မယ်။ ဒါဆိုရင် Ex303.cpp program ကဖြီးသွားပါဖြီ။ ပုံ (၃. ၆) မှာပြထားတဲ့အတိုင်း start = 2 နဲ့ last = 10000 လို့ data သွင်းခဲ့ရင် အကို အခုလိုမြင်ရမှာပါ။



Reversing an Integer Number

၁။ Ex304.cpp program ရဲ့ရည်ရွယ်ချက်က ကိန်းပြည့်ကဏ္န်းတစ်လုံးကို input data အနေနဲ့ ထပ်လိုက်မယ်ဆိုရင် အဲဒိုက်ဆိုရင် ပြောင်းပြန်အစဉ်နဲ့ဖြစ်အောင် ကွန်ပျူးတာကလုပ်ပေးမှာပါ။ ပုံ (၃. ၇) မှာ do statement တွေကိုရေးပြထားပါတယ်။

၂။ Ex304.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- စစ်ဆေးမှာ unsigned long int num, rev, q, r လို့ရေးထားတဲ့အတွက် ဒါ variable

Ex304.cpp

```

// Listing 3.4: This program reads an integer number
// and prints out the same number with digits reversed.

#include <iostream>

int main()
{
    unsigned long int      num, rev, q, r;

    cout << "Enter a number\n";
    cin >> num;
    q = num /10;
    rev = num % 10;
    START: if ( q != 0 )
    {
        num = q;
        q = num /10;
        r = num % 10;
        rev = rev * 10 + r;
        goto START;
    }
    cout << "The number reversed is\n" << rev << endl;
    return 0;
}

```

ဗု (၃။ ၃)

တွေခဲ့တန်ဖိုးကို ၀ ကနေ 4,264,967,295 အတွင်းမှာ ကြိုက်တဲ့ ဝက်နဲ့ assign လုပ်လို ရပါပြီ။ Enter a number ဆိုတဲ့ prompt ပေါ်လာပြီ။ `cin >> num` ကြောင့် num အတွက် data ကိုတောင်းနေပါပြီ။ keyboard ကနေ 123456789 ကိုရိုက်ထည့်ပေးလိုက်မယ်ဆိုရင် num = 123456789 ဖြစ်သွားမှာပါ။

q တန်ဖိုးကို $q = \text{num}/10 = 123456789/10 = 12345678$ လိုတွေ့က်ယူပါတယ်။ $\text{rev} = \text{num} \% 10 = 123456789 \% 10 = 9$ ပေါ့။ `if (q != 0)` ဆိုတဲ့ `<test expr>` ကို ကွန်ပျံပျောက TRUE လိုအပြောတဲ့ ပါတယ်။ q က သူညီမဟုတ်တာကိုး။ ဒီတော့ `if` block ထဲဝင်လာပြီး ကွန်ပျံပျောကအဲလို ဖြေရှင်းပါတယ်။

```

num = q = 12345678
q = num / 10 = 12345678 / 10 = 1234567
r = num % 10 = 12345678 % 10 = 8
rev = rev*10 + r = 9*10 + 8 = 98

```

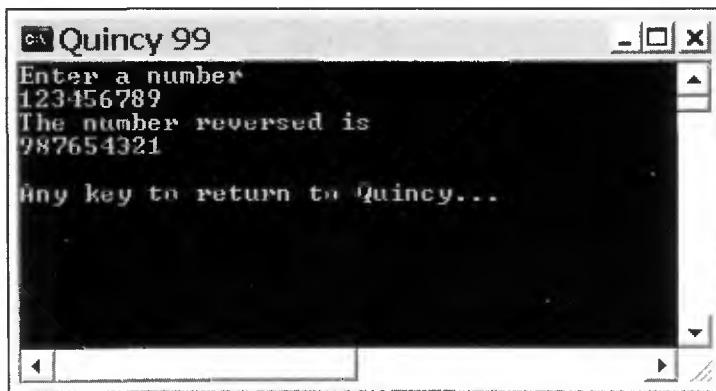
- goto START ကြောင့် ကွန်ပူးတာက START label ရှိတဲ့နေရာကို ပြန်သွားပါတယ်။ q က သုညမဟုတ်တဲ့အတွက် if block ထဲကို ကွန်ပူးတာက နောက်တစ်ကြိမ်ပြန်ဝင်လာပြီလေ။ if block ထဲမှာ အခုလိုဖြေရှင်းပါတယ်။

```

num = q = 1234567
q = num / 10 = 1234567 / 10 = 123456
r = num % 10 = 1234567 % 10 = 7
rev = rev*10 + r = 98*10 + 7 = 987
goto START ကိုပြန်သွားပါ။

```

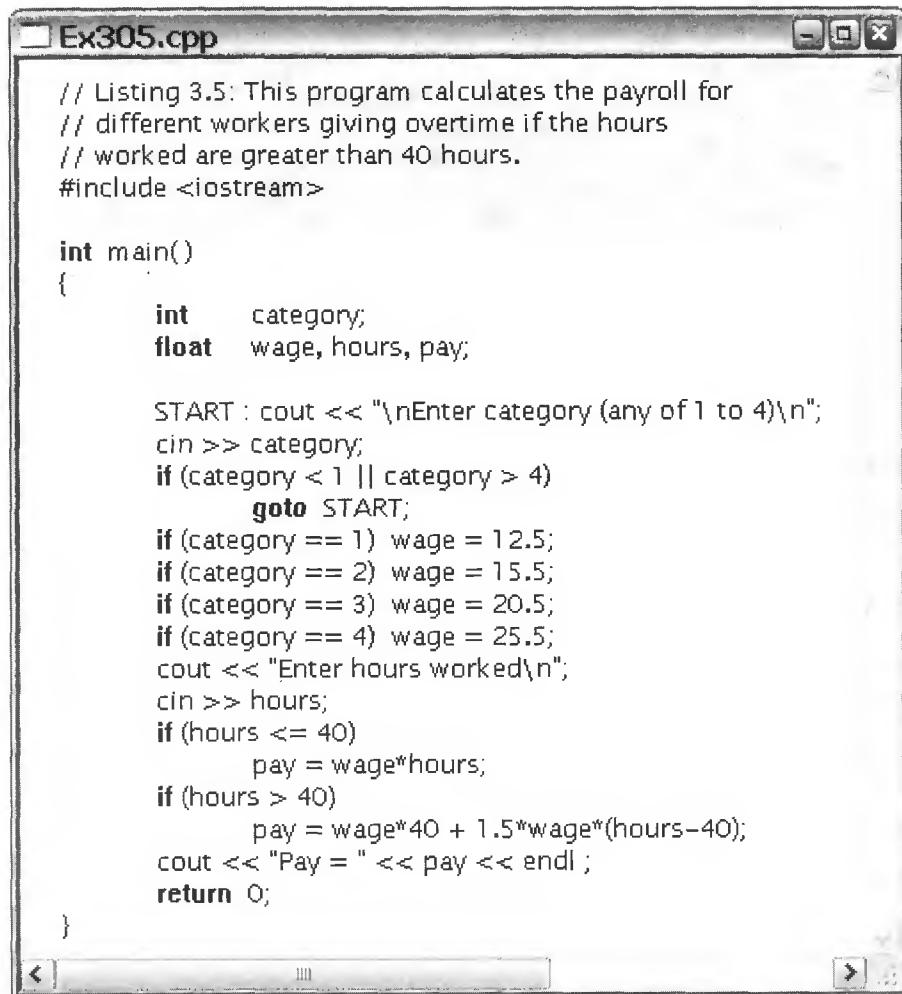
- တစ်ချိန်မှာ q = 0 ဖြစ်သွားပြီး if block ထဲကထွက်လာမှာပါ။ နောက်ဆုံးမှာ The number reversed is : 987654321 လို့ တစ်ကြောင်းစီခွဲပြီး အဖြေရှင်းကိုထုတ်ပေးပါလိမ့်မယ်။ ဒါဆိုရင် program ပြီးသွားပါပြီ။ အဖြေကို ပုံ (၃. ၁) မှာ program run ပြထားပါတယ်။



ပုံ (၃. ၁)

A Payroll Program

၁။ Ex305.cpp program ရဲ့ရည်ရွယ်ချက်က အလုပ်သမားတစ်ယောက်ရဲ့လုပ်ချိန်နဲ့ လုပ်အားခနှစ်နဲ့ကိုသိရင်လစာကိုတွက်ပေးတဲ့ program ဖြစ်ပါတယ်။ အလုပ်ချိန်နာရီ (40) ကော်ရင်အချိန်ပိုနာရီကို ပုံမှန်လုပ်အားခနှစ်နဲ့ ကစ်ဆွဲနဲ့တွက်ပေးမှာပါ။ ပုံ (၃။၉) မှာ code statement တွက်ရောပြထားပါတယ်။ လေ့လာကြည့်ပါ။



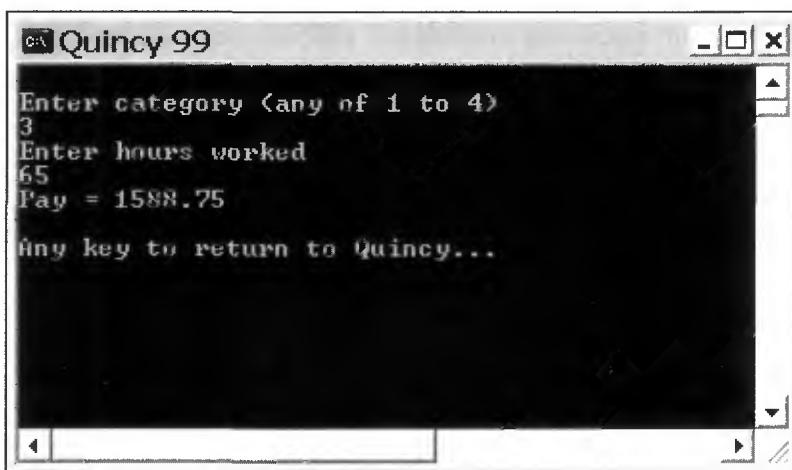
```
// Listing 3.5: This program calculates the payroll for
// different workers giving overtime if the hours
// worked are greater than 40 hours.
#include <iostream>

int main()
{
    int    category;
    float  wage, hours, pay;

    START : cout << "\nEnter category (any of 1 to 4)\n";
    cin >> category;
    if (category < 1 || category > 4)
        goto START;
    if (category == 1) wage = 12.5;
    if (category == 2) wage = 15.5;
    if (category == 3) wage = 20.5;
    if (category == 4) wage = 25.5;
    cout << "Enter hours worked\n";
    cin >> hours;
    if (hours <= 40)
        pay = wage*hours;
    if (hours > 40)
        pay = wage*40 + 1.5*wage*(hours-40);
    cout << "Pay = " << pay << endl ;
    return 0;
}
```

ပုံ (၃။၉)

JII ဒီ program ကို ကျွန်တော်ရှင်းမပြတော့ပါဘူး။ run ထားတာကို ပုံ (၃၁၀) မှာဖော်ပြထားပါတယ်။ စာဖတ်သူကိုယ်တိုင် data ပြောင်းထည့်ပြီး run ကြည့်ပါ။ program လမ်းတွောင်း trace လုပ်ကြည့်ပါ။ မခက်ပါဘူး။



ပုံ (၃၁၀)

၃.၃ The if-else Statement

OII program တစ်ခုရေးတဲ့အခါမှာ ရွေးစရာလမ်းတွောင်း (2) ခုရှိလာရင် if-else statement ကိုသုံးလို့ပါတယ်။ if နောက်က <test expr> ဟာမှန်တယ်ဆိုရင် <statement1> ကိုဖြေရှင်းပြီး if-else အပြင်ကိုထွက်သွားမှာဖြစ်ပါတယ်။ <test expr> မှားနေရင် <statement2> ကို execute လုပ်မှာပါ။ statement တွေဟာ တစ်တွောင်းထက်ပိုလာရင် brace { } တွေနဲ့ပိတ်ပေးရပါမယ်။ if-else statement ရဲ့ပုံစံက အခုလိုပါ။

```
if ( <test expr> )
    <statement1>;
else
    <statement2>;
```

ကောင်းပြီး ပုံ (၃၁၀) မှာဖော်ပြထားတဲ့ Ex306.cpp program ကနေ if-else statement အသုံးပြု

နည်းကိုလေ့လာကြည့်ရအောင်။ ဒါ program ဟာ lowercase letter တစ်ခုကို uppercase ဖြစ်အောင်ပြောင်းလေ့တဲ့ program ဖြစ်ပါတယ်။

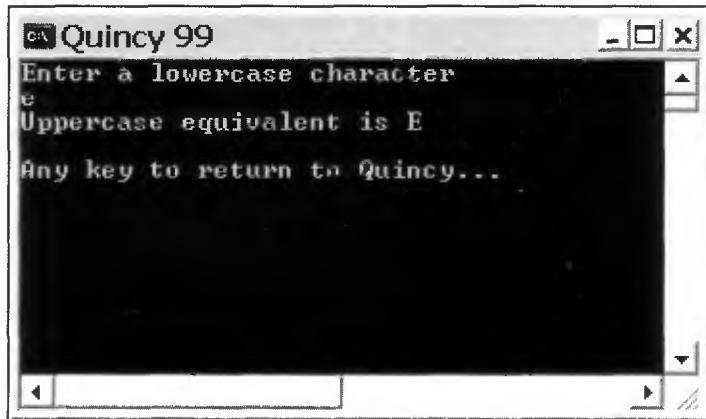
```
// Listing 3.6: This program reads a lowercase character,  
// converts it to uppercase and then writes out  
// the uppercase equivalent.  
#include <iostream>  
  
int main()  
{  
    char lower, upper;  
  
    cout << "Enter a lowercase character\n";  
    cin >> lower;  
    if (lower >= 'a' && lower <= 'z')  
        upper = 'A' + lower - 'a';  
    else  
        upper = lower;  
    cout << "Uppercase equivalent is " << upper << endl;  
    return 0;  
}
```

ပုံ (၃၀ ၁၁)

J Ex306.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- program စစချင်းမှာ lower နဲ့ upper ဆိုတဲ့ variable (2) ခုကို character တွေထိုကြော်ပါတယ်။ Enter a lowercase character ဆိုတဲ့ prompt တစ်ခု ကွန်ပူးတာမှာ ပေါ်လာတဲ့အခါ lower အတွက် data ကို e လိုခိုက်ထည့်ပါမယ်။ ဒါဆိုရင် lower = 'e' ဖြစ်သွားပါပြီ။
- if (lower >= 'a' && lower <='z') ဆိုတဲ့ relational expression မေးခွန်းမှာ lower က 'a' ထက်တိုးပြီး 'z' ထက်ငယ်ပါတယ်။ ဒီတော့အဖြေက TRUE ပေါ့။ upper = 'A'+lower-'a' ဖြစ်သွားပါပြီ။ upper ရဲ့ ASCII တန်ဖိုးကို upper = 65+101-97 =69 လိုတွက်လိုရပါတယ်။ ASCII တန်ဖိုး 69 က uppercase letter E ကိုဆိုလိုတာပါ။ ပြီးရင် else အောက်ကလိုင်းကို ကျော်ဆင်းသွားပါပြီ။

cout << "Uppercase equivalent is :\n" << upper සිදු statement ගෝදු
 ඇපු E තුළු පෙන්වා ඇතුළු ඇසු (Q. 1) මා program run ප්‍රතාස්ථානයේ නිශ්චල්‍ය පිහිටුව වූ නිශ්චල්‍ය පිහිටුව වූ නිශ්චල්‍ය පිහිටුව වූ



දී (Q. 1)

A Temperature Conversion Program

o Ex307.cpp program භා degree Fahrenheit ගණ Celsius ක්‍රියෝගයේ පෙෂීම් වලදී degree Celsius ගණ Fahrenheit ක්‍රියෝගයේ පෙෂීම් වූ program තක් ඇතිවයි. flag පැවත්වා ඇතුළු පෙෂීම් වලදී degree Celsius ගණ Fahrenheit ක්‍රියෝගයේ පෙෂීම් වූ ඇතුළු පෙෂීම් වූ 2 පැවත්වා ඇතුළු පෙෂීම් වූ ඇතුළු පෙෂීම් වූ ඇතුළු පෙෂීම් වූ ඇතුළු පෙෂීම් වූ ඇතුළු පෙෂීම් වූ

JII දී (Q. 1) මා program run ප්‍රතාස්ථානයේ flag පැවත්වා ඇතිවයි. flag පැවත්වා ඇතුළු data නිශ්චල්‍ය පෙෂීම් වූ degree Celsius = 100 ක් නිශ්චල්‍ය පෙෂීම් වූ Fahrenheit = 212 නිශ්චල්‍ය පෙෂීම් වූ display ලද ප්‍රතාස්ථානයේ පෙෂීම් වූ

```
// Listing 3.7: A program that gives the user the option of
// converting Fahrenheit to Celsius or Celsius to Fahrenheit.
#include <iostream>

int main()
{
    int      flag;
    double   tempr;

    cout << "\nType 1 to convert Fahrenheit to Celsius : "
        << "\nType 2 to convert Celsius to Fahrenheit : "
        << endl;
    cin >> flag;

    if (flag == 1)  {
        cout << "Enter temperature in deg Fahrenheit : ";
        cin >> tempr;
        cout << "Celsius = " << (tempr-32)/1.8 << endl;
    }
    else  {
        cout << "Enter temperature in deg Celsius : ";
        cin >> tempr;
        cout << "Fahrenheit = " << 1.8*tempr+32 << endl;
    }
    return 0;
}
```

Ø (2.09)

```
Quincy 99
Type 1 to convert Fahrenheit to Celsius :
Type 2 to convert Celsius to Fahrenheit :
2
Enter temperature in deg Celsius : 100
Fahrenheit = 212
Any key to return to Quincy...
```

Ø (2.09)

၃.၄ Nested if Statement

၁။ nested if-else statement ဟာ if statement အောက်မှာပဲအကျိုးဝင်ပါတယ်။ ရွှေ့ပိုင်းမှာတုန်းက ကျွန်ုတ်တို့လေ့လာခဲ့တဲ့ if-else statement တွေမှာ ရွေးစရာလမ်း (2) ခုပဲရှိတယ်လို့ပြောခဲ့ပါပြီ။ if နောက် `<test expr>` ဟာ TRUE ဆိုရင်သူ့နောက်က `<statement>` ကိုဖြေရှင်းမယ်၊ FALSE ဆိုရင် else နောက်က အလုပ်ပဲလုပ်မှာပါ။ ဒါကြောင့်မို့သွားစရာလမ်း (2) ခုပဲရှိတာလို့ပြောတာပါပဲ။ တစ်ကယ်လို့ သွားစရာလမ်းတွေ အများကြီးကို create လုပ်ချင်ရင် nested if-else statement တွေကိုအသုံးပြုရင်ရပါတယ်။ မှားမှာစိုးရင် brace တွေကို ပိုပိုလိုလို ထည့်ပေးပါ။ ပုံ (၃. ၁၅) မှာဖော်ပြထားတဲ့ Ex308.cpp program ဟာဆိုရင် complex nested if statement တွေ အသုံးပြုပုံပြန်ည်းကို နှုန်းရေးပြထားတာဖြစ်ပါတယ်။

၂။ Ex308.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- program စစချင်းမှာ num1 နဲ့ num2 တို့ကို integer variable တွေလို့ကြော်ပါတယ်။ Enter two numbers လို့ prompt ပေါ်လာတဲ့အခါကျရင် num1 တန်ဖိုးကို 13 လို့ရှိက ထည့်ပေးမယ်ဆိုပါစို့။ ဒါဆိုရင် num1 = 13 ဖြစ်သွားပါပြီ။ num2 တန်ဖိုးကိုတောင်းတဲ့အခါမှာ 4 ကိုရှိက်ထည့်ပေးပါမယ်။
- program က အပြင်ဘက်အကျခုံး if statement မှာ num1= 13 က num2= 4 ထက်ကြီးလား၊ ညီလားလို့မေးပါတယ်။ `<test expr>` ဟာ TRUE ဖြစ်တဲ့အတွက် သူ့အောက် if statement ကိုဆက်ပြီး execute လုပ်ပါတယ်။ အလယ်ပိုင်းကျတဲ့ if statement မှာလည်း num1 = 13 က num2 = 4 နဲ့ ညီလားလို့ထပ်မေးပါတယ်။ ဒါ <test expr> က FALSE ဖြစ်နေပါတယ်။ ဒါတော့ အတွင်းဘက်အကျခုံး if-else ကိုကျော်ပြီး ဒုတိယ if-else ခဲ့နောက် `<statement>` ကို execute လုပ်ပါပြီ။ They are not evenly divisible. ဆိုတဲ့စာကြောင်းကွန်ပူးတာမှာ display လုပ်ပြပါလိမ့်မယ်။
- အခုဆိုရင် အပြင်ဘက်အကျခုံး if-else ခဲ့အလုပ်ပြီးသွားပါပြီ။ ပြီးရင် closing brace () ဆီကိုရောက်သွားတဲ့အတွက် program ပြီးသွားပါပြီ။

၃။ ပုံ (၃. ၁၆) မှာ Ex308.cpp program ကို run ပြထားပါတယ်။ num1 = 13၊ num2 = 4 ဖြစ်ရင် They are not evenly divisible. ဆိုတဲ့ message ကို display လုပ်ပြမှာပါ။ num1 = 16၊ num2 = 4 ဖြစ်ရင် ဘယ်လို့ message ကို display လုပ်ပြမလဲ။ ခန်းမှန်းကြည့်ပါ။ ပြီးရင် program run ကြည့်ပြီး အဖြတ်က လို့ရပါတယ်။

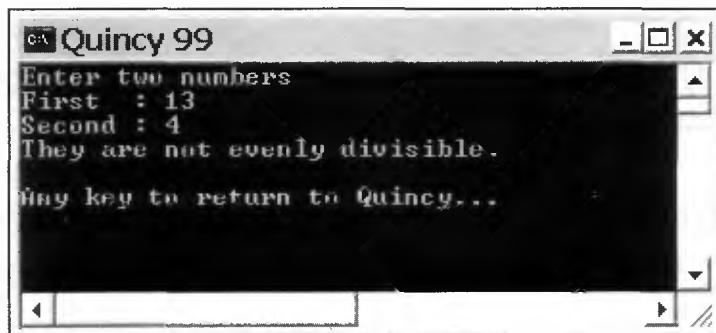
```
Ex308.cpp

// Listing 3.8: This program demonstrates a complex,
// nested if statement
#include <iostream>

int main()
{
    int num1, num2;

    cout << "Enter two numbers\nFirst : ";
    cin >> num1;
    cout << "Second : ";
    cin >> num2;
    if (num1 >= num2)
    {
        if (num1 % num2 == 0)
        {
            if (num1 == num2)
                cout << "They are the same.\n";
            else
                cout << "They are evenly divisible.\n";
        }
        else
            cout << "They are not evenly divisible.\n";
    }
    else
        cout << "The second one is larger.\n";
    return 0;
}
```

◊ (P. 09)



◊ (P. 06)

၄။ Ex308.cpp program မှာ brace တွေကိုဖြတ်ပြီး ပုံ (၃။ ၁၇) မှာပြထားတဲ့အတိုင်း ပြင်ရေးမယ်ဆိုရင် လည်း run လိုဂါတယ်။ အဖြေလည်းအတူတူရမှာပါ။ စာဖတ်သူကိုယ်တိုင်လက်တွေ့ run ကြည့်ပါ။

```
#include <iostream>

int main()
{
    int num1, num2;

    cout << "Enter two numbers\nFirst : ";
    cin >> num1;
    cout << "Second : ";
    cin >> num2;

    if (num1 >= num2)
        if (num1 % num2 == 0)
            if (num1 == num2)
                cout << "They are the same.\n";
            else
                cout << "They are evenly divisible.\n";
        else
            cout << "They are not evenly divisible.\n";
    else
        cout << "The second one is larger.\n";
    return 0;
}
```

ပုံ (၃။ ၁၇)

၅။ ဒါမေမဟုတ် program logic ကိုအခုလိုပြင်ရေးပြီး run ရင်လည်း အဖြေအတူတူရမှာပါ။ ဒါ program က compact စိနှစ်တယ်လို့ထင်ပါတယ်။ ပုံ (၃။ ၁၈) ကိုကြည့်ပါ။

၆။ Ex308B.cpp program ကို run ကြည့်ရင် ပုံ (၃။ ၁၉) မှာပြထားတဲ့အတိုင်းပြင်ရမှာပါ။ First = 15 နဲ့ Second = 100 လို့ data ထည့်ပေးရင် The second one is larger. ဆိုတဲ့အဖြေကို display လုပ်ပြပါလိမ့်မယ်။

```
Ex308B.cpp

#include <iostream>

int main()
{
    int num1, num2;

    cout << "Enter two numbers\nFirst : ";
    cin >> num1;
    cout << "Second : ";
    cin >> num2;

    if (num1 < num2)
        cout << "The second one is larger.\n";
    else if (num1 % num2 != 0)
        cout << "They are not evenly divisible.\n";
    else if (num1 != num2)
        cout << "They are evenly divisible.\n";
    else
        cout << "They are the same.\n";

    return 0;
}
```

Ø (૧.૦૧)

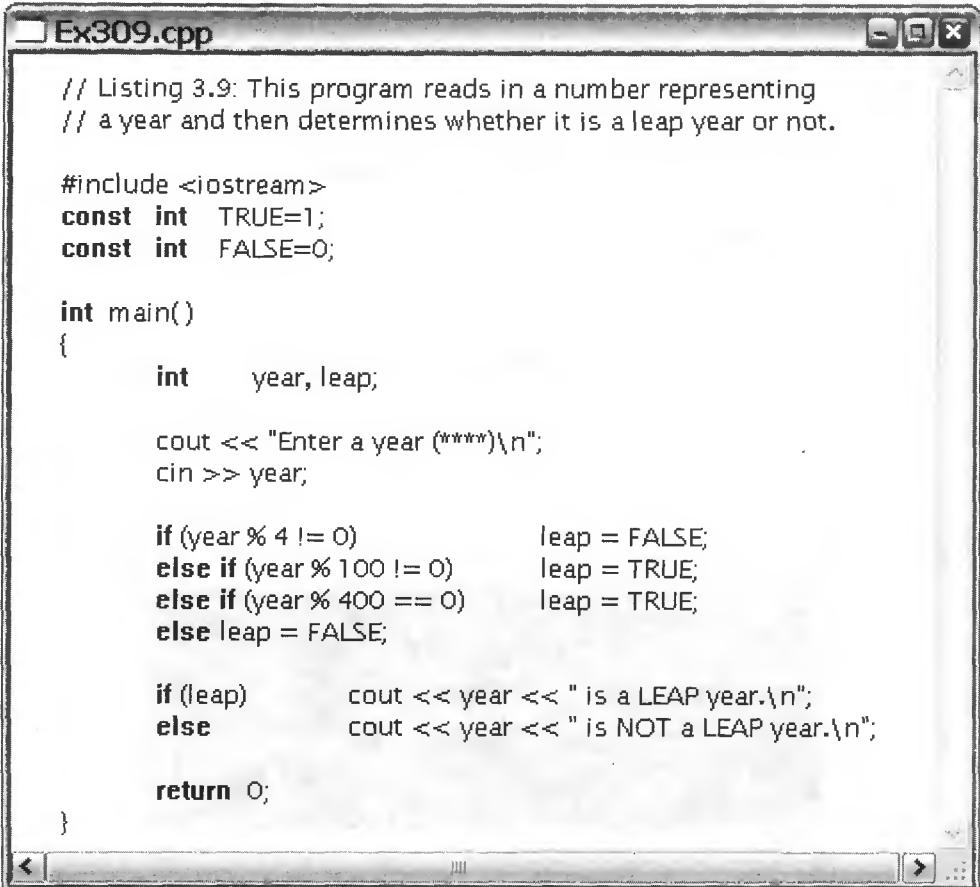
```
Quincy 99
Enter two numbers
First : 15
Second : 100
The second one is larger.

Any key to return to Quincy...
```

Ø (૧.૦૨)

Leap Year Test

၁။ အခုတင်ပြုမယ့် Ex309.cpp program ဟာဆိုရင် အင်လိုင် ဘယ်ခုနှစ်ဟာ ဝါထပ်လား၊ မထပ်ဘူးလား ဆိုတာကို စိစစ်ပေးမှာပါ။ စစ်နည်းက ခုနှစ်ကို 4 နဲ့အရင်တားကြည့်ပါ။ မပြတ်ဘူးဆိုရင် အဲဒီနှစ်ကတိမထပ်ပါဘူး။ 4 နဲ့တားလိုပြတ်ရင် ဝါထပ်ချင်ထပ်နိုင်ပါတယ်။ ထပ်စစ်ရပါဘူးမယ်။ 100 နဲ့ထပ်တားကြည့်ပါ။ တားလိုပြတ်ရင် အဲဒီနှစ်ဟာ သေခာပေါက် ဝါထပ်ပါတယ်။ ပြတ်တယ်ဆိုရင် အဲဒီနှစ်ဟာဝါထပ်လား၊ မထပ်လားမသေခာပြန်တော့ဘူး။ 400 နဲ့ ထပ်တားကြည့်ပါး။ ပြတ်တယ်ဆိုရင်ဝါထပ်ပါတယ်။ မပြတ်ရင်ဝါမထပ်ပါဘူး။ အဲဒီနည်းနဲ့ program ကိုရေးထားတာ ဖြစ်ပါတယ်။ ပုံ (၃၀၂၂) က code statement တွေကိုလေ့လာကြည့်ပါ။



```
Ex309.cpp

// Listing 3.9: This program reads in a number representing
// a year and then determines whether it is a leap year or not.

#include <iostream>
const int TRUE=1;
const int FALSE=0;

int main()
{
    int      year, leap;

    cout << "Enter a year (****)\n";
    cin >> year;

    if (year % 4 != 0)              leap = FALSE;
    else if (year % 100 != 0)        leap = TRUE;
    else if (year % 400 == 0)        leap = TRUE;
    else leap = FALSE;

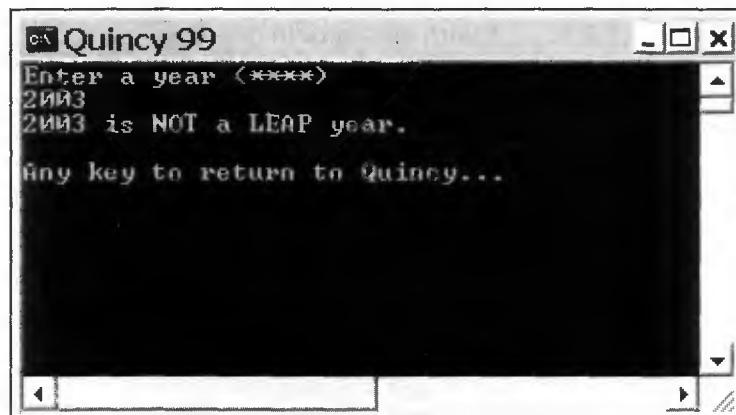
    if (leap)                      cout << year << " is a LEAP year.\n";
    else                            cout << year << " is NOT a LEAP year.\n";

    return 0;
}
```

ပုံ (၃၀၂၂)

Ex309.cpp program ကိုလေလာကြည့်မယ်ဆိုရင်

- 1998 ခုနှစ်ဟာ ဝါမထပ်ပါဘူး။ ဘာပြုလိုလဲဆိုတော့ 1998 ကို 4 နဲ့စားရင်မပြတ်လိုပါပဲ။ 2000 ခုနှစ်က ဝါထပ်ပါတယ်။ ဘာပြုလိုလဲဆိုတော့ 2000 ကို 4 နဲ့စားတာပြတ်တယ်။ 100 နဲ့ထပ်စားတာပြတ်သေးတယ်။ ဒီတော့ 400 နဲ့ထပ်စားတယ်။ ပြတ်သေးတာပဲ။ ဒါဆိုရင် 2000 ခုနှစ်ဟာ သေချာဝါထပ်တာပေါ့။
- 5000 ခုနှစ်ကျတော့ ဝါမထပ်တော့ဘူး။ ဘာပြုလိုလဲဆိုတော့ ပထမ 4 နဲ့စားတာပြတ်တော့ 100 နဲ့ထပ်စားတော့ပြတ်တယ်။ ဒါပေမယ့် 400 နဲ့စားတော့မပြတ်တော့ဘူး။ ဒါကြောင့်မို့ 5000 ခုနှစ်ဟာ ဝါမထပ်ဘူးလို့ပြောတာပါ။ ဒီစစ်နည်းတွေမှာ nested if-else statement တွေကို အသုံးပြုထားပါတယ်။ Ex309.cpp program ကို run ကြည့်ရင် ပဲ (၃.၂၁) မှာပြထားတဲ့အတိုင်းမြင်ရမှာပါ။ year= 2003 လိုထည့်ပေးရင် 2003 is NOT a LEAP year. ဆိုတဲ့အဖြေကို display လုပ်ပြပါလိမ့်မယ်။



ပဲ (၃.၂၁)

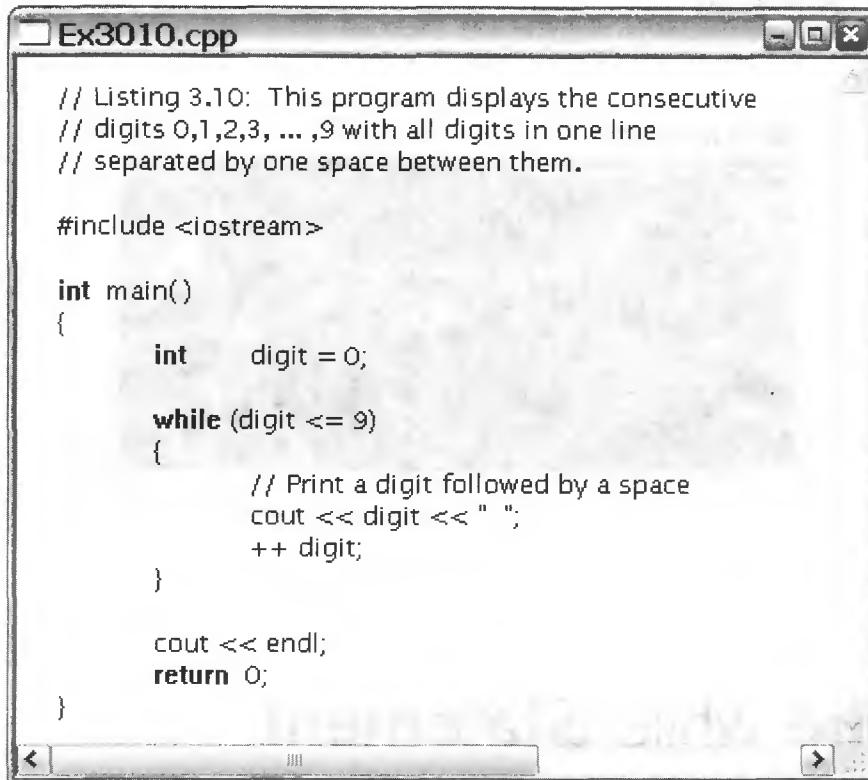
၃.၅ The while Statement

- ၁။ C++ program မှာ loop တစ်ခုကို ထပ်ခါတစ်လဲပတ်ရတော့မယ်ဆိုလိုရှိရင် while statement ကိုအသုံးပြုလိုရပါတယ်။ while statement ရဲ့ပုံစံက အခုလိုပါ။

while (<test expr>) <statement1> ;

ဒီပုံစံဟာဆိုရင် ကွန်ပျူတာမှာ စစချင်း while ရဲနောက်ကကွင်းထဲက <test expr> ကို test လုပ်ပါတယ်။ သူ့ရဲတန်ဖိုးက nonzero (TRUE) ဖြစ်ခဲ့ရင် ကွန်ပျူတာက <statement1> ကို execute လုပ်မှုပါ။ တစ်ကယ်လို <test expr> ရဲတန်ဖိုးက zero (FALSE) ဖြစ်သွားမယ်ဆိုရင် while loop ကို execute မလုပ်တော့ပါဘူး။ statement တွေကအများကြီးဆိုရင် brace တွေနဲ့ပိတ်ပေးဖို့မမေ့ပါနဲ့။

JL အခုတင်ပြုမယ့် Ex3010.cpp program ဟာဆိုရင် 1 ကနေ 9 အထိကော်း (9) လုံးကို စာတစ်ကြောင်း တည်းမှာပေါ်လာအောင် while statement အသုံးပြုပြီးရေးထားတာဖြစ်ပါတယ်။ ဝက်နှုံးတစ်လုံးနဲ့တစ်လုံးကြားမှာ space တစ်ခုခြားပေးထားပါတယ်။ ပုံ (R. JL) မှာရေးပြထားတဲ့ code statement တွေကိုလေ့လာကြည့်ပါ။



```
// Listing 3.10: This program displays the consecutive
// digits 0,1,2,3, ... ,9 with all digits in one line
// separated by one space between them.

#include <iostream>

int main()
{
    int      digit = 0;

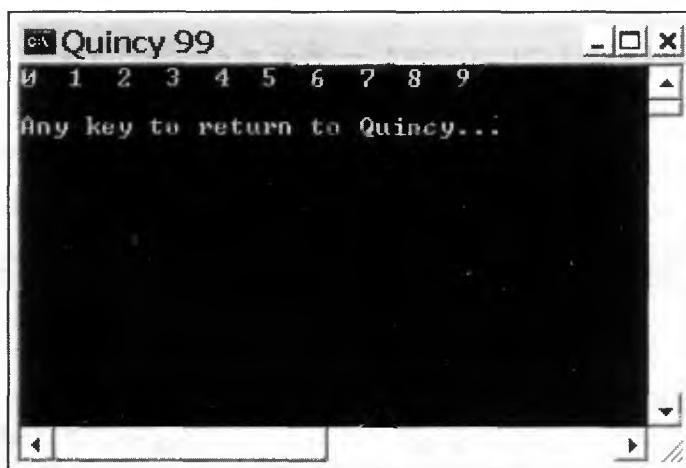
    while (digit <= 9)
    {
        // Print a digit followed by a space
        cout << digit << " ";
        ++ digit;
    }

    cout << endl;
    return 0;
}
```

ပုံ (R. JL)

Ex3010.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- program စစချင်းမှာ digit ကို integer variable လိုကြော်ပြီး သုညဖြစ်အောင် initialize လုပ်ထားပါတယ်။ while (digit <= 9) ရဲ့အဓိပ္ပာက digit ဟာ 9 ထက်ငယ်မယ်၊ ညီမယ် ဆုရင် brace { } တွေထဲက multiple statement တွေကို execute လုပ်မှာပါ။ digit=0 ဖြစ်တဲ့အတွက် ကွန်ပူးတာက while loop ထဲဝင်လာပါပြီ။
- cout << digit << " "; ဆုတဲ့ statement ကြောင့်ကွန်ပူးတာမှာ 0 <ကွက်လပ်> လို print လုပ်တာကိုမြင်ရပါလိမ့်မယ်။ ++digit လိုအေးထဲတဲ့အတွက် digit တန်ဖိုးကို ကွန်ပူးတာက 1 တိုးပေးလိုက်ပါပြီ။ ပြီးရင် closing brace () ကိုတွေ့တဲ့အခါမှာ while loop အစကို ပြန်တက်သွားမှာပါ။
- အခုခံစ်ခါ digit = 1 ဖြစ်နေပါတယ်။ digit = 1 က 9 ထက်ငယ်ပါတယ်။ ဒီတော့ while loop ထဲ ဒုတိယအကြိမ်ပြန်ဝင်ပြီး 1 <ကွက်လပ်> ကိုဆက်ပြီး print လုပ်ပါတယ်။ ပြီးရင် digit ကို တစ်တိုးပါတယ်။ ဒီနည်းအတိုင်း digit တန်ဖိုး 9 မကျော်မချင်း while loop ကိုပတ်နေမှာပါပဲ။ တစ်ခုချင်းမှာ digit = 10 ဖြစ်သွားတဲ့အခါ Iloop ထဲက အပြီးထွက်လာပြီးတော့ program လည်း ရပ်သွားပါလိမ့်မယ်။ program run ထားတာကို ပုံ (၃၁၂၂) မှာဖော်ပြထားပါတယ်။



ပုံ (၃၁၂၂)

- ဒီ program ကို ပိုပြီးကျစ်ကျစ်လစ်လစ်ဖြစ်အောင် အခုလိုပြောင်းရေးလိုပါတယ်။ (++) increment operator သုံးသွားပုံကို ပုံ (၃၁၂၅) က Ex3010A.cpp program ကိုလေ့လာကြည့်ပါ။

```

#include <iostream>

int main()
{
    int digit = 0;

    while (digit <= 9) cout << digit++ << " ";
    cout << endl;

    return 0;
}

```

ဗု (၃။ ၂၄)

Evaluating PI Series

၁။ အခြားတစ်ချို့မယ် Ex3011.cpp program ဟာဆိုရင် π ရဲတန်ဖိုးကို $\pi/4 = 1 - 1/3 + 1/5 - 1/7 + 1/9 - \dots$ series ကိုအသုံးပြုပြီး term တွေ အများကြီးပေါင်းခိုင်းမှာဖြစ်ပါတယ်။ ဗု (၃။ ၂၅) မှာရေးထားတဲ့ program ကိုလေ့လာကြည့်ပါ။ ဒါ series က slow converging ဖြစ်ပေမယ့် C++ compiler ဟာမြန်တဲ့အတွက် term (3) သန်းအထိ ပေါင်းခိုင်းတာကို အလွယ်တကူတန်းပေါင်းပေးနိုင်ပါတယ်။ ဒါ program ဘယ်လိုအလုပ်လုပ်သွားလိုတာ စာဖတ်သူကိုယ်တိုင် trace လုပ်ကြည့်ပါ။ while statement ကိုပိုနားလည်သွားပါလိမ့်မယ်။

၂။ Ex3011.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင် စစ်ဆေးမှာ pi = 0 လို့ initialize လုပ်ပြီး n = 30000000 ကိုရိုက်ထည့်ပေးပါတယ်။ term တွေကိုပေါင်းတဲ့အခါမှာ (2) စုချင်းပေါင်းသွားပေးပါတယ်။ ပထမအစုံ အတွက် counter (i) ကို 1 လို့သတ်မှတ်ပေးရင် ဒုတိယအစုံအတွက် counter (i) ကို i + 4 = 5 လို့တွက် ပေးရပါမယ်။ နောက်အစုံတွေကျရင်လည်း (4) စီတိုးသွားမှာပါပဲ။ ဗု (၃။ ၂၆) မှာ program run ပြထားပါတယ်။ pi = 3.141592 ရပါတယ်။ အဖြေမြန်က $\pi = 3.141592$ ဖြစ်ပါတယ်။

```
// Listing 3.11: This program evaluates the value
// of PI using the series
//      PI/4 = 1 - 1/3 + 1/5 - 1/7 + 1/9 - ...
#include <iostream>
int main()
{
    double n, i=1, pi=0;

    cout << "Enter n : ";
    cin >> n;
    while (i <= n)
    {
        pi += 1/i - 1/(i+2);
        i += 4;
    }
    cout.precision(10);
    cout << "PI = " << 4*pi << endl;
    return 0;
}
```

◊ (P. JG)

```
Quincy 99
Enter n : 3000000
PI = 3.141591987
Any key to return to Quincy...
```

◊ (P. JG)

Counting Characters and Words

၁။ အခုတင်ပြုမယ့် Ex3012.cpp program ဟာဆိုရင် keyboard ကနေ ကျွန်တော်တိုရှိက်ထည့်လိုက် တဲ့စာကြောင်းထဲမှာ character ဘယ်နှစ်လုံး၊ word ဘယ်နှစ်လုံးပါလဲဆိုတာ ရောွက်ပေးမယ့် program ပါပဲ။ ဒီ program မှာ getche() function ကိုအသုံးပြုလိုရအောင် <conio.h> header ဖိုင်ကို ထည့်ပေးထားပါတယ်။ program ဘယ်လိုအလုပ်လုပ်သွားလဲဆိုတာကို စာဖတ်သူကိုယ်တိုင် ပုံ (၃။၂) မှာရေးထားတဲ့ code တွေကို trace လုပ်ကြည့်ပါ။ while statement ကိုလည်း ပိုနားလည်သွားပါလိမ့်မယ်။

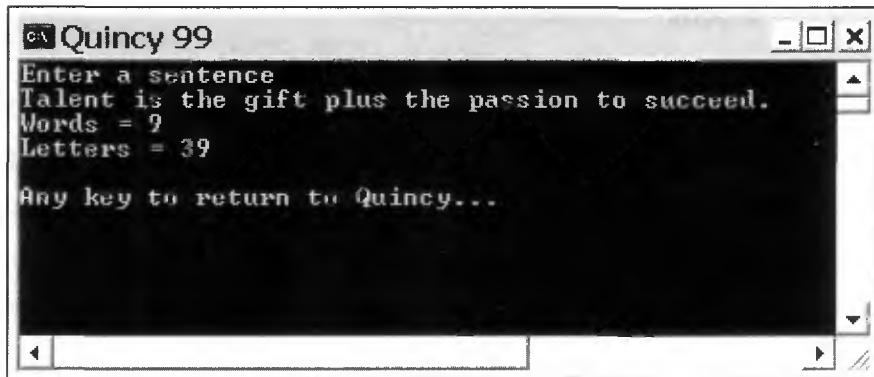
```
// Listing 3.12: This program counts characters
// and words that you typed in

#include <iostream>
#include <conio.h> // For getche() function

int main()
{
    int     chcount= 0, wdcount=1;
    char   ch='a';

    cout << "Enter a sentence\n";
    while (ch != '\r')
    {
        // While ch != ENTER
        ch = getche();
        if (ch == ' ') wdcount++;
        else chcount++;
    }
    cout << "\nWords = " << wdcount << endl;
    cout << "Letters = " << (chcount - 1) << endl;
    return 0;
}
```

ပုံ (၃။၂)



ပုံ (၃၁၂)

၃.၆ The do-while Statement

၁။ C++ program မှာ loop တစ်ခုကို while statement ကိုထုတေသနပြီးပတ်သင့်မပတ်သင့်ကို loop စစ်ဆေးမှာပဲ စစ်ဆေးပါတယ်။ တစ်ခါတစ်ရဲ ဒီလို test မျိုးကို loop ရွှေ့နောက်ဆုံးကျမှ စစ်ဆေးတာမျိုးလည်းရှိပါတယ်။ အဲဒါဆိုရင် do-while statement ကိုအသုံးပြုရမှာပါ။ do-while ရုပ်စံက ဒီလိုပါ။

```

do {
    <statement1>;
    <statement2>;
    ...
} while ( <test expression> );

```

စစ်ဆေးမှာ loop ထဲကိုအတားအဆီးမရှိ do ကိုကျဉ်ပြီး loop body ထဲ ဝင်လို့ရပါတယ်။ loop body ထဲက statement တွေအားလုံးကို execute လုပ်ပြီးသွားရင် while နဲ့တွေ့မှာပါ။ ဒီခါကျဉ် while နောက် <test expr> ကိုစစ်လိုက်လို့ TRUE ဆိုရင် do-while loop ကို ဒုတိယအကြိမ်ပတ်ခွွဲပြုမှာပါ။ FALSE ဆိုရင်ပတ်ခွွဲပြုမှာမဟုတ်ပါဘူး။ while အောက်က next line တွေဆီကိုတန်းဆောင်းရမှာပါ။ ဒါဟာ do-while control statement ရဲသေားတရားပါပဲ။ ကောင်းပြီ။ do-while statement ကို လက်တွေ့အသုံးပြုကြည့်ရအောင်။

အခုတင်ပြုမယ့် Ex3013.cpp program ရဲ့ရည်ရွယ်ချက်က ဝက်န်းတစ်လုံးပြောလိုက်တာနဲ့ ကွန်ပျူတာက အဲဒီဝက်န်းရဲ့ factorial ကိုရှာပေးမှာပါ။ ပုံ (၃။၂၉) မှာရေးထားတဲ့ code statement တွေကို trace လုပ်ကြည့်ပါ။

```
// Listing 3.13: This program computes the factorial of
// a positive integer n. The factorial of n represented
// by n! is defined as
//      n! = n(n-1)(n-2)(n-3)(n-4) .....(3)(2)(1)

#include <iostream>
int main()
{
    long double i= 2, n, nfac= 1;

    cout << "Enter n : ";
    cin >> n;
    do
        nfac *= i++;
    while (i <= n);
    cout << "Factorial of " << n << " is " << nfac << endl;

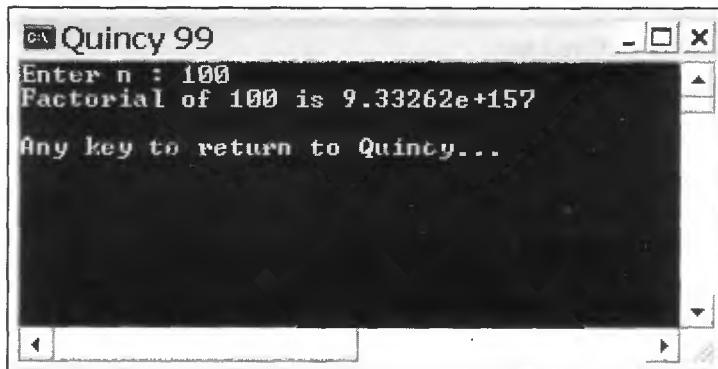
    return 0;
}
```

ပုံ (၃။၂၉)

Ex3013.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင် စစခင်းမှာ

- variable တွေကို long double လို့ declare လုပ်ပြီး i = 2 နဲ့ nfac = 1 လို့ initialize လုပ်ပါတယ်။ i က counter ပါ။ n! ရဲ့အဓိပ္ပာကို နောက်ဆုံးမှာ nfac ထဲရောက်အောင်တွက်မှာပါ။ n ကတေသ့ factorial ရှုမယ့်ဝက်န်းပါပဲ။ long double ဆိုရင် 3.4 e-4932 ကနေ 1.1e+4932 အထိ ကြိုက်ရာဝက်န်းတစ်ခုကို အသုံးပြုလို့ရပါတယ်။ Enter n : လိုက်ပျူတာက prompt လုပ်တဲ့အခါကျရင် 5 ကိုရှိက်ထည့်မယ်ဆိုပါမို့။ ဒါဆိုရင် n = 5 ဖြစ်သွားပါပြီ။

စစချင်း do-while loop ထဲဝင်လာဖြီးတော့ $nfac = nfac * i = 1 * 2 = 2$ လိုက္ခန်ပျူးတာက တွက်ပါတယ်။ ပြီးတော့ရင် i ကိုတစ်တိုးပါတယ်။ do-while loop ကို $i > 5$ မဖြစ်မချင်းပတ်နေ မှာပါ။ $i = 6$ ဖြစ်လာတဲ့အခါကျရင် do-while loop ကို ဆက်မပတ်တော့ပါဘူး။ next line ကို ဆင်းလာဖြီးတော့ Factorial of 5 is 120 လိုအဖြေထုတ်ပေးပါလိမ့်မယ်။ $n = 100$ အတွက်လည်း run ကြည့်ပါ။ ပုံ (၃။ ၃၀) ကိုကြည့်ပါ။ ဒါဟာ do-while statement ကိုအသုံးပြုနည်းပါပဲ။



ပုံ (၃။ ၃၀)

Prime Factors of a Number

၁။ အခုတင်ပြုမယ့် Ex3014.cpp program ရဲ့ရည်ရွယ်ချက်က ဂဏန်းတစ်လုံးပေးထားရင် အဲဒီဂဏန်းရဲ့ ဆွဲတိန်းဂဏန်းတွေကို ရှာပေးတဲ့ program ဖြစ်ပါတယ်။ ပုံ (၃။ ၃၀) ကိုကြည့်ပါ။

၂။ Ex3013.cpp program ကိုလွှေလာကြည့်မယ်ဆိုရင်

■ program စစချင်း Enter a number လို prompt ပေါ်လာတဲ့အခါန်မှာ 105 လိုရှိက်ထည့်ပေး လိုက်ရင် $n = 105$ ဖြစ်သွားပါပြီ။ divisor = 2, adder = 1 လို initialize လုပ်ထားပါတယ်။ do-while loop ကိုစာတိပါပြီ။ if statement နဲ့ $n \% \text{divisor}$ ဟာသူညလားလိုမေးပါတယ်။ 105 ကို 2 နဲ့စာတိပါပြီ။ အကြွင်းသူညမရပါဘူး။ ဒါတော့ else နောက်ကအလုပ်ကိုပဲ ကွန်ပျူးတာက လုပ်မှာပါ။ divisor ကို 1 တိုးပေးသလို adder = 2 လို assign လုပ်ပါတယ်။ divisor = 3 ဖြစ်သွားပါပြီ။ ဒါပေမယ့် $n = 105$ က မပြောင်းသေးပါဘူး။

Ex3014.cpp

```

// Listing 3.14: This program prints a list of
// the prime factors of a number
#include <iostream>

int main()
{
    int divisor=2, adder=1, n;

    cout << "Enter a number\n";
    cin >> n;
    cout << "Prime factors of " << n << " is\n";
    do {
        if (n % divisor == 0)
        {
            n /= divisor;
            cout << divisor << " ";
        }
        else
        {
            divisor += adder;
            adder = 2;
        }
    } while (n/2 > 1);
    cout << '\n';

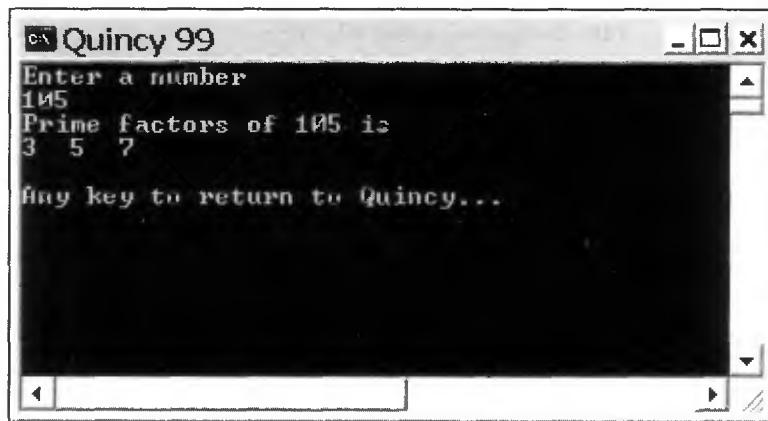
    return 0;
}

```

ပုံ (၃။၃၀)

- while ($n/2 > 1$) ဆိတဲ့အဓိပါယ်က $n/2$ ဟာ 1 ထက်ကြီးရင် do-while loop အစကိုပြန်သွားပါတဲ့။ $n/2 = 105/2 = 52$ ဆိတေသာ့ 1 ထက်ကြီးတာပေါ့။ do-while loop အစကို ပြန်သွားပါပြီ။ if ($n \% \text{divisor} == 0$) ဆိတေတက 105 ကို divisor= 3 နဲ့စားလိုပြောတယ်ဆိုရင် $n = n/\text{divisor} = 105/3 = 35$ လိုပြင်ရပါမယ်။ အဲဒီ 3 ကိုလည်း တွန်ပျောတာစကရင်မှာ display လုပ်ရပါမယ်။
- $n/2 = 35/2 = 17$ က 1 ထက်ကြီးတာမို့ do-while loop အစကိုပြန်သွားပါပြီ။ 35 ကို 3 နဲ့စားလိုပြောတယ်ဆိုရင် else နောက်က divisor = divisor+adder = 3+2 = 5 လိုပြင်ဖြီး

while ဆိတ်ရောက်လာပါတယ်။ $n/2 = 35/2 = 17$ ထက်ကြီးတာမို့ do-while loop အစား
ပြန်သွားပါတယ်။ 35 ကို 5 နဲ့စားလိုပြတ်ရင် $n = n/divisor = 35/5 = 7$ လိုပြင်ရပါမယ်။
အဲဒီ 5 ကိုလည်း display လုပ်ရပါမယ်။ အဲဒီတိုင်းဆက်လုပ်သွားရင် 105 ရဲ့ဆွဲကိန်းတွေဖြစ်တဲ့
3 + 5 + 7 တို့ကို အဖြေထုတ်လိုပါပြီ။ ပုံ (၃။၃၂) မှာ run ပြထားတဲ့ output ကိုဖြည့်ပါ။



ပုံ (၃။၃၂)

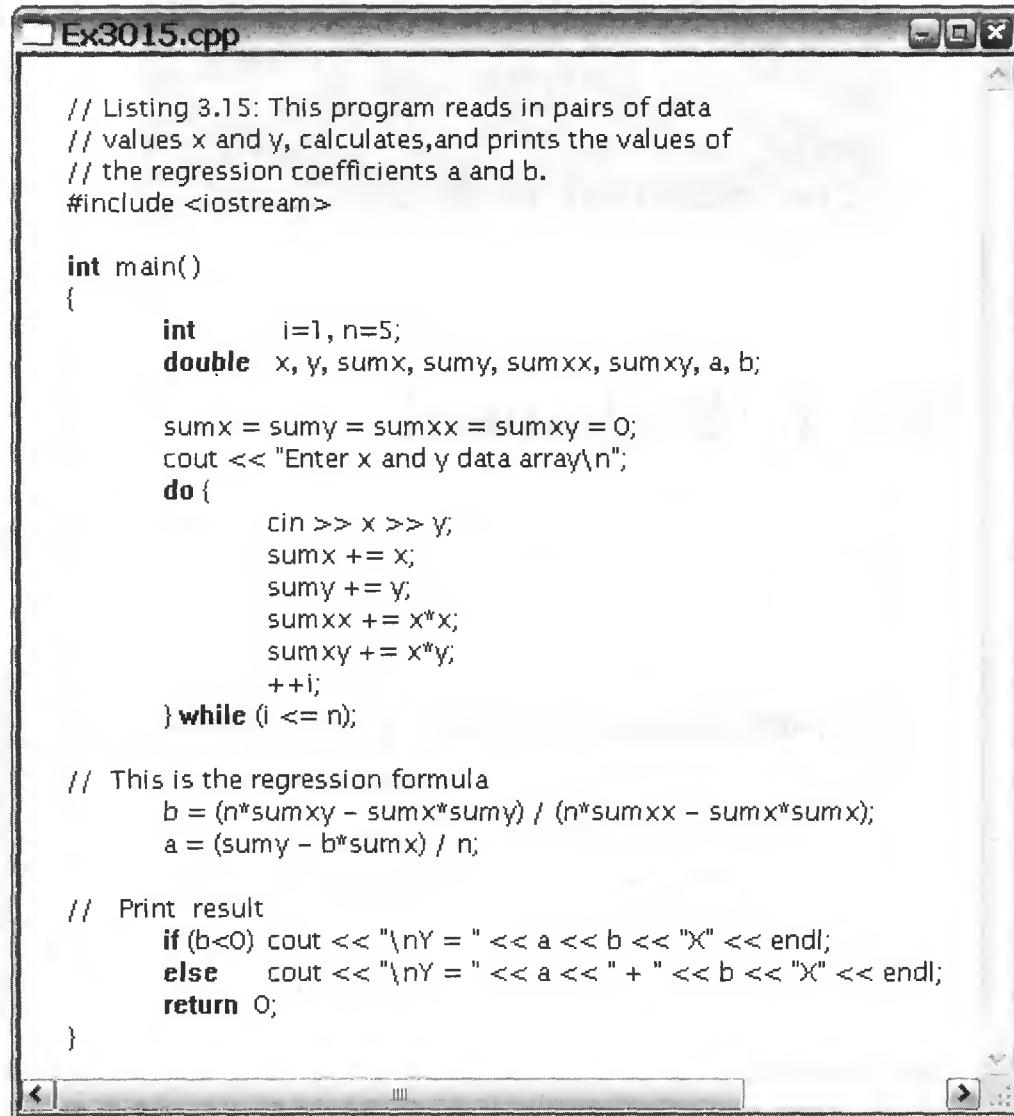
Linear Regression Method

၁။ အခုတင်ပြေမယ့် Ex3015.cpp program ဟာဆိုရင် ပေးထားတဲ့ x-y data array (5) စုကိုကိုယ်စားပြုတဲ့
error အနည်းဆုံး မျဉ်းဖြောင့်တစ်ကြောင်းရဲ့ equation ကိုရှာပေးမှာပါ။ ဒီညစာကို linear regression လုပ်တယ်
လိုပေါ်ပါတယ်။ တစ်ကယ်လို့ data point တွေဟာ မျဉ်းဖြောင့်ပေါ်မှာ အားလုံးကျနေမယ်ဆိုလိုရှိရင် equation
က ဘွက်တိမှန်နေမှာပါ။ ဒီမျဉ်းဖြောင့်ရဲ့ equation ကို $Y = a + bX$ လို့သတ်မှတ်မယ်ဆိုရင် $a \neq b$ တို့ကို

$$b = \frac{n (\sum xy) - (\sum x)(\sum y)}{n (\sum x^2) - (\sum x)^2}$$

$$a = \frac{\sum y - b(\sum x)}{n}$$

ဒုတိ **equation** တွေနဲ့ တွက်ယူလိုပါတယ်။ ဒီညီမျှမြင်းတွေမှာ n ဟာ x-y data pair အရေအတွက်ပါ။ အဥ
program မှာ n = 5 လိုအသေယူထားပါတယ်။ program ကို general ဖြစ်စေချင်ရင် n ကိုအရှင်ထားပေးပါ။
program အလုပ်လုပ်သွားပုဂ္ဂိုလ်မပြတော့ပါဘူး။ စာဖတ်သူကိုယ်တိုင် လေ့လာဖို့ချော်ထားခဲ့ပါမယ်။ လေ့လာကြည့်ပါ။



```

// Listing 3.15: This program reads in pairs of data
// values x and y, calculates, and prints the values of
// the regression coefficients a and b.
#include <iostream>

int main()
{
    int      i=1, n=5;
    double  x, y, sumx, sumy, sumxx, sumxy, a, b;

    sumx = sumy = sumxx = sumxy = 0;
    cout << "Enter x and y data array\n";
    do {
        cin >> x >> y;
        sumx += x;
        sumy += y;
        sumxx += x*x;
        sumxy += x*y;
        ++i;
    } while (i <= n);

    // This is the regression formula
    b = (n*sumxy - sumx*sumy) / (n*sumxx - sumx*sumx);
    a = (sumy - b*sumx) / n;

    // Print result
    if (b<0) cout << "\nY = " << a << b << "X" << endl;
    else     cout << "\nY = " << a << "+" << b << "X" << endl;
    return 0;
}

```

ဗျာ (၃၁ ၃၃)

```

Quincy 99
Enter x and y data array
2.1    2.9
6.22   3.83
7.55   5.98
11.5   6.31
13.7   7.88
Y = 1.94051 + 0.418735X
Any key to return to Quincy...

```

ပုံ (၃။၃၄)

၃.၇ The for Statement

၀။ for statement ဆိတာ C++ မှာ looping လုပ်ဖို့အတွက် အသုံးများဆုံး control statement တစ်ခုပါပဲ။ ဒီ statement ရဲပုံစံက အခဲ့ပါ။

```

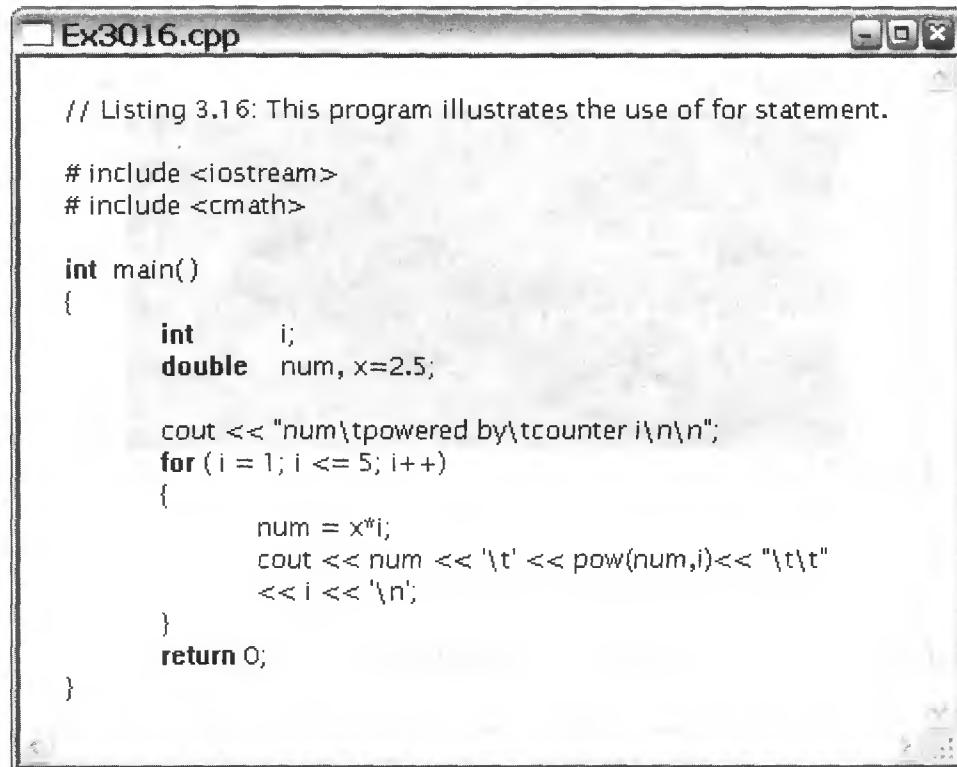
for ( [<initialization>] ; [<test expr>] ; [<increment>] )
{
    <statement1>;
    <statement2>;
    .....
}

```

ဒီပုံစံမှာ <initialization> expression ဟာ for loop ရဲစုံး counter ဖြစ်ပါတယ်။ initial index လို့
လည်းပြောလို့ရပါတယ်။ <test expr> က loop ကိုရွှေ့ဆက်ပတ်သင့် । မပတ်သင့်ဆိတာကို စီစစ်တဲ့
လုပ်ခဲ့ပါပဲ။ ဒီ conditional expression ဟာမှန်နေသမှု looping ဟာဆက်ပတ်နေမှာဖြစ်ပါတယ်။ <increment>
expression က counter ကို increment သို့မဟုတ် decrement လုပ်ပေးပါတယ်။ increment operator

decrement operator တွေကို for statement ထဲမှာထည့်သုံးလိုပါတယ်။ for(; ;) လိုပေးမယ်ဆိုရင် never ending loop ဖြစ်သွားမှာပါ။ infinite loop လိုလည်းခေါ်ပါတယ်။ ပြဿနာမတက်အောင် သတိထားကြည့်သုံးပါ။ for(; <condition>; <increment>) လိုလည်း ရေးလိုပါတယ်။

JII အခုတင်ပြုမယ့် Ex3016.cpp program ဟာဆိုရင် for loop ကိုအသုံးပြုပြီး ပေးရင်းကိန်းတစ်ခုဖြစ်တဲ့ num = 2.5 ကို (5) ကြိမ်တိတိ ထပ်ကိန်းတင်ပေးတာဖြစ်ပါတယ်။ counter ကို power တင်တဲ့ index အဖြစ် အသုံးပြုထားပါတယ်။ သူ့နဲ့ပတ်သက်တဲ့ pow() function ကိုအသုံးပြုလိုရအောင် <cmath> ဆိုတဲ့ math header ကိုထည့်ထားပါတယ်။ ပုံ (၃၁၃၅) ကိုကြည့်ပါ။



```
// Listing 3.16: This program illustrates the use of for statement.

#include <iostream>
#include <cmath>

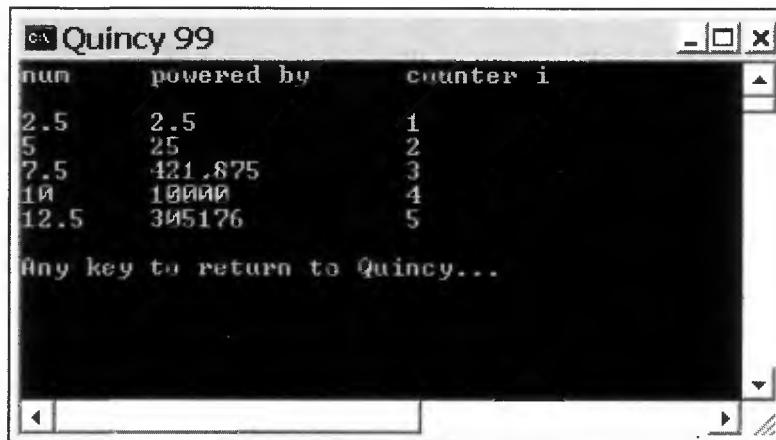
int main()
{
    int      i;
    double  num, x=2.5;

    cout << "num\tpowered by\tcounter i\n\n";
    for (i = 1; i <= 5; i++)
    {
        num = x*i;
        cout << num << '\t' << pow(num,i)<< "\t\t"
            << i << '\n';
    }
    return 0;
}
```

ပုံ (၃၁၃၅)

JII Ex3017.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင် စေချင်းမှာ

- counter အဖြစ်အသုံးပြုမယ့် variable i ကို integer လို့ declare လုပ်ထားပါတယ်။ num နဲ့ x တိုကို double လို့ကြော်ပါတယ်။ num i powered by နဲ့ counter i တိုကို tab escape sequence '\t' အသုံးပြုပြီး ခေါင်းစဉ်းမှာခြားရှိက်ခိုင်းပါတယ်။
- for loop ကို (5) ကြိမ်တိတိပတ်ခိုင်းပါတယ်။ စစချင်းမှာ num = x*i = 2.5*1 = 2.5 ကွန်ပူးတာကတွက်ယူလိုက်ပါတယ်။ နောက်ပြီး num=2.5 ကို powered by= pow(num, i) = pow(2.5, 1)= $2.5^1 = 2.5$ နဲ့ i = 1 လို့တွက်ယူပြီး ကွန်ပူးတာမှာ display လုပ်ပြပါ လိမ့်မယ်။ ပြီးရင်နောက်တစ်ကြောင်းကိုဆင်းပြီး for loop ကိုပတ်ညီးမှာပါ။ ဒါ program မှာ C++ ရဲ့ output format အသုံးပြုနည်းကိုပြထားပါတယ်။ Ex3016.cpp program ကို run မယ်ဆိုရင် ပုံ (၃၀.၃၆) မှာပြထားတဲ့အတိုင်းမြင်ရမှာပါ။



ပုံ (၃၀.၃၆)

Creating a Temperature Conversion Table

၁။ အခုံတင်ပြုမယ့် Ex3017.cpp program ဟာဆိုရင် -40 ဒီဂရီစင်တိဂရီတော် +40 ဒီဂရီစင်တိဂရီတော် အထိကို ယာရင်ဟိုက်အဖြစ် တစ်တွဲတည်း conversion table အဖြစ် create လုပ်ပေးတဲ့ program ဖြစ်ပါတယ်။ for loop ဘယ်လိုအလုပ်လုပ်သွားလဲဆိုတာကို လေ့လာကြည့်ရအောင်။

- program စစချင်း cel ကို signed int ၊ fah ကို float လို့ data type ကြော်ပါတယ်။ heading ကိုရေးပါတယ်။ တစ်ကြောင်းခြားခိုင်းပါတယ်။

Ex3017.cpp

```

// Listing 3.17: This program converts the temperatures
// in Celsius to Fahrenheit.
#include <iostream>

int main()
{
    signed int cel;
    float fah;

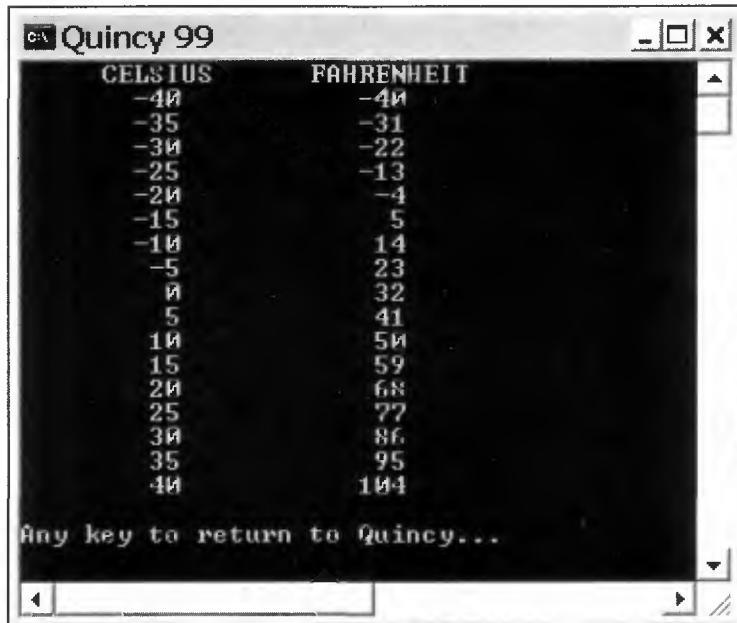
    cout << " CELSIUS      FAHRENHEIT\n";
    for (cel = -40; cel <= 40; cel += 10)
    {
        fah = 1.8*cel+32.0;
        cout.setf(ios::fixed); // set cout flag for fixed-point
        cout.width(10); // print width specified
        cout.precision(2); // specify two decimal places
        cout << cel << '\t';

        // width() manipulator does not stick from one
        // command to the next
        cout.width(10);
        cout << fah << endl;
    }
    return 0;
}

```

ද (၃။ ၃၃)

- for loop က <initialization> expression မှာ cel = -40 လိုအပေးပါတယ်။ fah ပေါ်မြဲလာ ကိုအသုံးပြုပြီး fah = $1.8 * \text{cel} + 32 = 1.8 * (-40) + 32 = -40$ လိုတွက်လိုက်ပါပြီ။ for loop အတွင်းမှာ cel ကို print စလုပ်ပါတယ်။ tab မြှားပါတယ်။ fah ကို print လုပ်ပါမယ်။
- အားလုံး print လုပ်ပြီးတာနဲ့ closing brace (}) နဲ့တွေ့ရင် cel ကို 10 တိုးပါတယ်။ cel = $-40 + 10 = -30$ ဖြစ်သွားပါပြီ။ cel = -30 ဟာ last counter ဖြစ်တဲ့ cel = 40 နဲ့ယူဉ်ရင် ငယ်လားညီလားလို့မေးပါတယ်။ ဒါ <test expr> ဟာ TRUE ဖြစ်တာမို့ for loop ကိုဆက် ပတ်ပါတယ်။ တစ်ချိန်မှာ cel = 50 ဖြစ်လာမှာပါ။ ဒါခါကျရင် cel = 50 ဟာ 40 ထက်ဘယ်ငယ် တော့မလဲ ။ ညီတော့မလဲ။ FALSE ဖြစ်သွားပြီ့ for loop ထဲကိုမဝင်တော့ပါဘူး။ program output ကို ပုံ (၃။ ၃၈) မှာဖော်ပြထားပါတယ်။



ও (১০ রে)

J# ৩ program কোড setw() manipulator অব্যঃপ্রিয়: আৰলি পুনৰেপ্তি:তোৱ run রেণ্ডলৰ্ন্স:ৱিপত্তি।

```
#include <iostream>
#include <iomanip>

int main( )
{
    signed int cel;
    float fah;

    cout << "    CELSIUS    FAHRENHEIT\n";
    for (cel = -40; cel <= 40; cel += 5)
    {
        fah = 1.8*cel + 32.0;
        cout << setw(10) << cel << setw(14) << fah << endl;
    }
    return 0;
}
```

Generating a Table of Multiples of a Number

- ၁။ အခုတင်ပြုမယ့် Ex3018.cpp program ဟာဆိုရင် user ကနေ input လုပ်ပေးတဲ့ ဂဏန်းတစ်ခုရဲ့ သွားကိန်းတွေကို (5) လုံးတစ်ခြောင်းပုံစံနဲ့ display လုပ်ပြီးတဲ့ program တစ်ခုဖြစ်ပါတယ်။ for loop ထဲတို့အလုပ်လုပ်သွားလဲဆိုတာကို ပုံ (၃။ ၃၉) က program ကိုလေ့လာကြည့်ပါ။

```
// Listing 3.18: This program generates a table of multiples
// of a number that you typed in. The program prints multiples
// every 5 numbers and starts next line.

#include <iostream>
#include <iomanip>

int main()
{
    unsigned long num;

    cout << "\nEnter a number : ";
    cin >> num;
    for (int i=1; i<=50; i++)
    {
        cout << setw(5) << i*num << " ";
        if (i%5 == 0) cout << endl;
    }
    return 0;
}
```

ပုံ (၃။ ၃၉)

- ၂။ Ex3018.cpp program ကို run မယ်ဆိုရင် ပုံ (၃။ ၄၀) မှာပြထားတဲ့အတိုင်းမြင်ရမှာပါ။ 12 အထက် 50 အထိဖော်ပြထားတဲ့ အလီယားတစ်ခုပဲပါ။ (5) လုံးစာတွက်ပြီးရင် next line ကူးတာကို if (i % 5 == 0) cout << endl; statement ကနေလုပ်ပေးပါတယ်။

	12	24	36	48	60
1	72	84	96	108	120
2	132	144	156	168	180
3	192	204	216	228	240
4	252	264	276	288	300
5	312	324	336	348	360
6	372	384	396	408	420
7	432	444	456	468	480
8	472	514	516	528	540
9	552	564	576	588	600

Any key to return to Quincy...

ပုံ (၃။ ၄၀)

၃.၈ Nested Loops

၁။ loop တစ်ခုကိုတစ်ခြား loop တစ်ခု သို့မဟုတ် တစ်ခုထက်ပိုတဲ့ loop တွေကင့်ထားတာကို nested loop လိုအပါတယ်။ inner looping နဲ့ outer looping တို့၏ control structure အဖိုးအစားတွေဟာ တူစရာ မလိုပါဘူး။ inner loop မှာ for နဲ့သုံးမယ်။ outer loop မှာ while သုံးမယ်။ ဒါလည်းရပါတယ်။ ဒါမှာမဟုတ် outer loop မှာ do-while သုံးပြီးတော့ inner loop မှာ for ကို သုံးမယ်ဆိုလည်းဖြစ်တာပါပဲ။ တစ်ခုပဲသတိထားရ မှာက loop တစ်ခုနဲ့တစ်ခုဟာ crossing ဖြတ်လို့မရပါဘူး။ အပြင် loop က အတွင်း loop ကို လုံးဝင်းနေရမှာပါ။

၂။ အခုတစ်ခါတင်ပြုမယ် Ex3019.cpp program ဘာ 2-equation (3-unknowns) ကို nested for loop အသုံးပြုပြီး ဖြေရှင်းထားတာပါ။ ပုံ (၃။ ၄၁) မှာ source code တွေကိုဖော်ပြထားပါတယ်။

၃။ Ex3019.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

■ program စေချင်း program အသုံးပြုမယ့် variable တွေအားလုံးကို integer လိုကြော်ပါတယ်။ outer for loop ပါတယ်။ $x = 1$ ပါ။ inner for loop ထဲကိုဆက်ဝင်ပါတယ်။ ဒါ ထဲမှာ $y = 1$ လိုပါတယ်။

```

// Listing 3.19: This program determines the three
// unknowns x,y, and z,which are defined by two equations:
//           x + y + z = 25
//           2.5x + 5y + 0.25z = 25
// where x, y, and z are integers.

#include <iostream>

int main()
{
    int x,y,z;

    for (x=1; x <= 10; ++x)
        for (y=1; y <= 5; ++y)
    {
        z = 25 - x - y;
        float sum = 2.5*x+ 5*y + 0.25*z;
        if (sum == 25) goto RESULT;
    }
    RESULT : cout << "X = " << x << endl;
    cout << "Y = " << y << endl;
    cout << "Z = " << z << endl;
    return 0;
}

```

ဗုဒ္ဓ (၃။ ၄၁)

ကွန်ပျိုတာက အခုလိုစပီးတွက်ယူပါတယ်။

$$z = 25 - x - y = 25 - 1 - 1 = 23$$

$$\text{float } \text{sum} = 2.5 * x + 5 * y + 0.25 * z$$

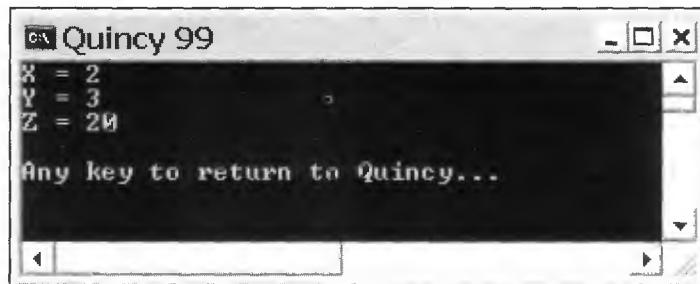
$$= 2.5 * 1 + 5 * 1 + 0.25 * 23 = 13.25$$

■ $\text{sum} = 25$ လားလို့မေးပါတယ်။ မှန်တယ်ဆိုရင် **RESULT label** ရှိတဲ့နေရာကို ကွန်ပျိုတာသွားမှာပါ။ $\text{sum} = 13.25$ ဟာ 25 နဲ့မတူဘူးဆိုတော့ အောက်ဆင်းလာပါတယ်။ closing brace နဲ့တွေ့ပြီး inner for loop ကို ဒုတိယအကြံမိထပ်ပတ်ပါပြီ။

■ ဒါတစ်ခါ $x = 1$ နဲ့ y ကို 2 လိုယူပြီးကွန်ပျိုတာကပြန်တွက် । ပြန်စစ်လုပ်ပါတယ်။ တစ်ခုနှင့်မှာတော့ တွက်လိုရတဲ့ sum တန်ဖိုးဟာ 25 ဖြစ်သွားပြီး nested loop ထဲကနေ ခုနှစ်ကိုလာပါပြီ။

RESULT label ရှိတဲ့နေရာမှာ $X = 2$ ၊ $Y = 3$ နဲ့ $Z = 20$ လို့ print လုပ်ပေးပါလိမ့်မယ်။

■ program output ကို ပုံ (၃။၄၂) မှာဖော်ပြထားပါတယ်။



ပုံ (၃။၄၂)

၅။ Ex3019.cpp program မှာအသုံးပြုတဲ့ nested for loop အစား while စု do-while statement တွေကိုအသုံးပြုပြီး အခုလိုပြင်ရေးလိုရပါတယ်။ ပုံ (၃။၄၃) မှာပြထားတဲ့ Ex3020.cpp program ကိုလေ့လာကြည့်ပါ။

```
#include <iostream>

int main()
{
    int x=1, y, z;

    while (1)
    {
        y = 1;
        do {
            z = 25 - x - y;
            float sum = 2.5*x+ 5*y + 0.25*z;
            if (sum == 25) goto RESULT;
            ++y;
        } while (y <=5);
        ++x;
    }
    RESULT : cout << "X = " << x << endl;
              cout << "Y = " << y << endl;
              cout << "Z = " << z << endl;
    return 0;
}
```

ပုံ (၃။၄၃)

၃.၉ The switch Statement

၁။ statement တွေ အများပြီးပါဝင်တဲ့ variable group အုပ်စုကနေ ကိုယ်လိုချင်တဲ့ statement group ဆိုတဲ့ ရောက်သွားစေချင်ရင် switch ဆိုတဲ့ statement ကိုသုံးရပါမယ်။ switch ခဲ့ syntax က ဒီလိုရှိပါတယ်။

```
switch ( <switch variable> )
{
    case <constant expr1> :
        <statement1>;
        [break;]

    case <constant expr2> :
        <statement2>;
        [break;]

    .....
    .....
    .....

    default :
        <statement_default>;
}
```

ဒီပုံစံမှာ switch နောက်မှာရှိတဲ့ <switch variable> ရဲ့ current value ဟာ <constant expr1> ရဲ့တန်ဖိုးနဲ့တူမယ်ဆိုလိုရှိရင် <statement1> ကို execute လုပ်ပြီး break တွေ့တာနဲ့ closing brace ()) အောက်ကို ဆင်းသွားမှာပါ။ break က optional ပါ။ လိုမှုထည့်ပေးရမှာပါ။ တစ်ကယ်လို့ <switch variable> ရဲ့ current value ဟာ <constant expr2> ရဲ့တန်ဖိုးနဲ့တူမယ်ဆိုရင် <statement2> ကို execute လုပ်ပြီး closing brace အောက်ကိုဆင်းသွားမှာပါပဲ။ <constant expr> တွေ တစ်ခုမှုမတူဘူးဆိုရင် default နောက်က <statement_default> ကို execute လုပ်မှာပါ။ ပြီးရင် switch block က ထွက်သွားမှာဖြစ်ပါတယ်။ Ex3021.cpp program ကိုအရင်လွှဲလာကြည့်ပါ။ ပုံ (၃.၄၄) မှာဖော်ပြထားပါတယ်။

```
// Listing 3.21: This program calculates the payroll for
// four different classes of workers.
#include <iostream>

int main()
{
    int category;
    float wage, hours, pay;

    START : cout << "Enter category (any of 1 to 4)\n";
    cin >> category;
    switch (category)
    {
        case 1:
            wage = 12.5; break;
        case 2:
            wage = 15.5; break;
        case 3:
            wage = 20.5; break;
        case 4:
            wage = 25.5; break;
        default: { cout << "Try again!\n";
                    goto START;      }
    }
    cout << "Enter hours worked\n";
    cin >> hours;
    if (hours <= 40)  pay = wage*hours;
    else             pay = wage*40 + 1.5*wage*(hours-40);
    cout << "Pay = " << pay << endl;
    return 0;
}
```

ပုံ (၃။၄၃)

JII Ex3021.cpp ကို run လိုက်မယ်ဆိုရင် ပုံ (၃။၄၃) မှာပြထားတဲ့အတိုင်းမြင်ရမှာပါ။ category အတွက် 3 နဲ့ working hours အတွက် 65 တို့ကိုရိုက်ထည့်ပေးမယ်ဆိုရင် Pay = 1588.75 လိုတွက်ပေးမှာပါ။ ဒါပေမယ့် category < 1 သို့မဟုတ် category > 4 ဖြစ်နေရင် data ပြန်တောင်းပါလိမ့်မယ်။

```
Quincy 99
Enter category (any of 1 to 4)
3
Enter hours worked
65
Pay = 1588.75
Any key to return to Quincy...
```

ပုံ (၃။၄၅)

Use a switch Statement to Control a Menu

- ၁။ အခြားတင်ပြမယ့် Ex3022.cpp program ဘာ switch statement ကိုအသုံးပြုပြီး menu တစ်ခုကို create လုပ်နည်းဖြစ်ပါတယ်။ ဒါ program ကို run လိုက်မယ်ဆိုရင် ပုံ (၃။၄၅) မှာပြထားတဲ့အတိုင်း menu list တစ်ခုကိုတွေ့ရမှာပါ။ choice (4) မျိုးထက် နံပါတ် (1) ကို choose လုပ်ထားပါတယ်။

```
Quincy 99
PC EXPRESS Computer Repair Center
What work performed? Here's the choices :
1. Replaced the CPU and RAM
2. Replaced the RAM only
3. Repaired the monitor
4. Fixed stuck keys

What work was performed? 1
Total charge is $350.00

Any key to return to Quincy...
```

ပုံ (၃။၄၆)

Ex3022.cpp

```

// Listing 3.22: This program uses a switch statement to
// control a user's selection.
#include <iostream>

int main()
{
    int choice;
    float charge=0;

    cout << "\tPC EXPRESS Computer Repair Center\n\n";
    cout << "What work performed? Here's the choices :\n";
    cout << "\t1. Replaced the CPU and RAM\n";
    cout << "\t2. Replaced the RAM only\n";
    cout << "\t3. Repaired the monitor\n";
    cout << "\t4. Fixed stuck keys\n";
    do {
        cout << "\nWhat work was performed? ";
        cin >> choice;
    } while (choice < 1 || choice > 4);

    switch (choice)
    {
        case 1 : charge = 200; // Notice no break here
        case 2 : charge += 150; break;
        case 3 : charge = 75; break;
        case 4 : charge = 12;
    }
    cout.precision(2);
    cout.setf(ios::showpoint);
    cout.setf(ios::fixed);
    cout << "\nTotal charge is $" << charge << endl;
    return 0;
}

```

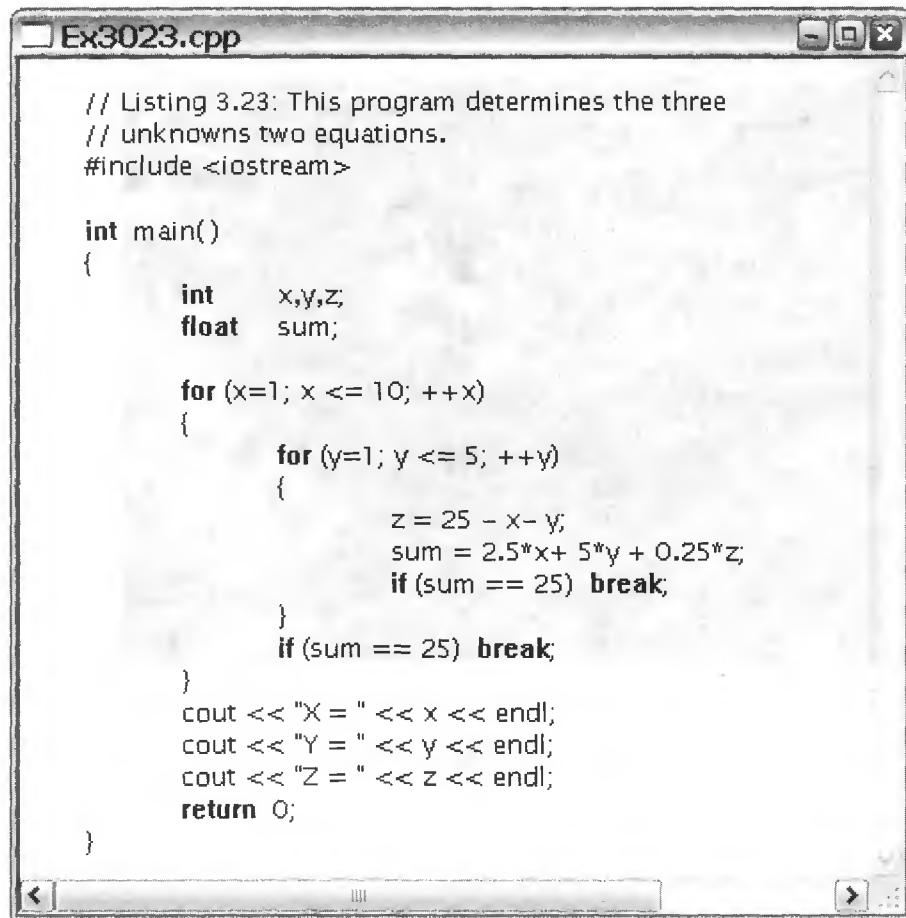
ဗ (၃။၄၆)

၂။ ဗ (၃။၄၅) ၏ program ၏ case 1: statement နောက်ဆုံးမှာ break ကို တမ်းပြတ်ထားတာကို
သတိမြှောင်းလွှာပါ။ switch (choice) ၏ choice=1 ဖြစ်မယ်ဆိုရင် charge=200 လိုက္ခန်ပူ။
တာက နားလည်ပါတယ်။ ဒါပေမယ့် case 1 statement ၏ break ပေါ်တဲ့အတွက် switch loop ၏ အပြင်ကိုခုန်မထွက်သေးပါဘူး။

အာက်ကိုဆက်ဆင်လာပြီး charge += 150 သို့မဟုတ် charge = 200+150 = 350 လိုတွက်ပါလိမ့်မယ်။ break ကိုတွေ့တော့မှ switch loop အပြင်ကိုခုန်ထွက်မှာပါ။ choice=2 ကို data input လုပ်ရင်တော့ အဖြေ charge = 150 ပဲရမှာပါ။

၃.၁၀ The break Statement

break ဆိုတာ looping သို့မဟုတ် switch အုပ်စုကနေ exit လုပ်ဖို့အတွက်အသုံးပြုတာပါ။ while , do-while , for , switch statement တွေနဲ့ break ကိုတွဲသုံးလိုပါတယ်။ ဒါပေမယ့် if-statement တနောက် exit လုပ်တာကိုတော့ break နဲ့သုံးလိုပါဘူး။ goto ပဲသုံးလိုပါမှာပါ။ Ex3023.cpp ဟာ Ex3020.cpp program



```
// Listing 3.23: This program determines the three
// unknowns two equations.
#include <iostream>

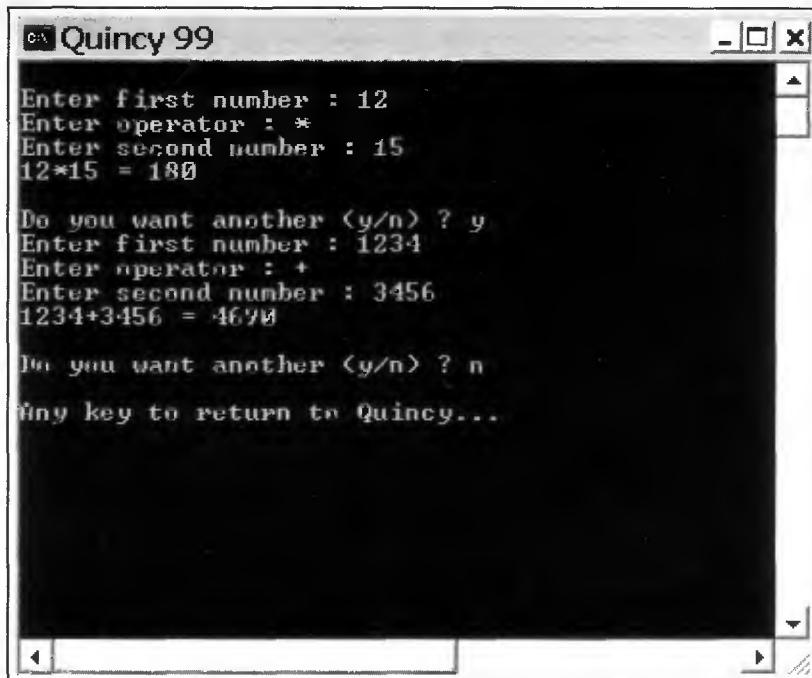
int main()
{
    int     x,y,z;
    float   sum;

    for (x=1; x <= 10; ++x)
    {
        for (y=1; y <= 5; ++y)
        {
            z = 25 - x - y;
            sum = 2.5*x+ 5*y + 0.25*z;
            if (sum == 25) break;
        }
        if (sum == 25) break;
    }
    cout << "X = " << x << endl;
    cout << "Y = " << y << endl;
    cout << "Z = " << z << endl;
    return 0;
}
```

ကို break statement အသုံးပြုပြီး နောက်တစ်မျိုးပြင်ရေးထားတာပါပဲ။ break statement ကိုအသုံးပြုတဲ့အခါ့သတိထားရမယ့်အချက်က break တစ်ခုဟာ loop တစ်ခုစာပဲခုနှင့်ထွက်လို့ရပါတယ်။ nested loop တွေက exitလုပ်တဲ့အခါ့မှာ break တစ်ခုမက အသုံးပြုရပါမယ်။ goto statement ကတော့ ဖြိုက်သလို ခုနှင့်ကျော်လို့ရပါတယ်။

The Equivalent of a Four-function Calculator

အခုံတစ်ခါတင်ပြုမယ့် break statement ကို အပေါင်းအနှံတဲ့၊ အမြှောက်အစားလုပ်တဲ့ calculator program မှာ ပုံ (၃။၄၈) မှာပြထားတဲ့အတိုင်းအသုံးပြုထားပါတယ်။ ဒီအတွက် program code တွေကို ပုံ (၃။၅၉) က Ex3024.cpp program မှာရေးပြထားပါတယ်၊ လေ့လာကြည့်ပါ။



The screenshot shows a terminal window titled "Quincy 99". The program is a simple four-function calculator. It prompts the user to enter two numbers and an operator, then performs the calculation and prints the result. The user can choose to perform another calculation by entering 'y' or quit by entering 'n'. The window has standard window controls (minimize, maximize, close) and scroll bars.

```
Quincy 99
Enter first number : 12
Enter operator : *
Enter second number : 15
12*15 = 180

Do you want another (y/n) ? y
Enter first number : 1234
Enter operator : +
Enter second number : 3456
1234+3456 = 4670

Do you want another (y/n) ? n
Any key to return to Quincy...
```

ပုံ (၃။၄၈)

```
// Listing 3.24: This program creates the equivalent of
// a four-function calculator. The program would request
// the user to enter a number, an operator, and another number.

#include <iostream>
#include <conio.h>

int main()
{
    double num1, num2, ans;
    char opr, ch;

    do {
        cout << "\nEnter first number : "; cin >> num1;
        cout << "Enter operator : "; cin >> opr;
        cout << "Enter second number : "; cin >> num2;
        switch (opr) {
            case '+': ans = num1+num2; break;
            case '-': ans = num1-num2; break;
            case '*': ans = num1*num2; break;
            case '/': ans = num1/num2; break;
            default: ans = 0;
        }
        cout << num1 << opr << num2 << " = " << ans << endl;
        cout << "\nDo you want another (y/n) ? ";
        ch = getch();
    } while (ch != 'n');

    cout << endl;
    return 0;
}
```

ও (১০.১৬)

২.১১ The continue Statement

- oii loop কর্তৃপক্ষের মধ্যে loop কর্তৃক statement টেক্সট অন্তর্ভুক্ত execute করা হবে।

ကနေ နောက်တွောင်းပြန်ပြီးတော့ loop ကို နောက်တစ်ကြိမ်ပြန်ပတ်ချင်ရင် continue ဆိုတဲ့ statement ကိုသုံးရပါတယ်။ continue ကို while ၊ do-while ၊ for တိုနဲ့တွဲပြီးသုံးနိုင်ပါတယ်။

```
// Listing 3.25: This program finds the average of the
// nonnegative integers from a list of numbers.
#include <iostream>

int main()
{
    int      n, navg=0;
    float    x, avg, sum=0;

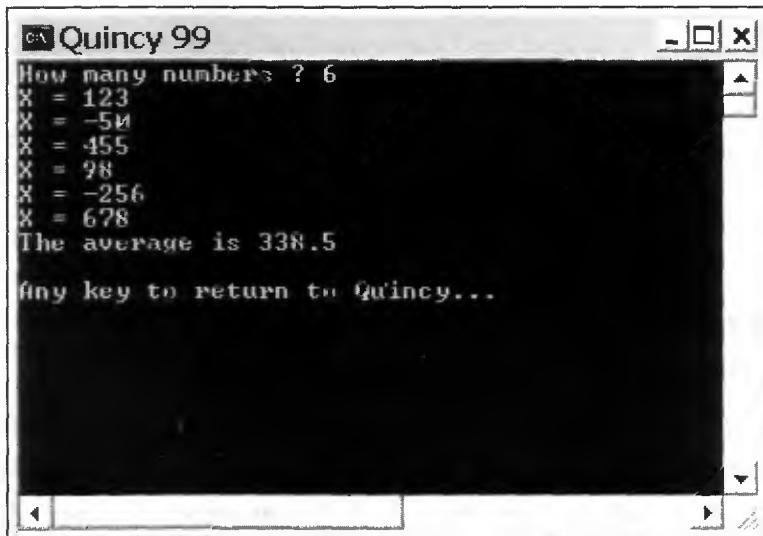
    cout << "How many numbers ? ";
    cin >> n;
    for (int count=1; count<= n; ++count)
    {
        cout << "X = ";
        cin >> x;
        if (x < 0) continue;
        sum += x;
        ++navg;
    }
    avg = sum / navg;
    cout << "The average is " << avg << endl;
    return 0;
}
```

ပုံ (၃၁၅၀)

၂။ Ex3025.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

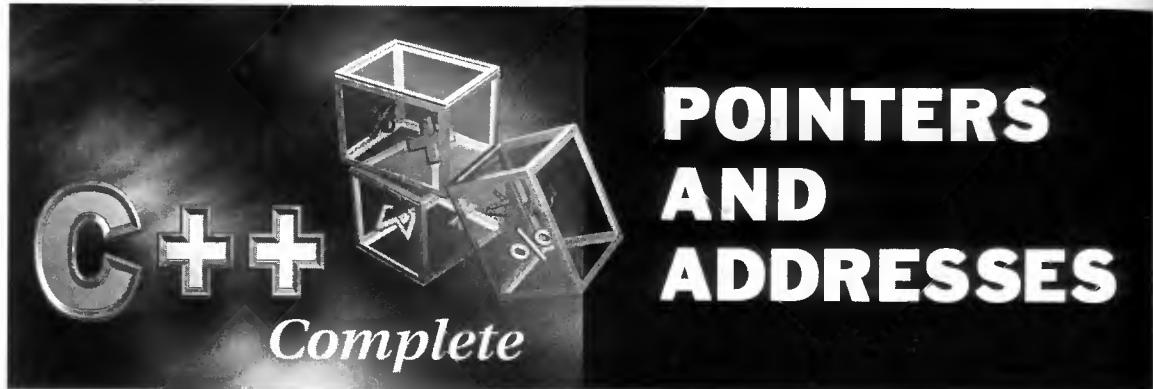
- ဒါ program မှာ ကျွန်တော်တိုထည့်လိုက်တဲ့ data တွေထဲက +ive number တွေကိုပဲပေါင်းပြီးတော့ average ကိုရှာပေးမှာပါ။ စစ်ချင်းမှာ variable တွေကို type declare လော်တယ်။ တစ်ချို့ကို initialize လည်းထပ်လုပ်ပါတယ်။ data တောင်းတဲ့ prompt ပေါ်ခိုင်တယ်။ data ထည့်တယ်။

- for loop အတွင်းမှာအလုပ်လုပ်ပါတယ်။ x က သုညထက်ငယ်တယ်ဆိုရင် (negative number ဆိုရင်) သူအောက်က statement (2) ခုကိုခုန်ကျဉ်ပြီး for loop ကို နောက်တစ်ကြိမ်ပတ်ပါ ဖိမ့်မယ်။ x ဟာ positive ဆိုရင် sum ထဲမှာထည့်ပေါင်းမှာပါ။ ပြီးရင် counter ကိုတစ်ခုတိုးပါလိမ့်မယ်။
 - ဒီနည်းအတိုင်း program ကိုတွက်သွားမယ်ဆိုရင် sum ထဲမှာ positive number တွေအားလုံး ရှုပေါင်းလှစ်တွေကိုရပါပြီ။ နောက်ပြီး navg ထဲမှာလည်း positive number အရေအတွက်ကို သိနေပြီးလေ။ ဒီတော့ avg = sum / navg ဟာအဖြေပေါ့ပေါ့။ program ကိုစာဖတ်သူကိုယ်တိုင် trace လုပ်ကြည့်ပါ၊ မခက်ပါဘူး။ break နဲ့ continue တို့အကြောင်းတွေကိုလည်း ပိုနားလည် သွားတာပေါ့။
- ၃။ Ex3025.cpp program ကို run လိုက်မယ်ဆိုရင် ပုံ (၃။ ၅၁) မှာပြထားတဲ့အတိုင်းမြင်ရမှာပါ။ ငက်န်း (6) လုံးကိုထည့်ပေးလိုက်ရင် positive number တွေကိုပဲကွက်ပေါင်းပြီး average ရှာပေးပါလိမ့်မယ်။



ပုံ (၃။ ၅၁)

Chapter 4



pointer ဆိတ် variable တစ်ခု သို့မဟုတ် function တစ်ခုရဲ့ address တည်နေရာကိုသိမ်းဆည်းထားတဲ့ variable တစ်ခုပါပဲ။ ဥပမာ x ဆိတ် integer variable တစ်ခုရဲ့ value ဟာ 100 ဖြစ်ပြီး address က &x ဖြစ်မယ်။ နောက်ပြီး x ရဲ့ address value ကို store လုပ်ထားတဲ့ variable က xp ဖြစ်မယ် ဆိုရင်အဲဒါ xp ကို pointer လိုခံဗျာပါတယ်။ x နဲ့ xp တို့ရဲ့ ဆက်သွယ်မှုကိုဖော်ပြချင်ရင် int* xp = &x; လိုအပ်ပါ။ & (ampersand) က address operator ပါ။ * ကို indirection operator လိုခံဗျာပါတယ်။ int* xp ရဲ့အမိုးပါယ်က variable xp ကို pointer လို့ define လုပ်ပြီး x ရဲ့ address value ကို xp နဲ့ assign လုပ် ပေးတဲ့သဘောပါပဲ။ အခုလိုဆက်သွယ်မှုဖြစ်နေရင် x ပဲ print လုပ်လုပ်၊ *xp ကိုပဲ print လုပ်လုပ် အငြေဖြတစ်မျိုး တည်းရမှာပါ။ အဲဒါ 10 ပေါ့။ စင်စစ် pointer ရဲ့အလုပ်ဟာ data item တစ်ခုရဲ့ တန်ဖိုးကို သာမန် variable name တစ်ခုနဲ့ ဖော်ပြနည်းမျိုးမဟုတ်ပဲ အဲဒါ data ရှိတဲ့ address နေရာကိုသွယ်ပိုက်ဖော်ပြပြီး သိခြင်းပါ။ pointer တွေနဲ့ပတ်သက်တဲ့ idea ဟာ ရှုပ်တွေးတယ်လို့မဆိုနိုင်ပါဘူး။ program တစ်ခုကိုကွန်ပျူးတာ memory ထဲထည့်သွင်းပေးလိုက်မယ်ဆိုလိုရှိရင် အဲဒါအတွက် address အုပ်စု တစ်ခုစာတော့နေရာယူမှာပါ။ နောက်ပြီး program

မှတ်ပါဝင်တဲ့ variable တွေ၏ function တွေက သူတို့အတွက်သတ်မှတ်ထားတဲ့ address နေရာအသီးသီးကနေ ပြုကြပါလိမ့်မယ်။

၄.၁ Referencing Variables with Pointers

- ဗုဒ္ဓ (၄.၁) မှတ်ပြုထားတဲ့ Ex401.cpp program ဘာဆိုရင် char | int | float type variable (3) ခုထဲက data value တွေကို pointer အသုံးပြုပြီး print လုပ်နည်းကိုဖော်ပြုထားပါတယ်၊ လေ့လာကြည့်ပါ။

```
// Listing 4.1: Referencing variables with pointers
#include <iostream>

int main()
{
    // initialize variables
    char c = 'A';
    int i = 123;
    float f = 567.89;

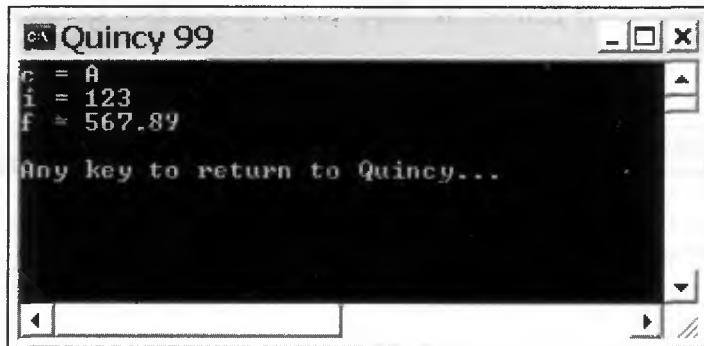
    // define pointer types and initialize with address
    char* cp = &c;
    int* ip = &i;
    float* fp = &f;

    // print variable values using pointers
    cout << "c = " << *cp << endl;
    cout << "i = " << *ip << endl;
    cout << "f = " << *fp << endl;
    return 0;
}
```

ဗုဒ္ဓ (၄.၁)

Ex401.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- စစချင်းမှာ `c`၊ `i` နဲ့ `f` ဆိုတဲ့ variable (3) ခုအတွက် data type တွေကိုကြောဖြီး သတ်မှတ်ထားတဲ့တန်ဖိုးအသီးသီးနဲ့ initialize လုပ်ပေးပါတယ်။ နောက်ပြီး pointer (3) ခုဖြစ်တဲ့ `cp`၊ `ip` နဲ့ `fp` တိုကို define လုပ်ပြီး `c`၊ `i` နဲ့ `f` variable တွေရဲ့ address value တွေနဲ့ initialize လုပ်ပေးမှာပါ။ variable `c` ထဲက 'A' ကိုလိုချင်ရင် `*cp` ကို display လုပ်ခိုင်းရပါတယ်။
- ဒီနည်းအတိုင်း `i` နဲ့ `f` တို့ရဲ့ value တွေကိုလည်း `display` လုပ်လို့ရပါတယ်။ ပုံ (၄၁၂) ဟာ Ex401.cpp program ကို run လိုက်လိုပေါ်လာတဲ့ output result ဖြစ်ပါတယ်။



ပုံ (၄၁၂)

Using the Pointer in a Loop

- ၁။ အခုတင်ပြမယ့် Ex402.cpp program ဟာဆိုရင် for loop ထဲမှာ pointer တစ်ခုကိုအသုံးပြုပြီး 1 ကနေ 1 အထိ ဂွန်ပျူတာမှာပေါ်ခိုင်းပါလိမ့်မယ်။ ပြီးရင် BOOM! ဆိုတဲ့စာလည်း ဆက်ပေါ်လာမှာပါ။ ပုံ (၄၁၃) မှာရေးထားတဲ့ program က code statement တွေကိုလေ့လာကြည့်ပါ။

J|| Ex402.cpp program ကို trace လုပ်ကြည့်မယ်ဆိုရင်

- program စစချင်း `countDown` ဆိုတဲ့ variable ကို type int ထိုကြောပါတယ်။ ပြီးထော် pointer `cdp` ကို define လုပ်ပြီး `countDown` ရဲ့ address value နဲ့ assign လုပ်၏

```

// Listing 4.2: Using a pointer, this program
// counts from 10 to 1 displaying the numbers
// followed by BOOM!.
#include <iostream>

int main()
{
    int countDown;
    int* cdp = &countDown;

    for (countDown=10; countDown >0; countDown--)
        cout << *cdp << ", ";
    cout << "BOOM!" << endl;
    return 0;
}

```

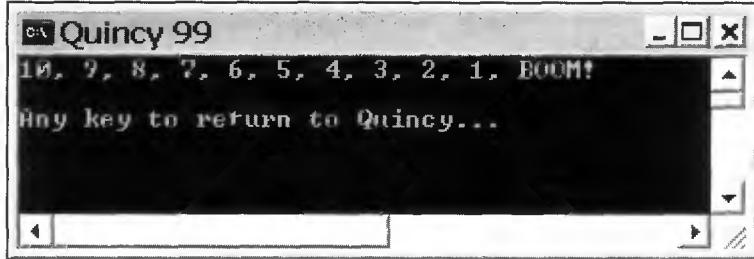
ဗု (၄။ ၃)

ပါတယ်။ for loop ထဲမှာ countDown ကို 10 ကနေ 1 အထိ decrement လုပ်သွားရင် *cdp ကို display လုပ်ခိုင်းပါတယ်။

ပထမဆုံး: cdp point လုပ်နေတဲ့ countDown က 10 ပါ။ ဒီတော့ *cdp ကို display လုပ်ရင် ကွန်ပူးတာမှာ 10 ပေပါ၏လာမှာပေါ့။ ပြီးရင် comma (,) တစ်ခုဆက်ပေါ်ပါလိမ့်မယ်။ for loop က အပေါ်ပြန်တက်သွားတဲ့အခါမှာ countDown က 9 ဖြစ်သွားပါပြီ။ ဒီတစ်ခါ cdp point လုပ်နေတဲ့ countDown က 9 ဖြစ်တဲ့အတွက် *cdp = 9 ဖြစ်ပါတယ်။ *cdp ကို display လုပ်ခိုင်းရင် 9 ပေါ်လာမှာပါ။

ဒီနည်းအတိုင်း: for loop ထဲမှာပတ်နေရင်: 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, တို့ကို display လုပ်ပြပါလိမ့်မယ်။ for loop ထဲကထွက်လာတဲ့အခါမှာ cout << "BOOM!" << endl; statement နဲတွေရင် 1 နောက်မှာကပ်ပြီး BOOM! ကို display ဆက်လုပ်ပြမှာပါပဲ။

၃။ Ex402.cpp program ကို ဗု (၄။ ၄) မှာ run ပြထားပါတယ်။ output result တွေကို စောင့်ပြုတော်းတွေများတွေမှာ ပြန်လည့်ရပါတယ်။



ပုံ (၄။ ၅)

၄.၂ Pointers and Arrays

၁။ နိုင်း variable တွေအတွက် address သတ်မှတ်တဲ့အခါး & (ampersand) ထည့်ပေးဖိုလိုပေမယ့် array တွေကျတော့ array name ကိုယ်တိုင်က address ဖြစ်နေတော့ & ထည့်စိုမလိုပါဘူး။ array notation အသုံးပြုပို့တွေကို ပုံ (၄. ၅) က Ex403.cpp program မှာနှိပ်တော်ပြထားပါတယ်။ လေ့လာကြည့်ပါ။

၂။ Ex403.cpp program ကို trace လုပ်ကြည့်မယ်ဆိုရင်

- စစချင်းမှာ type int ဖြစ်တဲ့ array တစ်ခုကို define လုပ်သလို initialize လည်းလုပ်ပါတယ်။ array name က x ပါ။ x ထဲမှာပါဝင်တဲ့ array element တစ်ခုချင်းကို array notation အသုံးပြုပြီး အရင်ဆုံး display လုပ်ခိုင်းပါတယ်။ $x[0] = 10$, $x[1] = 20$, $x[2] = 30$, $x[3] = 40$ နဲ့ $x[4] = 50$ တို့ဖြစ်ကြပါတယ်။
- ဒုတိယအုပ်စုမှာလုပ်တာက array element တွေကို pointer notation အသုံးပြုပြီးတော့ display လုပ်ခိုင်းတာပါပဲ။ $*(x+i)$ ဟာ $x[i]$ နဲ့အတူတူပါပဲ။ $x+2$ ဟာဆိုရင် x array ခဲ့တတိယမြောက် integer ရဲ့ address ကိုဆိုလိုပါတယ်။ ဒါတော့ $*(x+2)$ ဟာ 30 ဆိုတဲ့တန်ဖိုးနဲ့တူမှာပါ။
- address တွေကိုညွှန်ပြတဲ့အခါး x မှာ i ထည့်ပေါင်းသလိုပဲ pointer မှာ increment လုပ်ပေးပြီး ညွှန်လို့ရပါတယ်။ xp ဟာ x ကို point လုပ်နေတဲ့ pointer ဖြစ်တယ်ဆိုရင် $*(xp++)$ ကို loop ထဲမှာ display လုပ်ခိုင်းလိုက်ရင် array element တစ်ခုချင်းပေါ်လာမှာပါပဲ။

၃။ Ex403.cpp ကို ပုံ (၄. ၆) မှာ run ပြထားပါတယ်။ program နဲ့တိုက်ပြီး လေ့လာကြည့်လို့ရပါတယ်။

Ex403.cpp

```
// Listing 4.3: This program shows the different
// ways how the arrays are accessed.

#include <iostream>
int x[5] = {10,20,30,40,50};

int main()
{
    // array accessed with array notation
    for (int i=0; i<5; i++)
        cout << " " << x[i];
    cout << "\n\n";

    // array accessed with pointer notation
    for (int i=0; i<5; i++)
        cout << " " << *(x+i);
    cout << "\n\n";

    // array accessed with pointer
    int* xp = x;
    for (int i=0; i<5; i++)
        cout << " " << *(xp++);
    cout << "\n\n";
    return 0;
}
```

◊ (9. 9)

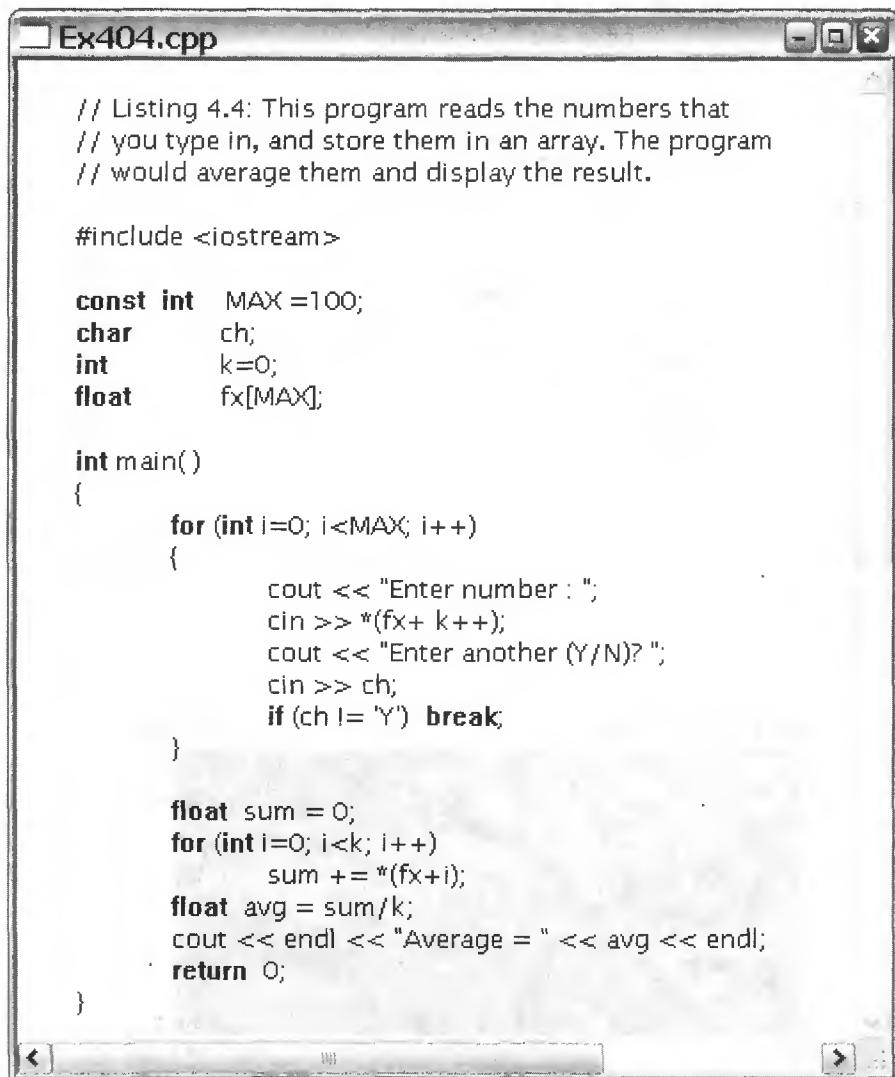
Quincy 99

```
10 20 30 40 50
10 20 30 40 50
10 20 30 40 50
Any key to return to Quincy...
```

◊ (9. 6)

Pointer Arithmetic

၁။ အခုတင်ပြမယ့် Ex404.cpp program ဟာဆိုရင် pointer notation အသုံးပြု၍ array တစ်ခုမှာ store လုပ်ထားတဲ့ value တွေကို average ရှာပေးတဲ့ program ပါ။ ပုံ (၄။ ၃) မှာရေးထားတဲ့ program က code statement တွေကိုလေ့လာကြည့်ပါ။



```
// Listing 4.4: This program reads the numbers that
// you type in, and store them in an array. The program
// would average them and display the result.

#include <iostream>

const int MAX =100;
char ch;
int k=0;
float fx[MAX];

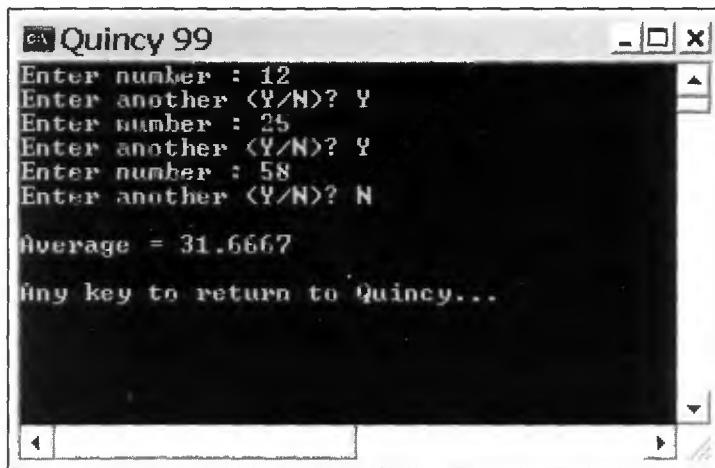
int main()
{
    for (int i=0; i<MAX; i++)
    {
        cout << "Enter number : ";
        cin >> *(fx+ k++);
        cout << "Enter another (Y/N)? ";
        cin >> ch;
        if (ch != 'Y') break;
    }

    float sum = 0;
    for (int i=0; i<k; i++)
        sum += *(fx+i);
    float avg = sum/k;
    cout << endl << "Average = " << avg << endl;
    return 0;
}
```

ပုံ (၄။ ၃)

Ex404.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- for loop နဲ့ pointer notation တို့ကိုအသုံးပြုပြီး x array ခဲ့ element တစ်ခုချင်းအတွက် value တွေကို ရှုက်ထည့်ပေးသွားပါတယ်။ `cin >> *(x + k++);` statement ကိုသတိပြုကြည့်ပါ။ k က ကျွန်ုန်တော်တို့ input လုပ်မယ့် array element အရေအတွက်ကိုမှတ်ပေးမယ့် variable တစ်ခုဖြစ်ပါတယ်။ data ကိုဆက်ထည့်ပေးခွင်တယ်ဆိုရင် 'Y' ကိုရှိရှိထည့်ပါ။ data ထပ်မထည့်ချင်ဘူးဆိုရင် 'N' ကိုရှိရှိထည့်ပါ။ ဒါဆိုရင် `break;`နဲ့တွေ့ပြီး for loop အပြင်ဘက် ကိုခုန်ထွက်လာပါလိမ့်မယ်။
- အခုခုရင် count ထဲမှာ တစ်ကယ်အသုံးပြုတဲ့ array element အရေအတွက်ပါလာပါပြီ။ array element စုစုပေါင်းရှုတန်ဖိုးကို ပေါင်းခိုင်းပြီးတော့ sum ထဲမှာ store လုပ်ပေးပါတယ်။ နောက်ပြီး `avg = sum/k` နဲ့ထွက်ယူပြီး အပြောက် `display` လုပ်ပြလိုက်ပါပြီ။ ပုံ (၄၁) မှာ Ex405.cpp program ကို run လိုက်လို့ရလာတဲ့ result ကိုဖော်ပြထားပါတယ်။



ပုံ (၄၁)

- တစ်ကယ်လို့ pointer ကို တိုက်ရှိက်အသုံးပြုပြီး Ex404.cpp program ကိုပြင်ရေးမယ်ဆိုရင် ပုံ (၄၂) မှာပြထားတဲ့ program မျိုးကိုတွေ့ရမှာပါ။ ဒီ program ကို run ရင်လည်း စောင့်က အဖြော်ရမှာပါ။

```
#include <iostream>

const int MAX = 100;
char ch;
int k=0;
float fx[MAX];

int main()
{
    float* fpx = fx;

    for (int i=0; i<MAX; i++)
    {
        cout << "Enter number : ";
        cin >> *(fx+ k++);
        cout << "Enter another (Y/N)? ";
        cin >> ch;
        if (ch != 'Y') break;
    }

    float sum = 0;
    for (int i=0; i<k; i++)
        sum += *(fpx+i);
    float avg = sum/k;
    cout << endl << "Average = " << avg << endl;
    return 0;
}
```

ပုံ (၄။ ၆)

၄.၃ Pointers to Structures

၁။ structure တွေနဲ့ဖြော်ဆုံး အလုပ်လုပ်မယ့် pointer တွေဟာ တစ်ခြား pointer တွေအလုပ်လုပ်သလိုပါ။ structure pointer တစ်ခုဟာ structure type ရဲ့ instance ကို point လုပ်နေပါတယ်။ ဒီတော့ pointer ကို increment သို့မဟုတ် decrement လုပ်တာနဲ့ structure size ဟာ ဆွေးပြီးတိုးသွားနိုင်ပါတယ်။ လေ့နိုင်ပါတယ်။ structure တစ်ခုက မေးပါတယ်။ structure တစ်ခုက မေးပါတယ်။ member တစ်ခုကို pointer နဲ့ access လုပ်ချင်ရင် member pointer

operator (->) ကိုအသုံးပြုရမှာပါ။ ပုံ (၄. ၁၀) မှာဖော်ပြထားတဲ့ Ex405.cpp program ဟာဆိုရင် point-to structures ဆိုတဲ့အဓိပ္ပာယ်ကို ပေါ်လွင်အောင်ရေးပြထားတဲ့ program တစ်ခုပါပဲ။

```
// Listing 4.5: Pointers to structures
#include <iostream>

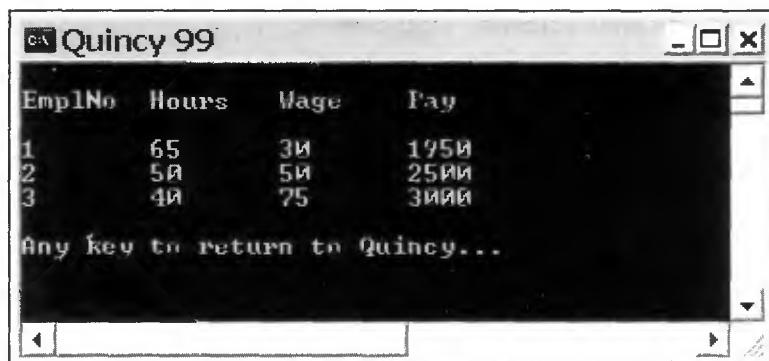
struct employeeRec
{
    int      emplNo;
    float    hours;
    float    wage;
};

void displayRec(const employeeRec *y)
{
    cout << "\nEmplNo Hours Wage Pay\n\n";
    while (y->emplNo != -1)
    {
        cout << y->emplNo << "\t"
            << y->hours << "\t"
            << y->wage << "\t"
            << (y->hours)*(y->wage) << endl;
        y++;
    }
}

int main()
{
    employeeRec x[] = {
        {1, 65, 30},
        {2, 50, 50},
        {3, 40, 75},
        {-1, 0, 0}
    };
    displayRec(x);
    return 0;
}
```

၁၂ (၁၀. ၁၀)

- စစချင်းမှာ employeeRec ဆိတဲ့ structure တစ်ခုကို declare လုပ်ထားပါတယ်။ main() function ထဲမှာ employee instance တစ်ခုဖြစ်တဲ့ x[] array ကို declare လုပ်ခြင်းအားဖူးသူထဲမှာ employeeRec structure variable တွေကို initialize လုပ်ထားလိုပါတယ်။ အောင် displayRec() function ကို call ခေါ်ပြီး x[0] element နဲ့ address ကို pass လုပ်ပေးလိုက်ပါပြီ။ displayRec() function ထဲမှာ ပထမဆုံး array element အတွက် တွက်တော်ချက်တာတွေလုပ်ပါလိမ့်မယ်။ တွက်ချက်ပြီးရင် pointer y ကို increment လုပ်ခြင်းအားဖြင့် x[1] element ၊ x[2] အတွက်ဆက်တွက်မှာပါ။ y->emplNo = -1 ဖြစ်သွားတဲ့အခါမှာ displayRec() ကနေဖြန့်ထွက်လာပြီး program ပြီးသွားပါပြီ။ ပုံ (၄. ၁၁) မှာ Ex405.cpp program ကို run ပြထားပါတယ်။



ပုံ (၄. ၁၁)

၄.၄ Pointers as Function Arguments

- program တစ်ခုမှာ argument တွေကို function တစ်ခုကနေတစ်ခုဆီကို pass လုပ်နည်း (၃) နည်းရှိပါတယ်။ (၁) passing argument by value (၂) passing arguments by reference နဲ့ (၃) passing argument by pointer တို့ဖြစ်ကြပါတယ်။ အဲဒါထဲက ဒုတိယနည်းနဲ့ တတိယနည်းကိုတင်ပြပါမယ်။ (၄. ၁၀) က Ex406.cpp မှာဖော်ပြထားတဲ့ passing by reference နည်းကို အရင်လေ့လာကြည့်ရအောင်။

```

// Listing 4.6: This program shows how arguments
// are passed by reference
#include <iostream>

void convert(double&);

int main()
{
    double inches;

    cout << "Enter inches : ";
    cin >> inches;
    convert(inches);
    cout << "Length = " << inches << " centimeters\n";
    return 0;
}

void convert(double &x)
{
    // convert inches to centimeters
    x *= 2.54;
}

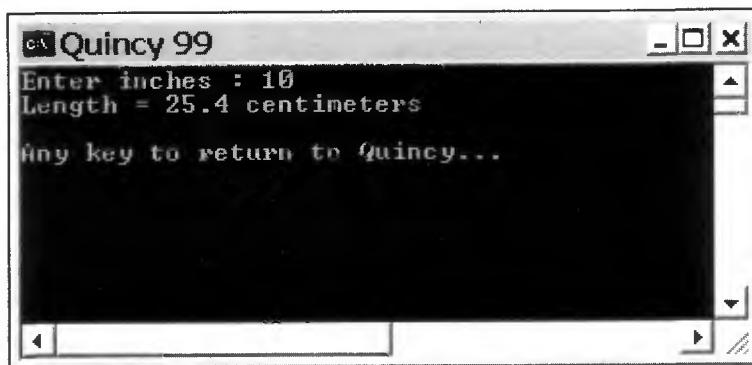
```

ပုံ (၄၀၂)

၂။ Ex406.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- main() function ထဲက inches ဆိုတဲ့ variable တစ်ခုကို လက်မကနေစင်တီမီတာကိုပြောင်းလေးတာပါ။ inches ကို centimeters ပြောင်းဖို့အတွက် convert() function ထဲကို by reference နည်းနှင့် data pass လုပ်ပေးထားပါတယ်။ by reference နည်းအရ main() ထဲက convert() function မှာရှိရှိး variable တစ်ခုကို argument အနေနဲ့ pass လုပ်ထားပေမယ့် prototype မှာ & ကို data type နောက်မှာ ထည့်ရေးထားတာကို သတိပြုကြည့်ပါ။ ဒါမှ data pass ပြန်လုပ်ပေးမှာပါ။ & ကိုဖြုတ်ထားမယ်ဆိုရင် program run ပေမယ့် အမြော်များမှာမဟုတ်ပါဘူး။ တွက်ထားတဲ့ x တန်ဖိုးကို return မပေါ်လိုပါပဲ။

- Ex406.cpp program ကို run လိုက်ပြီး inches = 10 ကိုရှုက်ထည့်ပေးရင် Length = 25.4 centimeters ဆုတေသနဖြစ်ကို ပုံ (၄. ၁၃) မှာပြထားတဲ့အတိုင်း မြင်ရပါလိမ့်မယ်။



ပုံ (၄. ၁၃)

- Ex406.cpp program ကို passing by pointer နည်းနှင့်ပြင်ရေးမယ်ဆိုရင် ပုံ (၄. ၁၄) မှာပြထားတဲ့ program မျိုးကိုတွေ့ရမှာပါ။ ဒါ program ကို run ရင်လည်း စောင့်ခဲ့ပေါ်ရမှာပါ။

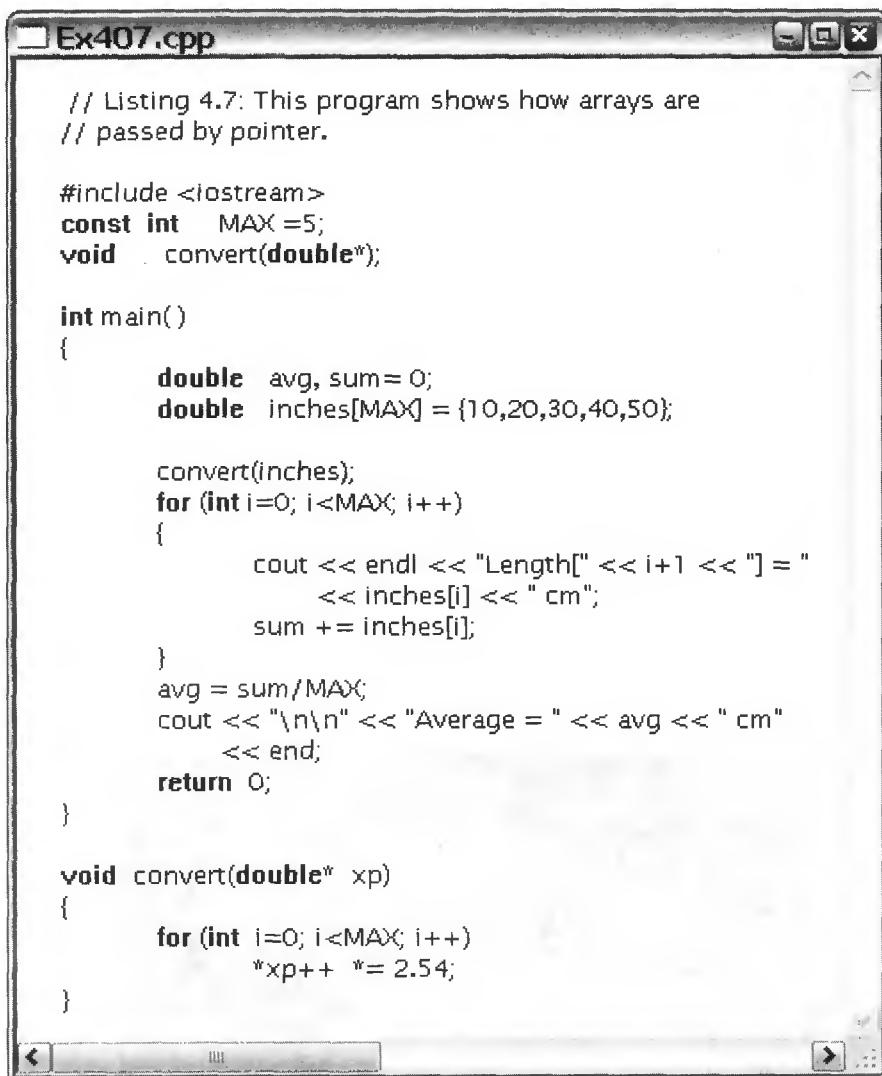
```
#include <iostream>
void convert(double* xp)
{
    // convert inches to centimeters
    *xp *= 2.54;
}

int main()
{
    double inches;
    cout << "Enter inches : ";
    cin >> inches;
    convert(&inches);
    cout << "Length = " << inches << " centimeters\n";
    return 0;
}
```

ပုံ (၄. ၁၄)

၄၁ Passing Arrays by Pointers

function တွေထက် array တွေ passလုပ်ခိုင်းတဲ့အခါမှာ အများအားဖြင့် array notation နဲ့ pass လုပ်တာထက် pointer နည်းကို အသုံးပြုတာများပါတယ်။ ဗုဒ္ဓ (၄.၁၅) က Ex407.cpp program မှာ array passed by pointer နည်းကိုဖော်ပြထားပါတယ်၊ လေ့လာကြည့်ပါ။



The screenshot shows a Windows Notepad window with the title "Ex407.cpp". The code inside the window is as follows:

```
// Listing 4.7: This program shows how arrays are
// passed by pointer.

#include <iostream>
const int MAX = 5;
void convert(double*);

int main()
{
    double avg, sum = 0;
    double inches[MAX] = {10,20,30,40,50};

    convert(inches);
    for (int i=0; i<MAX; i++)
    {
        cout << endl << "Length[" << i+1 << "] = "
            << inches[i] << " cm";
        sum += inches[i];
    }
    avg = sum/MAX;
    cout << "\n\n" << "Average = " << avg << " cm"
        << endl;
    return 0;
}

void convert(double* xp)
{
    for (int i=0; i<MAX; i++)
        *xp++ *= 2.54;
}
```

ဤ (၄.၁၅)

- စစချင်: main() function ထဲမှာ type double variable (2) ဖြစ်တဲ့ avg နဲ့ sum တို့ကို define လုပ်ပြီး sum = 0 လို့ initialize လုပ်ပါတယ်။ နောက်ပြီး inches နာမည်နဲ့ array တစ်ခုကို define । initialize လုပ်ပါတယ်။ array element အရေအတွက်ဖြစ်တဲ့ MAX ကို 5 လို့ အစကတည်းက သတ်မှတ်ပေးထားပြီးသားပါ။ ဒါတော့ inches[0] = 10 ၊ inches[1] = 20 ၊ inches[2] = 30 အစရိုသည်တို့ ဖြစ်မှာပါ။
- convert() function ကို call ခေါ်ပါတယ်။ passing argument က inches array ရဲ့ address ဖြစ်ပါတယ်။ array name ဟာ address ပဲ ဖြစ်တာမို့လို့ convert (inches) လို့ pass လုပ်ရင်ရပါတယ်။ inches array ကို pointer to double xp ကနေလက်ခံပါတယ်။
- for loop ကိုပတ်ခိုင်းပြီး xp က point လုပ်နေတဲ့ inches address နေရာက element တစ်ခုချင်းကိုဖွဲ့ယူပြီး လက်မကနေစင်တီမိတာကို ပြောင်းပေးနေပါပြီ။ array element တစ်ခု ကနေ နောက် element နေရာကို ပြောင်းပေးချင်ရင် pointer xp ကို indirection operator အသုံးပြုပြီး *xp++ လို့ increment တိုးပေးရင်ရပါတယ်။ တစ်ခိုန်တည်းမှာပဲ main() ထဲက inches element တွေအားလုံးဟာ စင်တီမိတာတွေဖြစ်နေပြီလေ။
- သေချာအောင် main() function ထဲက inches array element တန်ဖိုးတွေကို display လုပ်ကြည့်ပါ။ sum+= inches[i]; ကိုရေးထားတဲ့ အဓိပ္ပာယ်က array element အားလုံးရဲ့ တန်ဖိုးတွေကိုပေါင်းခိုင်းပြီး sum နဲ့ assign လုပ်ပေးတာပါ။ နောက်ပြီး array element တွေရဲ့ ပျမ်းမျှတန်ဖိုးကို တွက်ခိုင်းပြီး display လုပ်ခိုင်းပါမယ်။ ပုံ (၄။၁၆) ဟာဆိုရင် program run ပြထားတာပါ။ output result ကိုလေ့လာကြည့်ပါ။

```

Quincy 99
Length[1] = 25.4 cm
Length[2] = 50.8 cm
Length[3] = 76.2 cm
Length[4] = 101.6 cm
Length[5] = 127 cm
average = 76.2 cm
Any key to return to Quincy...

```

ပုံ (၄။၁၆)

Swapping Two Numbers with Pointers

ကျွန်တော်တိတွေ pointer အသုံးပြုပြီး array တွေ access လုပ်နည်းတိ သိသွားပြီဆိုလိုရှိရင် တစ်ဆင့် ကော်ပြီးတော့ array element တွေစိတ်နည်းကို လက်တွေ့စမ်းကြည့်လိုပါပြီ။ အလွယ်ခံးအနေနဲ့ကောင်း (2) ခု ရုံး function တစ်ခုမှာ pass လုပ်ခိုင်းပြီး pointer အကူအညီနဲ့ သူတို့ကိုနေရာချင်းပြောင်းပေးမယ်လေ။ ပုံ (၄၁၁၃)

မှာရေးထားတဲ့ program ကိုလေ့လာကြည့်ပါ။

The screenshot shows a Windows application window titled "Ex408.cpp". Inside the window, there is a code editor displaying C++ code. The code is as follows:

```
// Listing 4.8: This program swaps two numbers
// passed to function as argument.

#include <iostream>
void swap(int*, int*);

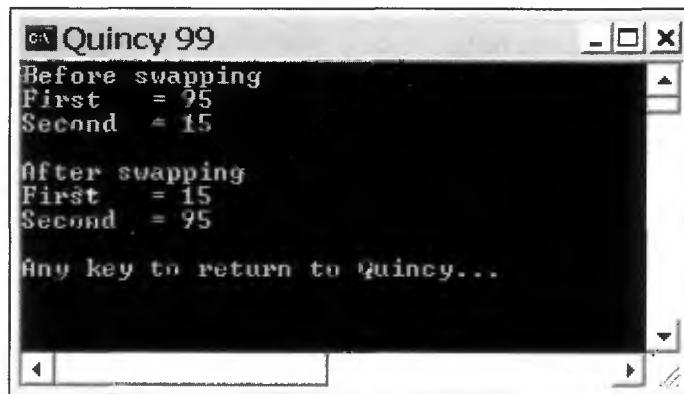
int main()
{
    int first=95, second=15;
    cout << "Before swapping";
    cout << "\nFirst = " << first
        << "\nSecond = " << second;

    swap(&first, &second);
    cout << "\n\nAfter swapping";
    cout << "\nFirst = " << first
        << "\nSecond = " << second << endl;
    return 0;
}

void swap(int *x, int *y)
{
    int temp = *x;
    *x = *y;
    *y = temp;
}
```

ပုံ (၄၁၁၃)

- စစချင်းမှာ first = 95, second = 15 လို့ integer variable (2) ခုကို initialize လုပ်ပါတယ်။ swap မလုပ်ခင်မှာဒီဂါဏ်နဲ့ (2) ခုကို display လုပ်ပြီးတော့ swap() function ကို call ခေါ်ပါတယ်။ passing argument တွေက first နဲ့ second တို့၏ address တွေပါ။
- ဒါ address တွေဟာ pointers to integers first နဲ့ second ဖြစ်တဲ့ x နဲ့ y ထဲကို pass လုပ်ဝင်သွားပါပြီ။ swap() function ထဲမှာ *x နဲ့ *y ကိုနေရာချင်းပြောင်းပေးပါတယ်။ဒါ ဆိုရင် *x = 15 နဲ့ *y = 95 ဖြစ်သွားပါပြီ။ main() function ကိုဖြန်ရောက်တဲ့အခါမှာ ဒါ ဂါဏ်နဲ့ (2) ခုကို display လုပ်ပြီးမှာပါ။
- Ex408.cpp program ကို run လိုက်ရင် output result ကို ပုံ (၄. ၁၈) မှာပြထားတဲ့အတိုင်း မြင်ရပါလိမ့်မယ်။



```
Quincy 99
Before swapping
First    = 95
Second   = 15

After swapping
First    = 15
Second   = 95

Any key to return to Quincy...
```

ပုံ (၄. ၁၈)

Bubble Sort

၁။ ပုံ (၄. ၁၉) မှာပြထားတဲ့ Ex409.cpp program ဘာဆိုရင် array တစ်ခုထဲက element တွေကို ငယ်စဉ်ပြီးလိုက်ဖြစ်အောင် pointer အသုံးပြုပြီး bubble sort နည်းနည်းပေးတဲ့ program ဖြစ်ပါတယ်။ ဒီဥစ္စာကို trace လုပ်ကြည့်မယ်ဆိုရင်

Ex409.cpp

```
// Listing 4.9: This program sorts an array
// using the bubble sort method

#include <iostream>
void swap(int*, int*);
void bubbleSort(int*, int);

int main()
{
    const int N=10;
    int x[N] = {10,50,20,70,40,30,90,80,60,100};

    cout << "\nBefore sorting\n";
    for(int j=0;j<N;j++)
        cout << x[j] << " ";
    bubbleSort(x,N);
    cout << "\n\nAfter sorting\n";
    for(int j=0;j<N;j++)
        cout << x[j] << endl;
    return 0;
}

void swap(int *x, int *y)
{
    if(*x > *y) {
        int temp = *x;
        *x = *y;
        *y = temp;
    }
}

void bubbleSort(int* xp, int n)
{
    for(int j=0; j<(n-1); j++)
        for(int k=j+1; k<n;k++)
            swap(xp+j, xp+k);
}
```

♂ (ç. œ)

- bubbleSort() function ထဲမှာ inner loop ပတ်နေတုန်း ပထမဂဏန်း (10) နဲ့ ဒုတိယဂဏန်း (50) ကိုနှင့် ယူဉ်ကြည့်ပါတယ်။ ဒုတိယဂဏန်းက ပထမဂဏန်းထက်ငယ်တယ်ဆိုရင် နေရာချင်းလဲ ပါမယ်။ မငယ်ဘူးဆိုရင် ပထမဂဏန်းနဲ့ တတိယဂဏန်း (20) နဲ့ ဆက်ယူဉ်ကြည့်မယ်။ တတိယ ဂဏန်းက ပထမဂဏန်းထက်မငယ်ဘူးဆိုတော့ နေရာချင်းမလဲတော့ပါဘူး။ ဒီနည်းအတိုင်း ဆက်လုပ်သွားရင် outer loop အတွက် ပထမအကြိမ်ပြီးသွားပါပြီ။ ပထမအကြိမ်ပြီးတဲ့အခါမှာ X[] array ကအခုလိုဖြစ်နေပါလိမ့်မယ်။ ဘာမှာပြောင်းလဲမှုမရှိသေးပါဘူး။

10, 50, 20, 70, 40, 30, 90, 80, 60, 100

- outer loop ဒုတိယအကြိမ်စရင် X[] array ရဲ့ဘယ်ဘက်အစွန်ဆုံးဂဏန်းဟာ အငယ်ဆုံးဖြစ်နေပါလိမ့်မယ်။ အဲဒီဂဏန်းကို ခန်ဖိယ်ထားပါ။ အခုတစ်ခါတော့ ပထမဂဏန်းကို (50) လို့သတ်မှတ်ရပါမယ်။ ကျေန်တဲ့ဂဏန်း (9) လုံးထဲက အငယ်ဆုံးဖြစ်တဲ့ (20) ကို ဒုတိယဘယ်အစွန်နေရာရောက်အောင် စောစောကလိုပဲ swap လုပ်ခဲ့ရင် X[] array ကအခုလိုဖြစ်နေပါလိမ့်မယ်။

10, 20, 50, 70, 40, 30, 90, 80, 60, 100

- ဒီနည်းအတိုင်းဆက်လုပ်သွားရင် X[] array element တွေဟာ အခုလို တစ်ဆင့်ပြီးတစ်ဆင့်နေရာပြောင်းသွားပြီး sorting ဟာပြီးသွားပါလိမ့်မယ်။

10, 20, 30, 50, 70, 40, 90, 80, 60, 100

10, 20, 30, 40, 50, 70, 90, 80, 60, 100

10, 20, 30, 40, 50, 60, 70, 90, 80, 100

10, 20, 30, 40, 50, 60, 70, 80, 90, 100

- Ex409.cpp program ကို run လိုက်ရင် output result ကို ပုံ (၄. ၁၈) မှာပြထားတဲ့အတိုင်း မြင်ရပါလိမ့်မယ်။

The screenshot shows a terminal window titled "Quincy 99". The output is as follows:

```

Before sorting
10 5и 2и 7и 40 30 90 80 60 100

After sorting
10
2и
3и
40
5и
6и
7и
8и
9и
100

Any key to return to Quincy...

```

ပုံ (၄၁၂၀)

၄.၆ Pointers and Strings

၁။ string ဆိတာ character type array အမျိုးအစားတစ်ခုဖြစ်ပါတယ်။ ဒီတော့ array element တွေကို pointer notation နဲ့ access လုပ်လိုရင် string တွေကိုလည်းလုပ်လိုရဖြေပေါ့။ string array တစ်ခုကို ဒီတိုင်း increment လုပ်လိုမရပေမယ့် pointer ကိုတော့လုပ်လိုရပါတယ်။ ပုံ (၄၁၂၀) မှာဖော်ပြထားတဲ့ Ex4010.cpp program ဟာဆိုရင် pointer notation ကိုအသုံးပြုပြီး ရေးထားတာဖြစ်ပါတယ်။

၂။ Ex4010.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- str နဲ့ strp (pointer to str) တိုကို define လုပ်ပြီး initialize လည်းလုပ်ပါတယ်။ နောက်ပြီး str နဲ့ strp တိုကို display လုပ်ကြည့်တော့ရပါတယ်။
- ဒီတစ်ခါ for loop ထဲမှာ strp ကို increment လုပ်ပေးခြင်းအားဖြင့် string str ထဲက character (8) လုံးဟာ pointer variable ထဲမှာ store ဖြစ်သွားပါပြီ။ strp ကို display လုပ်ကြည့်ရင် အဖြေပေါ်လာပါလိမ့်မယ်။

```
Ex4010.cpp

// Listing 4.10: This program shows how the strings
// can be defined using array and pointer notations

#include <iostream>

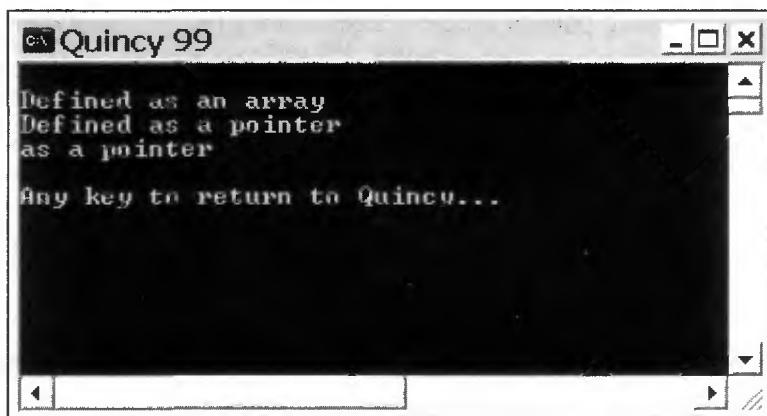
int main()
{
    char str[] = "Defined as an array";
    char* strp = "Defined as a pointer";

    cout << endl << str << endl << strp;

    for (int i=0; i<8; i++)
        strp++;
    cout << endl << strp << endl;
    return 0;
}
```

ဗုဒ္ဓ (၄။၂၂)

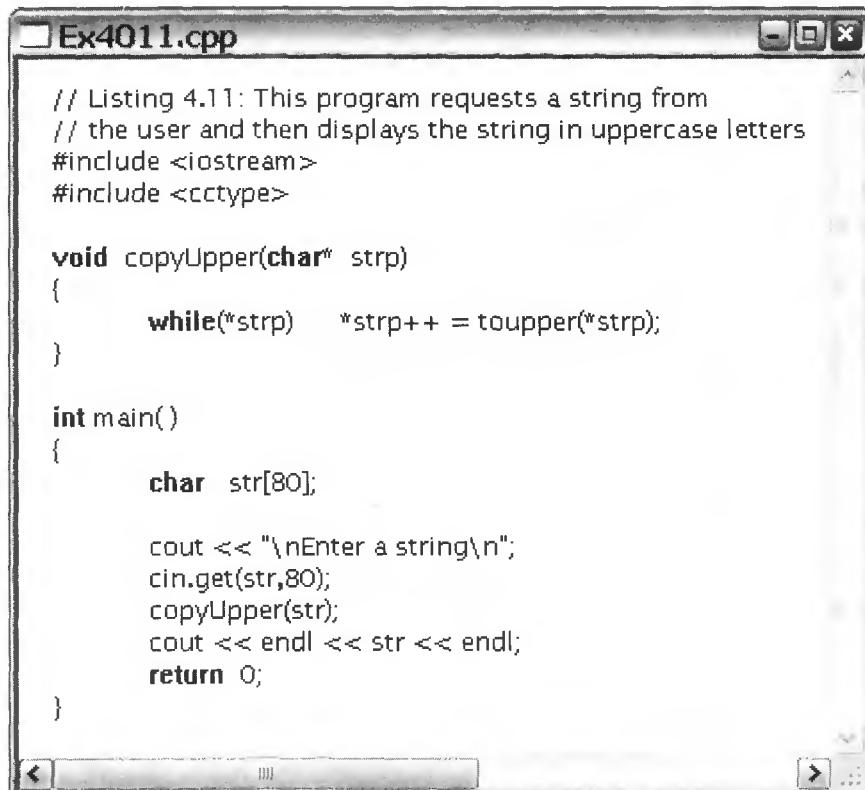
- Ex4010.cpp program ကို run လိုက်ရင် output result ကို ဗုဒ္ဓ (၄။၂၂) မှာဖြတေးတဲ့အတိုင်း
မြင်ရပါလိမ့်မယ်။ str string ထဲက character (8) လုံးပဲပေါ်နေတာကိုတွေ့မှာပါ။



ဗုဒ္ဓ (၄။၂၂)

Changing Characters of an Array to Uppercase

ପ୍ରକାଶ କାମିଙ୍ଗ୍ ମୁଦ୍ରଣଙ୍କାରୀଙ୍କରେ Ex4011.cpp program ହାତୁ character array ତଳ୍ଳାଯିଲା element ଏବଂ uppercase letter ଟେକ୍ସ୍ଟିଙ୍କ୍ୟୁନ୍ଡର୍ ଏବଂ pointer ଅବ୍ସର୍ପିଙ୍ଗିଲ୍ଲାର୍ ପାଇଁ program ଫ୍ରିଟିକାର୍ଯ୍ୟ ପାଇଁ ହେଉଥିଲା।



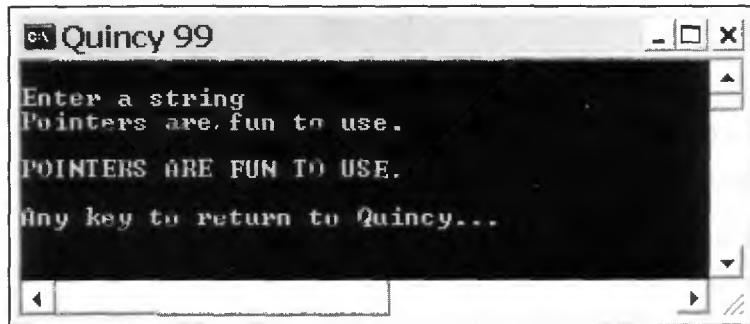
```
// Listing 4.11: This program requests a string from
// the user and then displays the string in uppercase letters
#include <iostream>
#include <cctype>

void copyUpper(char* strp)
{
    while(*strp)    *strp++ = toupper(*strp);
}

int main()
{
    char str[80];

    cout << "\nEnter a string\n";
    cin.get(str,80);
    copyUpper(str);
    cout << endl << str << endl;
    return 0;
}
```

ପ୍ରକାଶ କାମିଙ୍ଗ୍



ପ୍ରକାଶ କାମିଙ୍ଗ୍

Strings as Function Arguments

၁။ ပုံ (၄။ ၂၅) မှာပြထားတဲ့ Ex4012.cpp program ဟာဆိုရင် string တစ်ခုကို function တစ်ခုမှာ passing by pointer နည်းနဲ့ pass လုပ်တာကို program ရောပြထားတာဖြစ်ပါတယ် । လေ့လာကြည့်ပါ။

```
// Listing 4.12: This program shows a string
// used as a function argument. The function
// displays the string each character in turn.

#include <iostream>
void displayStr(char*);

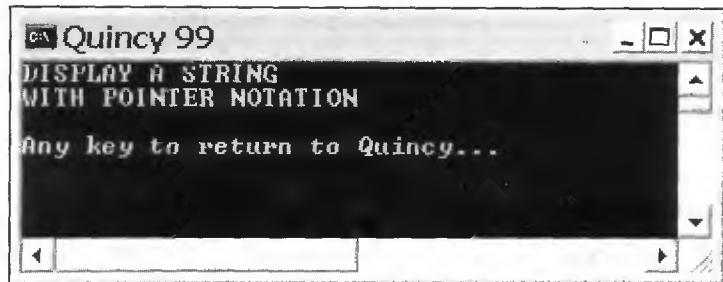
int main()
{
    char str1[] = "DISPLAY A STRING\n";
    char str2[] = "WITH POINTER NOTATION\n";

    displayStr(str1);
    displayStr(str2);
    return 0;
}

void displayStr(char* cp)
{
    while (*cp) cout << *cp++;
}
```

ပုံ (၄။ ၂၅)

၂။ Ex4012.cpp program ကို run လိုက်မယ်ဆိုရင် ပုံ (၄။ ၂၆) မှာပြထားတဲ့အတိုင်း output result ကိုမြင်ရမှာပါ။ string တစ်ခုကို pointer လဲ pass လုပ်ပေးပြီး pointer ကို increment လုပ်ရင်း string တဲ့ character တစ်ခုချင်းကို display လုပ်ပြသွားတာပါပဲ။



Copying a String Using Pointers

၁။ ဒု (၅. J2) မှာဖော်ပြထားတဲ့ Ex4013.cpp program ထဲ main() function ထဲမှာ copyStr() function ကို call ခေါ်ပြီ: blank string တစ်ခုနဲ့ pointer array တစ်ခုကို pass လုပ်ပေးပါတယ်။ copyStr

```
// Listing 4.13: This program copies one string
// to another using pointers.

#include <iostream>
void copyStr(char*, char*);

int main()
{
    char* strp = "Teach Yourself C++ Complete";
    char str[80];

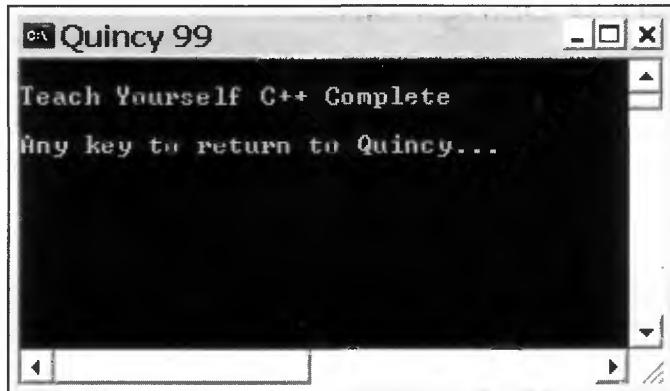
    copyStr(str, strp );
    cout << endl << str << endl;
    return 0;
}

void copyStr(char* to, char* from)
{
    while (*from)
        *to++ = *from++;
    *to = '\0';
}
```

ဒု (၅. J2)

function ထဲ ရောက်လာတဲ့အခါမှာ from pointer array ကို increment လုပ်ရင်နဲ့ blank string array ကို point လုပ်နေတဲ့ to pointer array ကိုလည်း increment လုပ်သွားပေးပါမယ်။ to++ ကို from++ နဲ့ တစ်ခုပြီးတစ်ခု assign လုပ်ပေးသွားတဲ့သဘောပေါ့။ ဒါဆိုရင် copy ကူးတဲ့အလုပ်ပြီးသွားပါပြီ။

J# Ex4013.cpp program ကို run လိုက်မယ်ဆိုရင် Teach Yourself C++ Complete ဆိုတဲ့ string ဟာ string str ထဲကို copy ကူးဝင်သွားပါတယ်။ ဒီတော့ str ကို display လုပ်ရင် စောစောကကူးဝင်လာတဲ့ စာကြောင်းကို ပုံ (၄၁၂) မှာပြထားတဲ့အတိုင်းမြင်ရမှာပါ။



ပုံ (၄၁၂)

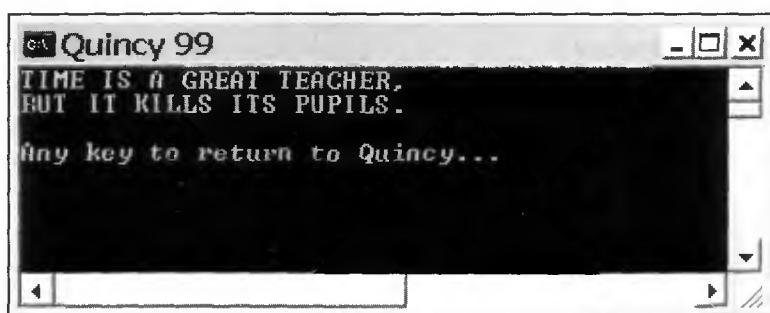
၄.၇ Arrays of Pointers to Strings

၁။ integer type ၊ float type ဖြစ်တဲ့ array တွေရှိထဲလို့ pointer type array တွေလည်း ရှိပါတယ်။ string တွေကို နှီးနှီးတန်းတန်း array ပုံစံမျိုးနဲ့ access လုပ်တဲ့အခါမှာ အားနည်းချက်တစ်ခုက subarray တွေဟာ တူညီတဲ့ string length လေးတွေကိုပဲ အသုံးပြုလိုပါတယ်။ ဒီတော့ မလိုတဲ့ space နေရာတွေဟာ ဖြုန်းသလိုဖြစ်တဲ့ တာပေါ့။ အဲဒီအားနည်းချက်ကို pointer type array တွေအသုံးပြုခြင်းအားဖြင့် လုံးဝချေဖျက်လိုပါပြီ။ ပုံ (၄၃၂) မှာဖော်ပြထားတဲ့ Ex4014.cpp program ကိုလေ့လာကြည့်ရင် သဘောပေါက်ပါလိမ့်မယ်။

```
// Listing 4.14: This program creates an array of pointers  
// to strings, rather than an array of strings.  
// to another using pointers.  
  
#include <iostream>  
const int MAX= 7;  
  
int main()  
{  
    char* strp[MAX] = {"TIME ", "IS ", "A GREAT ",  
                       "TEACHER,\n", "BUT ", "IT ",  
                       "KILLS ITS PUPILS."};  
  
    for (int i=0; i<MAX; i++)  
        cout << strp[i];  
  
    cout << endl;  
    return 0;  
}
```

ပုံ (၄။၂)

၂။ Ex4014.cpp program ကို run လိုက်မယ်ဆိုလိုရင် strp[i] pointer array ထဲက pointer element တစ်ခုချင်းကနေ point လုပ်နေတဲ့ string အချက်အစားအမျိုးမျိုးကို ပို့ဆုံး array တစ်ခုရဲ့ element တွေကို display လုပ်နည်းအတိုင်း display လုပ်ပြပါလိမ့်မယ်။ ပုံ (၄။၂) ကိုကြည့်ပါ။



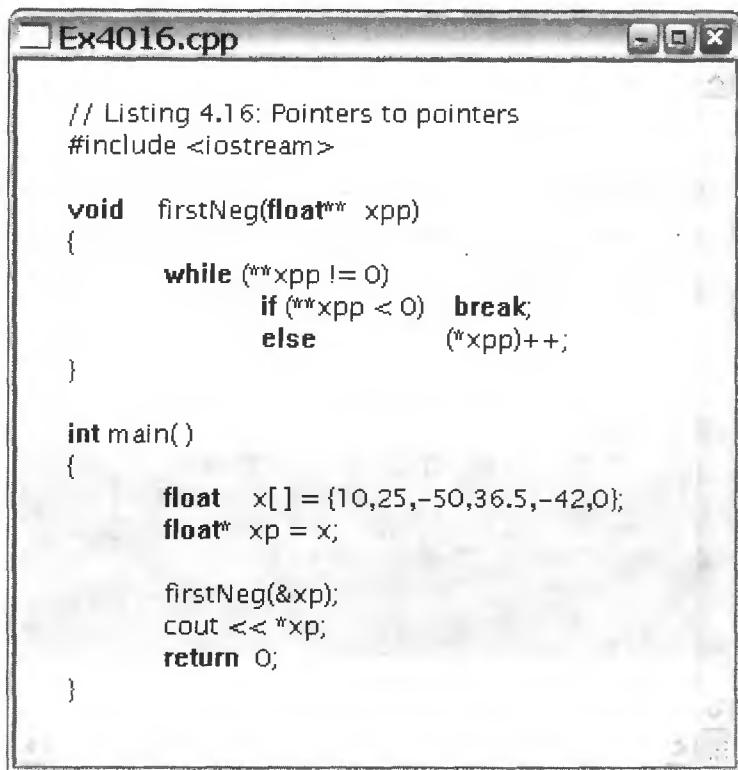
ပုံ (၄။၃)

၄.၈ Pointers to Pointers

၁။ pointer ကနေ pointer ကို pointလုပ်ပေးချင်ရင် * (asterisk) (2) ခုကို အသုံးပြုရပါမယ်။ ဥပမာ
cp ဟာ char pointer တစ်ခုဆိုရင် char** cpp = &cp; လိုပေးလိုပါတယ်။ cpp ဟာ pointer to
pointer ဖြစ်ပါတယ်။ pointer to pointer ကို initialize လုပ်ပေးချင်ရင် အခုံလိုအောက်ဖြည့်ပါ။

```
char c = 'A';           // char variable
char* cp = &c;          // pointer to char variable
char** cpp = &cp;        // pointer to pointer
```

pointer to pointer ကနေ point လုပ်နေတဲ့ pointer သို့မဟုတ် char variable တွေက data တွေ
ကိုထုတ်ယူချင်ရင် char*cp1 = *cpp; သို့မဟုတ် char c1 = **cpp; လိုပေးပြီး retrieve လုပ်ရင်ရပါတယ်။
ကောင်းပြီ၊ ပုံ (၄.၂၉) မှာရေးထားတဲ့ Ex4016.cpp program ကိုလေ့လာကြည့်ရင် pointer to pointer ကို
သဘောပေါက်ပါလိမ့်မယ်။



```
// Listing 4.16: Pointers to pointers
#include <iostream>

void firstNeg(float*** xpp)
{
    while (**xpp != 0)
        if (**xpp < 0) break;
        else (*xpp)++;
}

int main()
{
    float x[] = {10,25,-50,36.5,-42,0};
    float* xp = x;

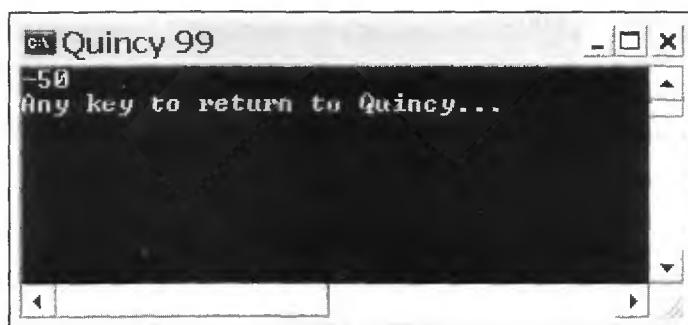
    firstNeg(&xp);
    cout << *xp;
    return 0;
}
```

ပုံ (၄.၂၉)

Ex4016.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- စစချင်းမှာ float type array ဖြစ်တဲ့ x ကို initialize လုပ်ပါတယ်။ နောက်ပြီး x ကို point လုပ်နေတဲ့ pointer xp ကို define လုပ်ပါတယ်။ firstNeg() function ကို call ခေါ်ပါတယ်။ function မှာ pointer address ကို pass လုပ်ပေးပါတယ်။
- called function မှာ argument ကို pointer to pointer xpp ဖြစ်ပါတယ်။ float **xpp လို့ define လုပ်ထားတာကိုကြည့်ပါ။ while(**xpp != 0) ဆိုတဲ့ statement ဟာ ဆိုရင် pass ဝင်လာတဲ့ (**xpp ကို point လုပ်နေတဲ့ x[] array element) ကို သူ့လူဟုတ် မဟုတ်စိစစ်မှာပါ။ မဟုတ်ဘူးဆိုရင် if (**xpp < 0) break; statement ကနေ (x[] array element ဟာ negative value ဖြစ်ခဲ့ရင်) function ထဲကပြန်ထွက်ခိုင်းပါလိမ့်မယ်။ တစ်ကယ်လို့ negative value မဟုတ်သေးဘူးဆိုရင် **xpp ကို increment လုပ်ခြင်းအားဖြင့် next element ကို point လုပ်ပေးမှာပါပဲ။

၃။ Ex4016.cpp program ကို run လိုက်မယ်ဆိုရင် ပထမဆုံးတွေ့တဲ့ negative value ကို display လုပ်ပြပါလိမ့်မယ်။ ပုံ (၄၀.၃၀) ကိုကြည့်ပါ။

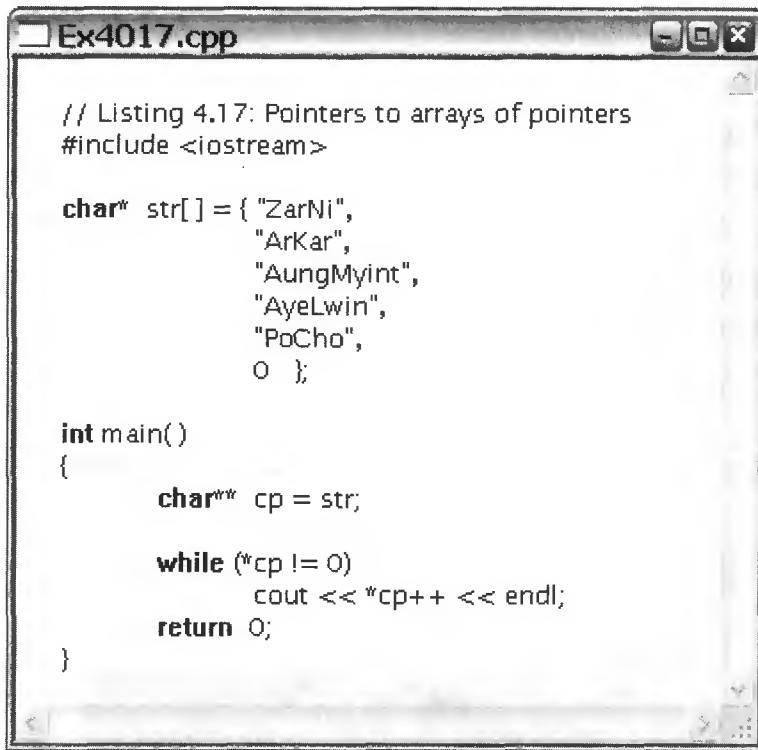


ပုံ (၄၀.၃၀)

Pointers to Arrays of Pointers

၁။ pointer to pointer ကိုနောက်တစ်မျိုးအသုံးပြုလိုပောက် pointer array တွေကို point လုပ်ချင်တဲ့ အခါမှာဖြစ်ပါတယ်။ တစ်ချို့က ရှုံးရှုံး array မျိုးကိုအသုံးပြုပောက် pointer array အမျိုးအစားကိုအသုံးပြုတာ

ပိုကောင်းတယ်လိုပြောပါတယ်။ ဗု (၄။ ၃၁) က Ex4017.cpp program ဟာဆိုရင် pointer array တစ်ခုကို pointer to pointer နဲ့ access လုပ်နည်းကိုတင်ပြထားတာပါ။ လေ့လာကြည့်ပါ။



```
// Listing 4.17: Pointers to arrays of pointers
#include <iostream>

char* str[] = { "ZarNi",
                 "ArKar",
                 "AungMyint",
                 "AyeLwin",
                 "PoCho",
                 0 };

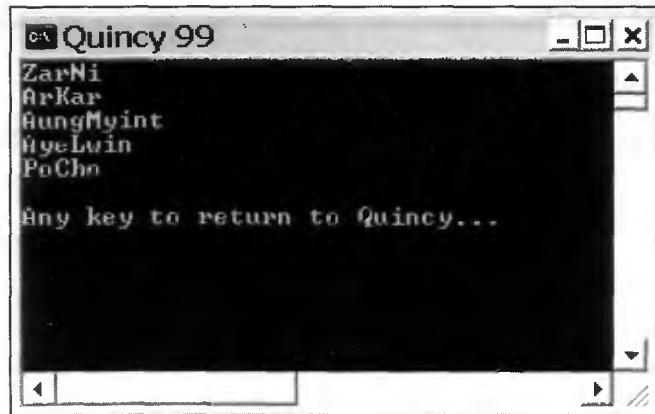
int main()
{
    char*** cp = str;

    while (*cp != 0)
        cout << *cp++ << endl;
    return 0;
}
```

ဗု (၄။ ၃၁)

၂။ Ex4017.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- စစချင်းမှာ char pointer array တစ်ခုကို နာမည်တွေနဲ့ initialize လုပ်ပါတယ်။ point array ရဲ့နောက်ဆုံး element ကို null pointer လုပ်ထားပါတယ်။ main() function ထဲမှာ pointer to pointer cp ကို pointer array နဲ့ assign လုပ်ပေးပြီး while loop ထဲမှာ *cp ကို pointer array မကုန်မချင်း တစ်ခုစီ display လုပ်သွားပါတယ်။ *cp++ ဆိုတာ next pointer array element တွေကို point ဆက်လုပ်သွားတာဖြစ်ပါတယ်။
- ဗု (၄။ ၃၂) မှာ Ex4017.cpp program ကို run ပြထားပါတယ်။



ပု (၄။ ၃၂)

Pointers to const Variables

၁။ ပု (၄။ ၃၃) မှာရေးထားတဲ့ Ex4018.cpp program မှာ pointer to const variable ရေးနည်းကို
တင်ပြထားပါတယ်။ main() function ထဲမှာ const char variable တစ်ခုဖြစ်တဲ့ str2[] ကို string တစ်ခုနဲ့
initialize လုပ်ပေးပြီး copyUpper() function ကို call ခေါ်ပါတယ်။ calling function မှာ str1 နဲ့ str2
char array (2) ခုကို pass လုပ်ပေးပြီး called function ကနေ char pointer တွေဖြစ်တဲ့ s1 ၊ s2 တို့က
လက်ခံပါတယ်။ function return type ကို char pointer လိုအပ်မှတ်ပေးထားတာကို သတိပြုကြည့်ပါ။
called function ထဲမှာ s1 ကို point လုပ်နေတဲ့ pointer ကို s လို့ assign လုပ်ပေးပြီး s2 point လုပ်နေတဲ့
str2 ထဲက character တွေကိုတစ်ခုချင်း uppercase letter ဖြစ်အောင်ပြောင်းရင်းနဲ့ *s1++ နဲ့ assign လုပ်
ပေးပါတယ်။ အားလုံးပြောင်းပြီးသွားရင် s1 ကို point လုပ်နေတဲ့ pointer s ကို main() ဆီ return လုပ်ပေးမှပါ။
ဒါတော့ main() function မှာ str1 ကို display လုပ်ခိုင်းရင် str2 ကို uppercase ပြောင်းထားတာကို
တွေ့ရပါလိမ့်မယ်။

၂။ Ex4018.cpp program ကို run လိုက်မယ်ဆိုရင် ကွန်ပူ၍တာမှာ BEFORE... နဲ့ AFTER... ဆိုတဲ့
တော်း (4) ပြောင်းကို တွေ့ရပါလိမ့်မယ်။ ပု (၄။ ၃၄) ကိုကြည့်ပါ။

Ex4018.cpp

```
// Listing 4.18: const pointer arguments
#include <iostream>
#include <cctype>

char* copyUpper(char* s1, const char* s2)
{
    char* s = s1;

    while ((*s1++ = (toupper(*s2++))) != '\0')
        return s;
}

int main()
{
    char str1[80];
    const char str2[] = "Be the first to know C++";

    cout << "\nBEFORE...\n" << str2 << endl;

    copyUpper(str1, str2);
    cout << "\nAFTER...\n" << str1 << endl;
    return 0;
}
```

◊ (§. 22)

Quincy 99

```
BEFORE...
Be the first to know C++

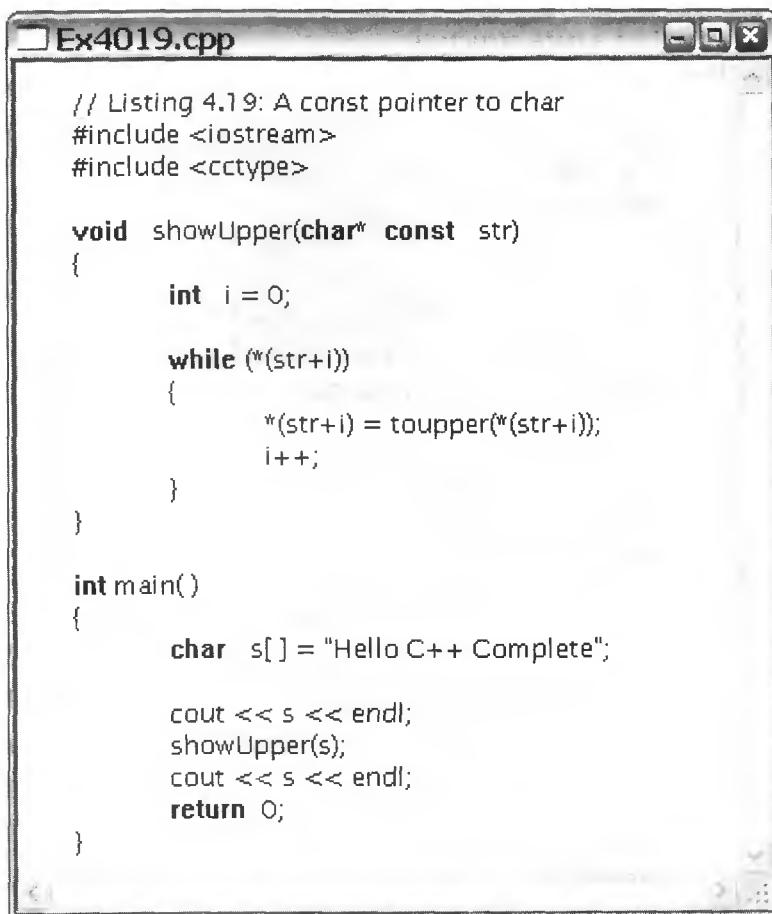
AFTER...
BE THE FIRST TO KNOW C++

Any key to return to Quincy...
```

◊ (§. 29)

const Pointer Variables

ဗု (၄။ ၃၄) က Ex4019.cpp program မှာ char array s[] ကို showUpper() function ထဲ
pass လုပ်တဲ့အခါမှာ array address အဖြစ် pass လုပ်ပေးပြီး called function မှာ char const pointer
variable တစ်ခွာဖြစ်လက်ခံပါတယ်။ အဲဒီ const pointer variable ကို uppercase ဖြစ်အောင်ပြောင်း
လိုက်တဲ့အခါမှာ မူလ char array s ဘာ uppercase letter တွေချည်းဖြစ်ကုန်ပါပြီ။ ဒါ program ကို run
လိုက်ရင် Hello C++ complete နဲ့ HELLO C++ COMPLETE ဆိုတဲ့စာကြောင်းတွေပေါ်လာမှာပါ။ စာဖတ်သူ
ကိုယ်တိုင် လက်တွေ့လုပ်ကြည့်ပါ။



```
// Listing 4.19: A const pointer to char
#include <iostream>
#include <cctype>

void showUpper(char* const str)
{
    int i = 0;

    while (*str+i)
    {
        *(str+i) = toupper(*str+i);
        i++;
    }
}

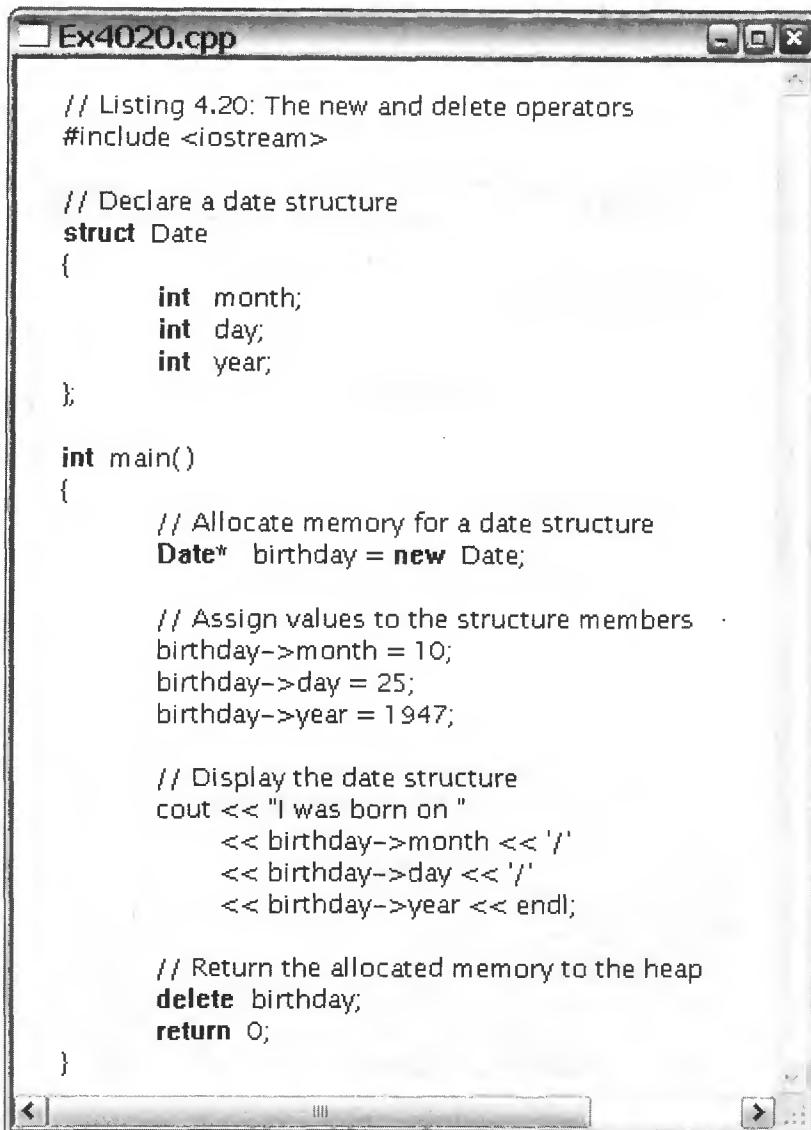
int main()
{
    char s[] = "Hello C++ Complete";

    cout << s << endl;
    showUpper(s);
    cout << s << endl;
    return 0;
}
```

ဗု (၄။ ၃၅)

၄.၆ Memory Allocation

၁။ ကွန်ပျူတာတစ်ခုရဲ့ global storage memory ကို free store သိမဟုတ် heap လိုခေါ်ပါတယ်။ heap ဆိတာ program အတွက် အသုံးပြုလိုရတဲ့ RAM အရွယ်အစား ဖြစ်ပါတယ်။ program တစ်ခုအတွက် memory allocation ကို ထတ်မှတ်ပေးချင်ရင် new ဆိတ္တ operator ကိုအသုံးပြုရမှာပါ။



```
// Listing 4.20: The new and delete operators
#include <iostream>

// Declare a date structure
struct Date
{
    int month;
    int day;
    int year;
};

int main()
{
    // Allocate memory for a date structure
    Date* birthday = new Date;

    // Assign values to the structure members
    birthday->month = 10;
    birthday->day = 25;
    birthday->year = 1947;

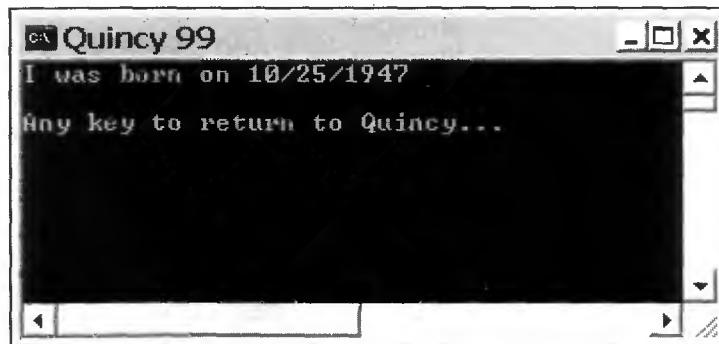
    // Display the date structure
    cout << "I was born on "
        << birthday->month << '/'
        << birthday->day << '/'
        << birthday->year << endl;

    // Return the allocated memory to the heap
    delete birthday;
    return 0;
}
```

ပုံ (၄၃၃)

ဒါနိုင် heap memory ကနေ program အတွက် memory block တွေကို assign လုပ်ပေးပါလိမ့်မယ်။ program ပြီးသွားလို့ block တွေကို heap ဆိုပြန်ပြုပေးချင်ရင် delete operator ကိုအသုံးပြုရပါမယ်။ ကောင်းပြီး (၄. ၃၅) မှာဖော်ပြထားတဲ့ Ex4020.cpp program မှာ new နဲ့ delete operator တွေအသုံးပြုပုံကို တင်ပြထားပါတယ်။ လေ့လာကြည့်ပါ။

Ex4020.cpp program ကို လေ့လာကြည့်မယ်ဆိုရင် main() function ထဲမှာ စစချင်း birthday date တစ်ခုအတွက် memory block ကို heap ကနေရယူပါတယ်။ ပြီးတော့ရင် အဲဒီ date မှာ month၊ day၊ year value တွေကို assign လုပ်ပေးပါတယ်။ ဒါဆိုရင် birthdaymonth ကို display လုပ်ရင် birthday အတွက် month value ကို တွေ့ရမှာပါ။ birthday date အတွက် memory allocation ကို ပြန်ဖျက်ချင်ရင် delete birthday; လို့ ရေ့တာနဲ့ရပါတယ်။ ပုံ (၄. ၃၆) မှာ Ex4020.cpp program ကို run ပြထားပါတယ်။



ပုံ (၄. ၃၆)

Using new and delete Operators with Arrays

တစ်ကယ်လို့ birthday date ကို array တစ်ခုအနေနဲ့ဖော်ပြုမယ်ဆိုရင် memory allocation ကိုအခုံဖို့ပြင်ရေးရပါမယ်။ ပုံ (၄. ၃၇) မှာဖော်ပြထားတဲ့ Ex4021.cpp program ဟာ Ex4020.cpp program ကို ပြင်ရေးထားတာဖြစ်ပါတယ်။ array အတွက် allocated memory ကို ဖျက်ချင်ရင် delete [] birthday လို့ ရေးရပါမယ်။ သတိထားပါ။ ဒါ ပြီး program ကို run ရင် စောစောကအဖြေပဲရမှာပါ။

```

// Listing 4.21: The new and delete operators
#include <iostream>

int main()
{
    // Get memory for an array of integers
    int* birthday = new int[3];

    // Assign values to the array elements
    birthday[0] = 10;
    birthday[1] = 25;
    birthday[2] = 1947;

    // Display the values in the array
    cout << "I was born on "
        << birthday[0] << '/'
        << birthday[1] << '/'
        << birthday[2] << endl;

    // Return the allocated memory to the heap
    delete [] birthday;
    return 0;
}

```

ပုံ (၄။၃၃)

Allocating Dynamic Arrays

၁။ ပုံ (၄။၃၈) မှာဖော်ပြထားတဲ့ Ex4022.cpp program ဘာ variable array dimension တစ်ခုကို new operator အသုံးပြုပြီး allocate လုပ်နည်းကိုဖော်ပြထားပါတယ် । လေ့လာကြည့်ပါ။

၂။ Ex4022.cpp program ကို run လိုက်ပြီး Enter the array size : လို prompt လုပ်ပြတဲ့အခါမှာ (5) ကိုရှိက်ထည့်မယ်ဆုံးရင် ကွန်ပူးတာမှာ random number (5) လုံးပေါ်လာတာကိုတွေ့ရပါလိမ့်မယ်။ အဲဒါ လုပ်ပြီးရင် array အတွက် memory allocation ကို program က delete လုပ်ပေးလိုက်ပါပြီ။ ပုံ (၄။၃၉) ကိုကြည့်ပါ။

Ex4022.cpp

```
// Listing 4.22: The new operator and dynamic arrays
# include <iostream>
# include <cstdlib>

int main()
{
    int size;

    // Get the array size from the user
    cout << "Enter the array size: ";
    cin >> size;

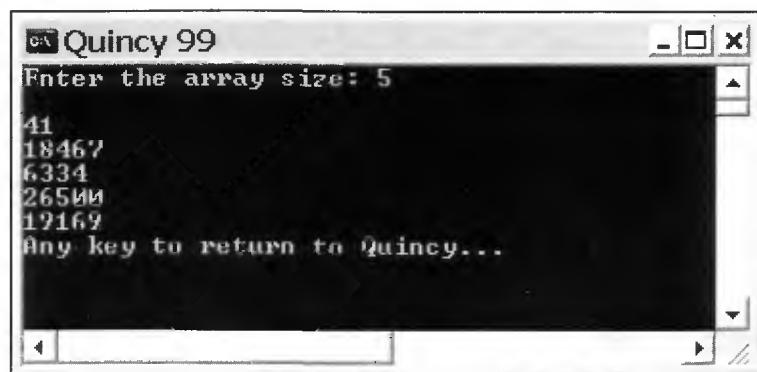
    // Allocate memory for the array
    int* arrayX = new int[size];

    // Load the array with random numbers
    for (int i=0; i<size; i++)
        arrayX[i] = rand();

    // Display the array with random numbers
    for (int i=0; i<size; i++)
        cout << endl << arrayX[i];

    // Return the allocated memory to the heap
    delete [ ] arrayX;
    return 0;
}
```

♀ (g. 20)



♀ (g. 26)

၄.၁၀ Using Reference Variables

၁။ reference variable ဆိတ် data object တစ်ခုရဲ့ copy မဟုတ်သလို data object ကို point လုပ်နေတဲ့ pointer လည်းမဟုတ်ပါဘူး။ address ချင်းတူညီတဲ့ တစ်ခြား variable တစ်ခုရဲ့အမည်ကဲ့ (alias) တစ်ခုပါပဲ။ variable တစ်ခုကို &(ampersand) operator အသုံးပြုပြီး reference variable ဖြစ်အောင် သတ်မှတ်ပေးလို့ရပါတယ်။ ပုံ (၄.၄၀) မှာဖော်ပြထားတဲ့ Ex4023.cpp program မှာ reference variable ရဲ့ သဘောတရားကို လေ့လာလို့ရပါတယ်။

```
// Listing 4.23: Reference variable
#include <iostream>

int main()
{
    int actualint = 1234;
    int& otherint = actualint;

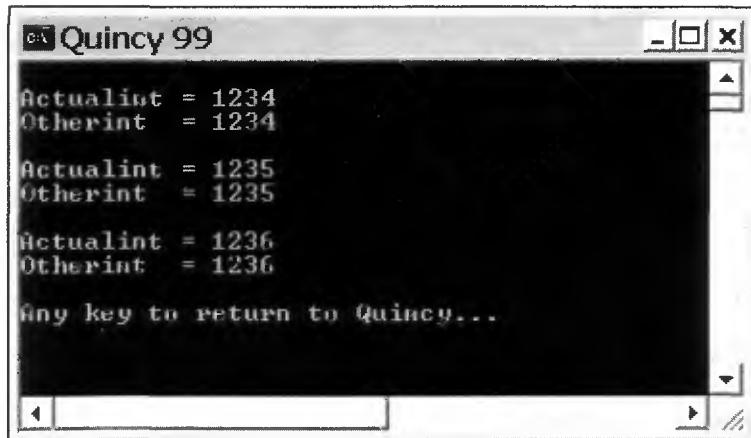
    cout << "\nActualint = " << actualint << endl;
    cout << "Otherint = " << otherint << endl;
    otherint++;

    cout << "\nActualint = " << actualint << endl;
    cout << "Otherint = " << otherint << endl;
    actualint++;

    cout << "\nActualint = " << actualint << endl;
    cout << "Otherint = " << otherint << endl;
    return 0;
}
```

ပုံ (၄.၄၀)

၂။ Ex4023.cpp program ကို run လိုက်မယ်ဆိုရင် ပုံ (၄.၄၀) မှာပြထားတဲ့အတိုင်းမြင်ရမှာပါ။ actualint နဲ့ otherint တို့ရဲ့ ဆက်သွယ်မှုကိုလေ့လာကြည့်ပါ။



ပုံ (၄။၄၀)

References to Reduce Complex Notation

၁။ C++ program မှာ structure member တွေလို အရည်ကြီးဖော်ပြရမယ့်နေရာမှာ reference ကို အသုံးပြုရင် လွယ်ကူတာကိုတွေ့ရပါတယ်။ အဲဒီအတွက် Ex4024.cpp program ကို နမူနာအဖြစ် အောက်မှာရေး ပြထားပါတယ်။ လေ့လာကြည့်ပါ။

// Listing 4.24: Initializing a reference

```
#include <iostream>

struct Date
{
    int month,day,year;
};

struct EmployeeRec
{
    int emplNo;
    char name [80];
    Date dates [3];
    float salary;
};
```

```

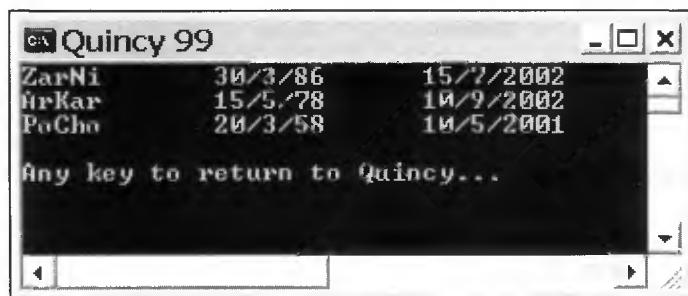
EmployeeRec    staff[ ] =
{
    {001, "ZarNi",{{30,3,86},{15,7,2002}}, 45000},
    {002, "ArKar",{{15,5,78},{10,9,2002}}, 38000},
    {003, "PoCho",{{20,3,58},{10,5,2001}}, 35000},
    {0}
};

int main( )
{
    EmployeeRec* xp = staff;

    while (xp->emplNo != 0)
    {
        cout << xp->name << "\t\t";
        for (int i=0; i<2; i++)
        {
            Date& rd = xp->dates[i];
            cout << rd.month << '/'
            << rd.day << '/'
            << rd.year << "      ";
        }
        cout << endl;
        xp++;
    }
    return 0;
}

```

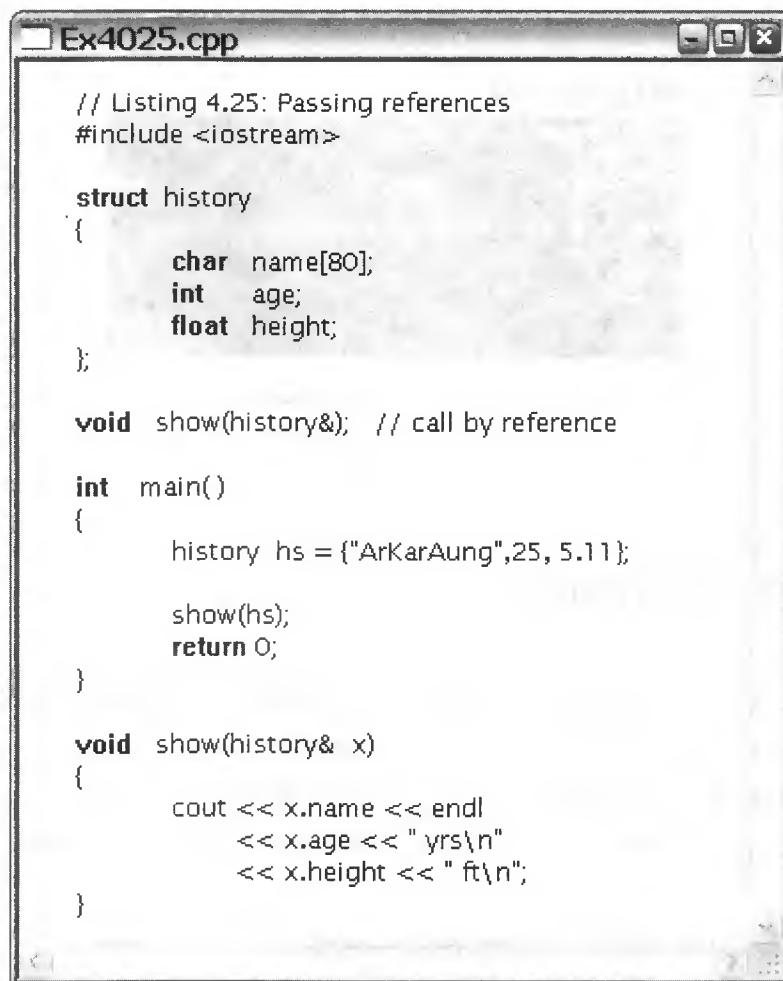
၃ Ex4023.cpp program ၊ run လိုက်မယ်ဆိုရင် ဖူ (၄။၄၂) မှာဖြတေးတဲ့အတိုင်းမြင်ရမှာပါ။



ဖူ (၄။၄၂)

Passing References

၁။ function တစ်ခုကနေ reference တစ်ခုကို တစ်ခြား function ဆီ pass လုပ်ပေးမယ့်အိုင် calling function က argument copy ကို called function တအသုံးပြုမှာပါ။ called function ရဲ့ local copy အသုံးမပြုပါဘူး။ ဗုဒ္ဓ (၄၀၄၃) မှာဖော်ပြထားတဲ့ Ex4025.cpp program ဟာ reference pass လုပ်နည်း ရေးပြထားပါတယ်၊ လေ့လာကြည့်ပါ။



```
// Listing 4.25: Passing references
#include <iostream>

struct history
{
    char name[80];
    int age;
    float height;
};

void show(history&); // call by reference

int main()
{
    history hs = {"ArKarAung", 25, 5.11};

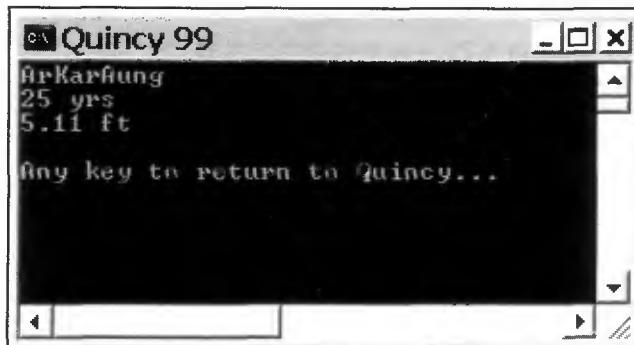
    show(hs);
    return 0;
}

void show(history& x)
{
    cout << x.name << endl
        << x.age << " yrs\n"
        << x.height << " ft\n";
}
```

ဗုဒ္ဓ (၄၀၄၃)

JII Ex4025.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- စေချင်းမှာ structure type history ကို define လုပ်ပြီး show() ဆိုတဲ့ prototype ကို လည်း declare လုပ်ပါတယ်။
- main() function ထဲမှာ history instance တစ်ခုဖြစ်တဲ့ hs ကို initialize လုပ်ပြီး အော့ show() function ကို call ခေါ်ပါတယ်။ called function ထဲမှာ call by reference နည်းကိုအသုံးပြုပြီး hs member တစ်ခုချင်းကို display လုပ်ပြခိုင်းထားပါတယ်။
- ပုံ (၄၁၅၅) ဟာ Ex4025.cpp program ကို run ပြထားတာပါ။



ပုံ (၄၁၅၅)

Returning References

- ၁။ reference တစ်ခုကို parameter တစ်ခုအနေနဲ့ function ထဲ pass လုပ်ပေးတဲ့နည်းကို စာဖတ်သူ သိသွားပြီမဟုတ်လား။ တစ်ကယ်လို့ reference တစ်ခုကို function ကနေ return လုပ်ပေးတာမျိုးကိုလုပ်ချင်ရင် ရမလား။ ဥပမာ Date& getDate(); ဆိုတာမျိုးပေါ့။ အဲဒါမျိုးလည်း လုပ်လို့ရပါတယ်။ ပုံ (၄၁၅၆) မှာဖော်ထားတဲ့ Ex4026.cpp program ဟာ reference return လုပ်နည်းကို ရေးပြထားပါတယ်။

JII Ex4026.cpp program ကိုလေ့လာကြည့်မယ်ဆိုရင်

- စေချင်းမှာ structure type Date ကို define လုပ်ပြီး Date instance ဖြစ်တဲ့ birthdays

```
Ex4026.cpp

// Listing 4.26: Returning references
#include <iostream>

struct Date
{
    int month, day, year;
};

Date birthdays[] = { {12, 10, 1995},
                        { 7, 11, 1998},
                        {10, 5, 2000},
                        {15, 7, 2001},
                        { 9, 10, 2002},
};

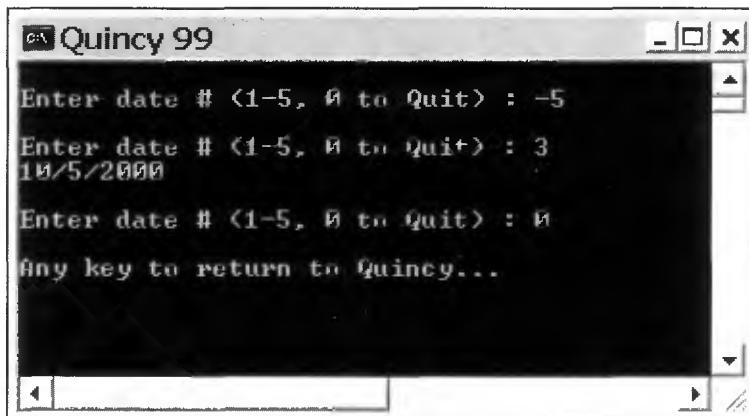
const Date& getDate(int n)
{
    return birthdays[n-1];
}

int main()
{
    int dt = 2003;
    while (dt != 0)
    {
        cout << "\nEnter date # (1-5, 0 to Quit) : ";
        cin >> dt;
        if (dt>0 && dt<6)
        {
            const Date& rd = getDate(dt);
            cout << rd.month << '/' << rd.day << '/'
                << rd.year << endl;
        }
    }
    return 0;
}
```

ပုံ (၄၁၄၅)

ကို initialize လုပ်ထားပါတယ်။ နောက်ပြီ Date တစ်ခုကို retrieve လုပ်ပေးနိုင်တဲ့ getDate function ကိုလည်းရေးထားပါတယ်။

- main() function ထဲမှာစစချင်း dt ကို positive ကဏန်းတစ်ခုနဲ့ assign လုပ်ပေးခြင်းအားဖြင့် while loop ထဲကိုဝင်လိုပါပြီ။ dt value ကို (-5) လိုထည့်ကြည့်ပါ။ if block statement ထဲဝင်လိုပေါ်ပါဘူး။ ဒါနိုင် dt value ကို (3) ရိုက်ထည့်ကြည့်ပါ။ if block ထဲကိုဝင်လိုပေါ်ပြီ။ const Date& rd = getDate(dt); ဆိတဲ့ statement ကနေ local reference rd ကို returned reference getDate(dt) နဲ့ initialize လုပ်ပေးပါတယ်။
- returned reference function ထဲရောက်လာတဲ့အခါမှာ n = 3 ဖြစ်သွားပါပြီ။ ဒီတော့ birthdays[3-1] ကို return လုပ်ခိုင်းတဲ့အခါမှာ birthdays[2] = {10,5,2000} ကို main() ထဲက rd နဲ့ assign ပြန်လုပ်ပေးမှာပါ။ နောက်ပြီး rd member တွေကိုတစ်ခုချင်း display လုပ်ခိုင်းထားပါတယ်။ ပုံ (၄၁၄၆) ဟာ Ex4026.cpp program ကို run ပြထားတာပါပဲ။



ပုံ (၄၁၄၆)

const References

ပုံ (၄၁၄၇) မှာဖော်ပြထားတဲ့ Ex4027.cpp program မှာစစချင်း struct type Date ရဲ့ instance တစ်ခုဖြစ်တဲ့ dt ကို {10, 5, 2002} လို့ initialize လုပ်ပေးထားပါတယ်။ နောက်ပြီး const reference to Date တစ်ခုဖြစ်တဲ့ drကို dt နဲ့ assign လုပ်ပေးပြီး dr member တစ်ခုချင်းကို ကွန်ပျူတာမှာ display ပြန်လုပ်ခိုင်းထားတာပါ။ Ex4027.cpp program ကို ပုံ (၄၁၄၈) မှာ run ပြထားပါတယ်။ လေ့လာကြည့်ပါ။

Ex4027.cpp

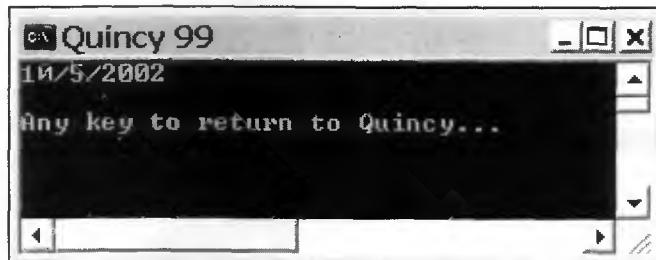
```
// Listing 4.27: const references
#include <iostream>

struct Date
{
    int month, day, year;
} dt = {10,5,2002};

int main()
{
    const Date& dr = dt;

    cout << dr.month << '/'
        << dr.day << '/'
        << dr.year << endl;
    return 0;
}
```

දැ (ශ. 92)



දැ (ශ. 93)

const Reference Parameters

Ex4027.cpp program තුළ const reference parameter pass ලද්දක්වයි. ප්‍රාග්‍රැම් සඳහා මෙම ප්‍රාග්‍රැම් නොමැත්තු කළ යුතුයි.

Ex4028.cpp

```
// Listing 4.28: const reference parameters

#include <iostream>
struct Date
{
    int month, day, year;
} dt = {10,5,2002};

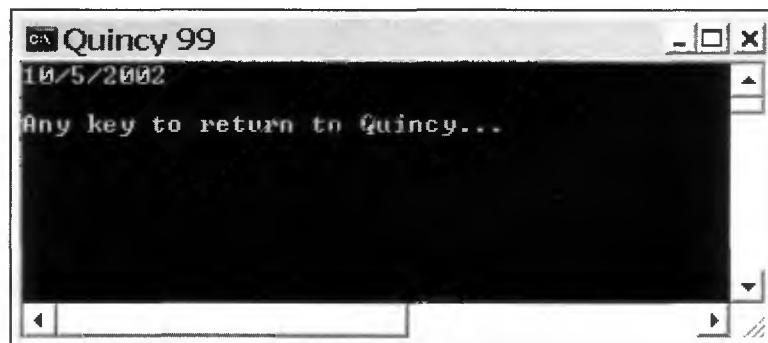
void displayDate (const Date&);

int main()
{
    const Date& dr = dt;

    displayDate (dr);
    return 0;
}

void displayDate (const Date& x)
{
    cout << x.month << '/'
        << x.day << '/'
        << x.year << endl;
}
```

◊ (9. 98)



◊ (9. 99)