CMF MSU

Introduction to C++

Artem Isaev, Sergey Nikulin

2016 CMF

Useful Libraries and key points

- #include – include directive, library connection.
- using namespace std; - using directive
- #include<iostream> - input/output and standard library
- #include <cmath> - mathematic library
- #include <fstream> - file input/output

Example:

```
#include<iostream>
using namespace std;
int main()
{
cout<<"Welcome to the C++ introduction course"<<endl;
return 0;
}
```

Types of data

There are a lot of different types in C++, which are important to use in your C++ programs.

Type	Low	High	Precision	Size, byte
bool	False	True	No	1
Char	-128	127	No	1
short	-32768	32767	No	2
int	-2147483648	2147483647	No	4
long	-2147483648	2147483647	No	4
float	$3.4*10^{-38}$	$3.4*10^{38}$	7	4
double	$1.7*10^{-308}$	$1.7*10^{308}$	15	8
Unsigned char	0	255	No	1
Unsigned short	0	65535	No	2
Unsigned int	0	4294967295	No	4
Unsigned long	0	4294967295	No	5

Key operators

- For (init-expression; cond-expression; loop-expression)

  statement;

- While (expression)

  statement;

- If (expression)

  statement1;

  else

  statement2;

Example:

```
for (i = 1 ; i<10 ; i++)
    a = b+i;
while(a<10)
    a = a++;
if(a > b)
    c = a;
else
    c = b
```

Run your first program

This program solves a quadratic equation by prompting the user for the coefficient values:

```cpp
#include<iostream>
#include<cmath>
using namespace std;
int main()
{
int a, b, c; // 3 variables of type integer.
float root; // float is a real type.
float x1, x2;
cout<<"Enter the values of a, b & c"<<endl; // Line 9
cin>>a; // Line 10
cin>>b;
cin>>c;
root=sqrt(b*b-4*a*c);
x1=(-b+root)/(2*a);
x2=(-b-root)/(2*a);
cout<<"The roots are " << x1 << " and " <<x2<<endl; // Line 16
return 0;
}
```

Functions in C++

Like a variable, a function must (obviously) be declared before it is called.

A function declaration has three components: its ***return type***, its ***name*** and its ***parameter list***. Good programming also encourages the use of comments to briefly explain the role of the function

Example: Consider a function, which takes three real numbers and returns the average. A typical function declaration called a prototype would be:

```
float Average(float x, float y, float z);
```

Float - the return type, i.e. a real of type float will be returned upon completion

Average – name of function

(float x, float y, float z) parameter/argument list together with types of parameter

It could be noticed:

The function prototype does not need to contain actual names of the parameters, just the type. So the declaration float Average(float, float, float); is also perfectly legal.

Functions in C++, Example

```cpp
#include<iostream>
using namespace std;
float Average(float, float, float); //function declaration
/* MAIN PROGRAM: */
int main()
{
float x, y, z;
cout << "Enter numbers: "; /* <--- line 11 */
cin >> x >> y >> z;
cout << endl;

cout << "The average of " << x <<", "<< y << " & ";
cout << z <<" is " << Average(x, y, z)<<endl;
return 0;
}
/* END OF MAIN PROGRAM */
/* FUNCTION TO CALCULATE AVERAGE: */
float Average(float x, float y, float z) /* start of function
definition */
{
float aver;
aver = (x+y+z)/3;
return aver;
} /* end of function definition */
/* END OF FUNCTION */
```

Pointers

- It’s an address.
- Most, but not all items within your executing program have an address
- In C++ you pass parameters to a function In C++, you pass parameters to a function by value. This means a copy of the variable is made, used and then thrown away
- Value parameters can’t be changed
- To pass a variable that you want to be changed you can pass it’s address, i.e. a pointer.

Example:

```
void Fail(int victim)
{
    victim++;
}
void Succeed (int * Victim)
{
    *Victim=3;
}
```

So, What is the Pointer?

- A pointer is simply the memory address of a variable, so that a pointer variable is just a variable in which we can store different memory addresses. Pointer variables are declared using a “*", and have data types like the other variables we have seen. Pointers are widely used in C++ to facilitate efficient dynamic processing of data.

  For example, the declaration:

  int number_ptr;*

  states that "number_ptr" is a pointer variable that can store addresses of variables of data type "int".

- A useful alternative way to declare pointers is using a "typedef" construct. For example, if we include the statement:

  typedef int IntPtr*

- We can then go on to declare several pointer variables in one line, without the need to prefix each with a “*":

  IntPtr number_ptr1, number_ptr2, number_ptr3;

Example of financial program, option pricing

The Black-Scholes equation for the price of an option V(S,t) in the absence of dividends is

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0$$

that is solved together with two boundary conditions and one final condition (called the Payoff function). If V(S,t) is a European call option C(S,t) then the condition are:

1. $C(S,t) = 0$ when $S = 0$
2. $C(S,t) \to S$ as $S \to \infty$
3. $C(S,T) = \max(S - E,\ 0)$

Example of financial program, option pricing

The solution gives a pricing formula for Call Option C(S,t) with

$$C(S,t) = SN(d_1) - E\exp(-r(T-t))N(d_2)$$

where

$$d_1 = \frac{\log(S/E) + \left(r + \tfrac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}} \qquad d_2 = \frac{\log(S/E) + \left(r - \tfrac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}$$

$$N(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} \exp\left(-\frac{1}{2}\phi^2\right)d\phi \qquad d_2 = d_1 - \sigma\sqrt{T-t}$$

Example of financial program, option pricing

Consider the following information

$$S = 100,\ (T - t) = 1.0;\ r = 5\%,\ \sigma = 20\%;\ E = 100$$

The following code use the cumulative distribution function for calculating N(x), which will be your first homework on this course. With this function we can calculate d_1 and d_2 for the given variables and parameters, which are then passed through cumulative distribution function as parameters. We need $N(d_1)$ and $N(d_2)$ to calculate the price of call option.

```
double Option, d1, d2;

double S=100, E=100, r=0.05, vol=0.2, tau=1.0; //tau=time to expiry

d1=(log(S/E)+(r+0.5*vol*vol)*tau)/(vol*sqrt(tau));
d2=d1-vol*sqrt(tau);

Option=S*CDF(d1)-E*exp(-r*tau)*CDF(d2);
```

Example of financial program, option pricing

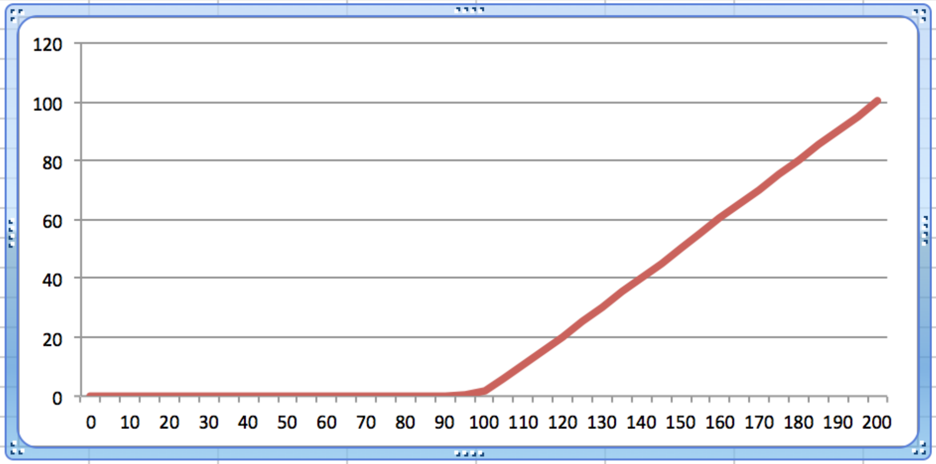
Option pricing realization:

```
#include<iostream>
#include<cmath>
#include<fstream>
using namespace std;
const double pi=4.0*atan(1.0); //define constant pi=3.142
double CDF(double);
int main()
{
double Call_Option, d1, d2;
double S=100, E=100, r=0.05, vol=0.2, tau=0.05;
ofstream out; // create object “out” to printout to excel file
out.open("BSE.xls"); //out.open(“BSE.xls”); - 2 parameters
for (S=0; S<=200; S+=5){
d1=(log(S/E)+(r+0.5*vol*vol)*tau)/(vol*sqrt(tau));
d2=d1-vol*sqrt(tau);
Call_Option=S*CDF(d1)-E*exp(-r*tau)*CDF(d2);
out<<S<<'\t'<<Call_Option<<endl;
}
out.close();
return 0;
}
```

Example of financial program, option pricing

Results of program:

55	0
60	0
65	0
70	0
75	0
80	0
85	0
90	0,02
95	0,3
100	1,91
105	5,54
110	10,27
115	15,25
120	20,25
125	25,25
130	30,25
135	35,25
140	40,25
145	45,25
150	50,25
155	55,25
160	60,25
165	65,25
170	70,25
175	75,25
180	80,25
185	85,25
190	90,25
195	95,25
200	100,25



Literature and Homework

- Gerbert Shildt C++
- Bjarne Straustrup C++
- B. Kernighan & Ritchie. The C Programming Language

- ***Homework***: Write CDF function (CDF.h file), which calculate cumulative distribution function.