Fibonacci series

0 1 1 2 3 5

```
int main()
{
     int n, t1=0, t2=1, next=0;
    cout<<”Enter the value of n”;
cin>>n;
for (int i=1; i <=n i++)
{
    if(i==1)
{
     cout<<t1;
    }
if (i==2)
  {
     cout<<t1<<t2<<endl;
}
next=t1+t2;
t1=t2;
t2=next;
cout<<next<<,<<endl;
}
return 0;
}
```

/*prog to find largest of three numbers using if statement*/

```
int main()
{
        int n1, n2, n3;
cout<<"Enter three numbers";
cin>>n1>>n2>>n3;
```

```
if(n1>=n2 && n1>=n3)
   cout<<"largest is "<<n1;
if(n2>=n1 && n2>=n3)
   cout<<"largest is "<<n2;
if(n3>=n1 && n3>=n2)
   cout<<"largest is "<<n3;


     return 0;
}

return 0;
}
```

//Sample program 1

```
#include<iostream.h>

using namespace std;

int main (  )

{

    cout<<”Welcome to C++ Lab session”<<endl;

    return 0;

}
```

//Sample program 2

```
#include <iostream>
 using namespace std;
int main (void)
```

```
{
        int x, y;    // local variables of type int
        cout <<"Enter the value of x and y";   // Insertion operator
        cin>>x;   // cin>>x>>y    // Extraction operation
        cin >>y;
        cout<<"The entered value of x and y is "<<x<<'\t'<<y;
        return 0;
 }
```

1. **Write a program in C++ to print a welcome text in a separate line**

```
#include <iostream>  // preprocessor directive
using namespace std;

int main()
{
   cout << "Print a welcome text in a separate line :"<<endl;
   cout << "-----------------------------------------"<<endl;
   cout << " Welcome to"<<endl ;
   cout << " Programming in C++ Class "<<endl ;
   return 0;

}
```

2. Write a program in C++ to print the sum of two numbers

```
#include <iostream>
using namespace std;
int main()
{
   cout << "\n\n Print the sum of two numbers :\n";
   cout << "---------------------------------\n";
   cout << " The sum of 29 and 30 is : "<< 29+30 <<"\n" ;
```

```
   return 0;
}
```

```
#include <iostream>
using namespace std;
int main()
{
   cout << "\n\n Print the sum of two numbers :\n";
   cout << "---------------------------------\n";
   int x, y, z;
   cout<<"The two numbers are read from keyboard"<<endl;
   cin >> x>>y;
   z=x+y;
   cout<<"The sum of two numbers is"<<z<<endl;
   return 0;
}
```

primitive data types supported in C/C++

```
char, int , float and double  // 1 byte  2 bytes  4 bytes  8 bytes
```

3. **Write a program in C++ to find Size of fundamental data types**

```
#include <iostream>

int main()
{
    cout << "\n\n Find Size of fundamental data types :\n";
    cout << "----------------------------------------\n";
   cout << " The sizeof(char) is :          " << sizeof(char) << " bytes \n" ;
   cout << " The sizeof(short) is :         " << sizeof(short) << " bytes \n" ;
   cout << " The sizeof(int) is :           " << sizeof(int) << " bytes \n" ;
   cout << " The sizeof(long) is :          " << sizeof(long) << " bytes \n" ;

   cout << " The sizeof(float) is :         " << sizeof(float) << " bytes \n" ;
   cout << " The sizeof(double) is :        " << sizeof(double) << " bytes \n";
   cout << " The sizeof(long double) is :   " << sizeof(long double) << " bytes \n";
   cout << " The sizeof(bool) is :          " << sizeof(bool) << " bytes \n\n";
   return 0;
```

```
}
```

Output:

```
 Find Size of fundamental data types :

----------------------------------------

 The sizeof(char) is :          1 bytes

 The sizeof(short) is :         2 bytes

 The sizeof(int) is :           4 bytes

 The sizeof(long) is :          8 bytes

 The sizeof(float) is :         4 bytes

 The sizeof(double) is :        8 bytes

 The sizeof(long double) is :   16 bytes

 The sizeof(bool) is :          1 bytes
```

4. Write a program in C++ to print the sum of two numbers using variables.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "\n\n Print the sum of two numbers :\n";
      cout << "-----------------------------------\n";
      int a;    // int a, b, sum;  variable declaration
      int b;
      int sum;
      a=29;       // Variable name=value
      b=30;     // variable initialization  int a=29, b=30
      sum=a + b;    // expression operator and operand
      cout << " The sum of "<< a << " and "<<b <<" is : "<< sum <<"\n" ;
}
```

5. Write a program in C++ to check the upper and lower limits of integers.

```
#include <iostream>
#include <climits>   // integer limits in header file
using namespace std;
```

```
int main()
{
    cout << "\n\n Check the upper and lower limits of integer :\n";
    cout << "-------------------------------------------------\n";
    cout << " The maximum limit of int data type :                  " << INT_MAX << endl;
    cout << " The minimum limit of int data type :                  " << INT_MIN << endl;
    cout << " The maximum limit of unsigned int data type :         " << UINT_MAX << endl;
    cout << " The maximum limit of long long data type :            " << LLONG_MAX << endl;
    cout << " The minimum limit of long long data type :             " << LLONG_MIN << endl;
    cout << " The maximum limit of unsigned long long data type :   " << ULLONG_MAX << endl;
       cout << " The Bits contain in char data type :                  " << CHAR_BIT << endl;
       cout << " The maximum limit of char data type :                 " << CHAR_MAX << endl;
       cout << " The minimum limit of char data type :                 " << CHAR_MIN << endl;
       cout << " The maximum limit of signed char data type :          " << SCHAR_MAX << endl;
       cout << " The minimum limit of signed char data type :          " << SCHAR_MIN << endl;
       cout << " The maximum limit of unsigned char data type :        " << UCHAR_MAX << endl;
       cout << " The minimum limit of short data type :                " << SHRT_MIN << endl;
    cout << " The maximum limit of short data type :                " << SHRT_MAX << endl;
    cout << " The maximum limit of unsigned short data type :       " << USHRT_MAX << endl;
   cout << endl;
   return 0;
}
```

Output:

```
 Check the upper and lower limits of integer :

-------------------------------------------------

 The maximum limit of int data type :                  2147483647

 The minimum limit of int data type :                  -2147483648

 The maximum limit of unsigned int data type :         4294967295

 The maximum limit of long long data type :            9223372036854775807

 The minimum limit of long long data type :             -9223372036854775808

 The maximum limit of unsigned long long data type :   18446744073709551615

 The Bits contain in char data type :                  8

 The maximum limit of char data type :                 127

 The minimum limit of char data type :                 -128

 The maximum limit of signed char data type :          127

 The minimum limit of signed char data type :          -128

 The maximum limit of unsigned char data type :        255

 The minimum limit of short data type :                -32768

 The maximum limit of short data type :                32767

 The maximum limit of unsigned short data type :       65535
```

6. Write a program in C++ to display various type or arithmetic operation using mixed data type

```
#include <iostream>
#include <iomanip>   // formatting floating-point numbers with 1 decimal place
using namespace std;

int main()
{
  int m1 = 5, m2 = 7;        // integer datatype
  double d1 = 3.7, d2 = 8.0;     // double datatype
    cout << "\nDisplay arithmetic operations with mixed data type :\n";
  cout << fixed << setprecision(1);
   cout <<" "<< m1 << " + " << m2 << " = " << m1+m2 << endl;
  cout <<" "<< d1 << " + " << d2 << " = " << d1+d2 << endl;

  cout <<" "<< m1 << " + " << d2 << " = " << m1+d2 << endl;
   cout <<" "<< m1 << " - " << m2 << " = " << m1-m2 << endl;

  cout <<" "<< d1 << " - " << d2 << " = " << d1-d2 << endl;
  cout <<" "<< m1 << " - " << d2 << " = " << m1-d2 << endl;

   cout <<" "<< m1 << " * " << m2 << " = " << m1*m2 << endl;
  cout <<" "<< d1 << " * " << d2 << " = " << d1*d2 << endl;
  cout <<" "<< m1 << " * " << d2 << " = " << m1*d2 << endl;
   cout <<" "<< m1 << " / " << m2 << " = " << m1/m2 << endl;
  cout <<" "<< d1 << " / " << d2 << " = " << d1/d2 << endl;
  cout <<" "<< m1 << " / " << d2 << " = " << m1/d2 << endl;
  cout << endl;
  return 0;
}
```

7. Write a program in C++ to display the operation of pre and post increment and decrement

int x=10;

++x; //preincrement

x++; // postincrement

```
#include <iostream>
using namespace std;

int main()
{
   int num = 57;
```

```
    cout << "\nDisplay the operation of pre and post increment and decrement :\n";

   cout <<" The number is : " << num << endl;
    num++;                // increase by 1 (post-increment)
    cout <<" After post increment by 1 the number is : " << num << endl;
   ++num;              // increase by 1 (pre-increment)
   cout <<" After pre increment by 1 the number is : " << num << endl;
   num = num + 1;  // num is now increased by 1.
   cout <<" After increasing by 1 the number is : " << num << endl; // 79
   num--;              // decrease by 1 (post-decrement)
   cout <<" After post decrement by 1 the number is : " << num << endl;
   --num;              // decrease by 1 (pre-decrement)
   cout <<" After pre decrement by 1 the number is : " << num << endl;
   num = num - 1; // num is now decreased by 1.
   cout <<" After decreasing by 1 the number is : " << num << endl;
   cout << endl;
   return 0;
}
```

```
Output:
```

Display the operation of pre and post increment and decrement :
The number is : 57
After post increment by 1 the number is : 58
After pre increment by 1 the number is : 59
After increasing by 1 the number is : 60
After post decrement by 1 the number is : 59
After pre decrement by 1 the number is : 58
After decreasing by 1 the number is : 57

8. Write a program in C++ to format the output.

```
#include <iostream>
#include <iomanip>    // Needed to do formatted I/O
int main()
{
    cout << "\n\n Formatting the output :\n";
   double pi = 3.14159265; // this is floating point number
   cout << fixed << setprecision(4); // number is set to display with 4 decimal places
   cout <<" The value of pi : " << pi << endl;
   cout << " The value of pi 4 decimal place of total width 8   : |" << setw(8) << pi << "|"
<< endl; // setw() sets the total width
   cout << " The value of pi 4 decimal place of total width 10  : |" << setw(10) << pi <<
"|"<< endl;

   cout << setfill('*'); // setfill() sets to fill the blanks with specified character
   cout << " The value of pi 4 decimal place of total width 8   : |" << setw(8) << pi << "|"
<< endl;
   cout << " The value of pi 4 decimal place of total width 10  : |" << setw(10) << pi <<
"|"<< endl;

   cout << scientific;  // set value in scientific format with exponent
   cout <<" The value of pi in scientific format is : " << pi << endl;
```

```
   bool done = false; // this is boolean variable
   cout <<" Status in number : " << done << endl;
   cout << boolalpha;     // set output in alphabet true or false
   cout <<" Status in alphabet : " << done << endl;
   cout << endl;
   return 0;
}
```

Output:

```
 Formatting the output :
 The value of pi : 3.1416
 The value of pi 4 decimal place of total width 8   : |  3.1416|
 The value of pi 4 decimal place of total width 10  : |    3.1416|
 The value of pi 4 decimal place of total width 8   : |**3.1416|
 The value of pi 4 decimal place of total width 10  : |****3.1416|
 The value of pi in scientific format is : 3.1416e+00
 Status in number : 0
 Status in alphabet : false
```

Write a program in C++ to print the result of the specified operations

```
#include <iostream>
using namespace std;

int main()
{
   cout << "\n\n Print the result of some specific operation :\n";
  cout << "-------------------------------------------------\n";
   cout << " Result of 1st expression is : "<< (-1+4*6) <<"\n";
   cout << " Result of 2nd expression is : "<< ((35+5)%7) <<"\n" ;
   cout << " Result of 3rd expression is : "<< (14+-4*6/11) <<"\n" ;
   cout << " Result of 4th expression is : "<< (2+15/6*1-7%2) <<"\n\n" ;
}
```

Output:

(2+15/6*1-7%2)

(2+2-1)

((35+5) %7) ==== 40 %7 == reminder 5

(2+15/6*1-7%2)
(2+2*1-7%2)
(2+2-1)
(4-1)
3

Print the result of some specific operation :

Result of 1st expression is : 23
Result of 2nd expression is : 5
Result of 3rd expression is : 12
Result of 4th expression is : 3

Write a program in C++ to add two numbers accept through keyboard

```
#include <iostream>
using namespace std;

int main()
{
        int num1, num2, sum;
       cout << "\n Sum of two numbers :\n";
        cout << " Input 1st number : ";
        cin >> num1 ;
        cout << " Input 2nd number : ";
        cin >> num2;
        sum = num1 + num2;
        cout <<" The sum of the numbers is : " << sum << endl;
        return 0;
}
```

Write a program in C++ to swap two numbers using temp variable

num1=100 num2=200

num1=200 num2=100

```
#include <iostream>
using namespace std;

int main()
{
        cout << "\n\n Swap two numbers :\n";
        cout << "----------------------\n";
        int num1, num2, temp;
        cout << " Input 1st number : ";
        cin >> num1 ;  //num1= 100
        cout << " Input 2nd number : ";
        cin >> num2;     // num2=200
        temp=num2;      // temp=200
        num2=num1;     //num2=100
```

```
        num1=temp;     //num1=200
   cout << " After swapping the 1st number is : "<< num1 <<"\n" ;
   cout << " After swapping the 2nd number is : "<< num2 <<"\n\n" ;
}
```

Write a program in C++ to swap two numbers without using temporary variable
Hint: arithmetic operators

```
int a=10 b=5;
a=a+b;    // a=15
b=a-b;   // 15-5=10=b
a=a-b;   // 15-10=5=a
#include<iostream>
using namespace std;
int main()
{
        cout << "\n\n Swap two numbers :\n";
        int a,b;
        cout << " Input 1st number : ";
        cin >> a ;         //10
        cout << " Input 2nd number : ";
        cin >> b;       //5
        a=a+b;     // 15
        b=a-b;           // 15-5=10
        a=a-b;          // 15-10=5
   cout << " After swapping the 1st number is : "<< a <<"\n" ;
   cout << " After swapping the 2nd number is : "<< b <<"\n\n" ;
}
```

Output:

```
 Swap two numbers :
 Input 1st number : 10
 Input 2nd number : 5
 After swapping the 1st number is : 5
 After swapping the 2nd number is : 10
```

Write a program in C++ to calculate the volume of a sphere

```
#include <iostream>
#define PI  3.142          // preprocessor directive
using namespace std;

   int main()
   {
          int rad1;
          float volsp;
          cout << "\n\n Calculate the volume of a sphere :\n";
           cout<<" Input the radius of a sphere : ";
          cin>>rad1;
          volsp=(4*PI*rad1*rad1*rad1)/3;    // volume of sphere= 4/3 *PI*r*r*r
       cout<<" The volume of a sphere is : "<< volsp << endl;
             return 0;
   }
```

Output:

```
 Calculate the volume of a sphere :
---------------------------------------
 Input the radius of a sphere : 5
 The volume of a sphere is : 523.667
```

Write a program in C++ to find the Area and Perimeter of a Rectangle

```
#include <iostream>
using namespace std;

   int main()
   {
          int width, length, area, peri;
          cout << "\n\n Find the Area and Perimeter of a Rectangle :\n";
          cout << "-------------------------------------------------\n";
          cout<<" Input the length of the rectangle : ";
          cin>>length;
          cout<<" Input the width of the rectangle : ";
          cin>>width;
          area=(length*width);
          peri=2*(length+width);
       cout<<" The area of the rectangle is : "<< area << endl;
       cout<<" The perimeter of the rectangle is : "<< peri << endl;
       return 0;
   }
```

Output:

```
 Find the Area and Perimeter of a Rectangle :
-------------------------------------------------
 Input the length of the rectangle : 10
```

```
Input the width of the rectangle : 15
The area of the rectangle is : 150
The perimeter of the rectangle is : 50
```

Write a program in C++ to find the area of any triangle using Heron's Formula

```
#include <iostream>
#include <math.h>
using namespace std;

  int main()
  {
        float side1, side2, side3, area, s;
                cout << "\n\n Find the area of any triangle using Heron's Formula :\n";
                cout << "------------------------------------------------------------\n";
    cout<<" Input the length of 1st side  of the triangle : ";
        cin>>side1;
    cout<<" Input the length of 2nd side  of the triangle : ";
        cin>>side2;
    cout<<" Input the length of 3rd side  of the triangle : ";
        cin>>side3;
                s = (side1+side2+side3)/2;
                area = sqrt(s*(s-side1)*(s-side2)*(s-side3));
    cout<<" The area of the triangle is : "<< area << endl;
    cout << endl;
    return 0;
  }
```

```
 Find the area of any triangle using Heron's Formula :
------------------------------------------------------------
 Input the length of 1st side  of the triangle : 5
 Input the length of 2nd side  of the triangle : 6
 Input the length of 3rd side  of the triangle : 7
 The area of the triangle is : 14.6969
```

Write a program in C++ to find the area and circumference of a circle

```
#include <iostream>
#define PI 3.14159
using namespace std;
  int main()
  {
          float radius, area, circum;
          cout << "\n\n Find the area and circumference of any circle :\n";

         cout<<" Input the radius(1/2 of diameter) of a circle : ";
         cin>>radius;
          circum = 2*PI*radius;
          area = PI*(radius*radius);
    cout<<" The area of the circle is : "<< area << endl;
    cout<<" The circumference of the circle is : "<< circum << endl;
```

```
 return 0;
   }

 Find the area and circumference of any circle :
-----------------------------------------------------
 Input the radius(1/2 of diameter) of a circle : 5
 The area of the circle is : 78.55
 The circumference of the circle is : 31.42
```

Write a program in C++ to convert temperature in Celsius to Fahrenheit

```
#include <iostream>
using namespace std;

int main()
{
   float frh, cel;
                        cout << "\n\n Convert temperature in Celsius to Fahrenheit :\n";
                        cout << "-------------------------------------------------\n";
   cout << " Input the temperature in Celsius : ";
   cin >> cel;
   frh = (cel * 9.0) / 5.0 + 32;
   cout << " The temperature in Celsius    : " << cel << endl;
   cout << " The temperature in Fahrenheit : " << frh << endl;
           cout << endl;
   return 0;
}
```

Write a program in C++ to convert temperature in Fahrenheit to Celsius

```
#include <iostream>
using namespace std;

int main()
{
   float frh, cel;
           cout << "\n\n Convert temperature in Fahrenheit to Celsius :\n";
           cout << "-------------------------------------------------\n";
   cout << " Input the temperature in Fahrenheit : ";
   cin >> frh;
   cel = ((frh * 5.0)-(5.0 * 32))/9;
   cout << " The temperature in Fahrenheit : " << frh << endl;
   cout << " The temperature in Celsius : " << cel << endl;
           cout << endl;
   return 0;
}
```

Write a program in C++ to find the third angle of a triangle

```
#include <iostream>
using namespace std;

int main()
{
   float ang1, ang2, ang3;
           cout << "\n\n Find the third angle of a triangle :\n";
           cout << "-----------------------------------------\n";
```

```
   cout<<" Input the 1st angle of the triangle : ";
   cin>>ang1;
   cout<<" Input the 2nd angle of the triangle : ";
   cin>>ang2;
        ang3=180-(ang1+ang2);
   cout << " The 3rd of the triangle is : " << ang3 << endl;
        cout << endl;
   return 0;
}
```

Write a program in C++ that converts kilometers per hour to miles per hour

```
#include <iostream>
using namespace std;

int main()
{
   float kmph, miph;
        cout << "\n\n Convert kilometers per hour to miles per hour :\n";
        cout << "-------------------------------------------------\n";
   cout << " Input the distance in kilometer : ";
   cin >> kmph;
        miph = (kmph * 0.6213712);
   cout << " The "<< kmph <<" Km./hr. means "<< miph << " Miles/hr." << endl;
        cout << endl;
   return 0;
}
```

Write a program in C++ to compute the total and average of four numbers

```
#include <iostream>
using namespace std;

   int main()
   {
        float n1,n2,n3,n4,tot,avrg;
             cout << "\n\n Compute the total and average of four numbers :\n";
             cout << "-------------------------------------------------\n";
      cout<<" Input 1st two numbers (separated by space) : ";
        cin>> n1 >> n2;
      cout<<" Input last two numbers (separated by space) : ";
        cin>> n3 >> n4;
        tot=n1+n2+n3+n4;
             avrg=tot/4;
      cout<<" The total of four numbers is : "<< tot << endl;
      cout<<" The average of four numbers is : "<< avrg << endl;
      cout << endl;
      return 0;
   }
```

Write a program in C++ to input a single digit number and print a rectangular form of 4 columns and 6 rows

```
#include <iostream>
using namespace std;

   int main()
   {
        int x;
```

```
                cout << "\n\n Make a rectangular shape by a single digit number :\n";
                cout << "-------------------------------------------------------\n";
    cout<<" Input the number : ";
        cin>> x;
    cout<<" "<<x<<x<<x<<x<<endl;
    cout<<" "<<x<<" "<<" "<<x<<endl;
    cout<<" "<<x<<" "<<" "<<x<<endl;
    cout<<" "<<x<<" "<<" "<<x<<endl;
    cout<<" "<<x<<" "<<" "<<x<<endl;
    cout<<" "<<x<<x<<x<<x<<endl;
    cout << endl;
    return 0;
  }
```

Write a program in C++ to check whether a number is positive, negative or zero

```
#include <iostream>
using namespace std;

int main()
{
  signed long num1 = 0;
        cout << "\n\n Check whether a number is positive, negative or zero :\n";
        cout << "-------------------------------------------------------\n";
        cout << " Input a number : ";
  cin >> num1;
  if(num1 > 0)
  {
    cout << " The entered number is positive.\n\n";
  }
  else if(num1 < 0)
  {
    cout << " The entered number is negative.\n\n";
  }
  else
  {
    std::cout << " The number is zero.\n\n";
  }
  return 0;
}
```

Write a program in C++ to test the Type Casting.

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{

   cout << "\n\n Formatting the output using type casting:\n";
        cout << "-----------------------------------------------\n";
        cout<<"Print floating-point number in fixed format with 1 decimal place: ";
  cout << fixed << setprecision(1);
 cout<<"\nTest explicit type casting :\n";
  int i1 = 4, i2 = 8;
  cout << i1 / i2 << endl;
  cout << (double)i1 / i2 << endl;
```

```
  cout << i1 / (double)i2 << endl;
  cout << (double)(i1 / i2) << endl;

  double d1 = 5.5, d2 = 6.6;
cout<<"\nTest implicit type casting :\n" ;
  cout << (int)d1 / i2 << endl;
  cout << (int)(d1 / i2) << endl;
   cout <<"\nint implicitly casts to double: \n";
  d1 = i1;
  cout << d1 << endl;  // 4.0
   cout<<"double truncates to int!: \n";
  i2 = d2;
  cout << i2 << endl;  // 6
}
```

Write a program in C++ that takes a number as input and prints its multiplication table upto 10

```
#include <iostream>
using namespace std;

int main()
{
int a,i=0;
   cout << "\n\n Print the multiplication table of a number upto 10:\n";
            cout << "-------------------------------------------------------\n";
            cout << " Input a number: ";
            cin>> a;
            for (i=1;i<=10;i++)
            {
            cout << a<<" x "<< i << " = "<<a*i<<"\n" ;
            }

}
```

Write a language program in C++ which accepts the user's first and last name and print them in reverse order with a space between them

```
# include <iostream>
# include <string>
using namespace std;
int main()

{
char fname[30], lname [30];
 cout << "\n\n Print the name in reverse where last name comes first:\n";
 cout << "-------------------------------------------------------------\n";
cout << " Input First Name: ";
cin >> fname;
cout << " Input Last Name: ";
cin >> lname;

cout << " Name in reverse is: "<< lname << " "<< fname <<endl;
            cout << endl;
return 0;
}
```

Write a language program which accepts the radius of a circle from the user and compute the area and circumference

```
#include <iostream>
#define PI 3.14159
using namespace std;

  int main()
  {
        float radius, area, circum;
              cout << "\n\n Find the area and circumference of any circle :\n";
              cout << "-----------------------------------------------------\n";
    cout<<" Input the radius(1/2 of diameter) of a circle : ";
        cin>>radius;

              circum = 2*PI*radius;
              area = PI*(radius*radius);
    cout<<" The area of the circle is : "<< area << endl;
    cout<<" The circumference of the circle is : "<< circum << endl;

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

Write a program in C++ which swap the values of two variables not using third variable.

```
#include <iostream>
using namespace std;

int main()
{
        cout << "\n\n Swap two numbers without using third variable:\n";
        cout << "-----------------------------------------------------\n";
        int num1, num2, temp;
        cout << " Input 1st number : ";
        cin >> num1 ;
        cout << " Input 2nd number : ";
        cin >> num2;
        num2=num2+num1;
        num1=num2-num1;
        num2=num2-num1;
  cout << " After swapping the 1st number is : "<< num1 <<"\n" ;
  cout << " After swapping the 2nd number is : "<< num2 <<"\n\n" ;
}
```

Write a program in C++ to print the code (ASCII code / Unicode code etc.) of a given character

```
#include <iostream>
using namespace std;
int main()
{
  char sing_ch;
        cout << "\n\n Print code (ASCII code / Unicode code etc.) of a  given character:\n";
              cout << " Input a character: ";
           cin >> sing_ch;

 cout <<" The ASCII value of "<<sing_ch<<" is: " <<(int)sing_ch << endl;
 cout <<" The character for the ASCII value "<<(int)sing_ch <<" is: "<<(char)((int)sing_ch)<< endl;
 return 0;
}
```

Write a program in C++ to enter P, T, R and calculate Simple Interest.

```
#include<iostream>
using namespace std;

int main()
{
        int p,r,t,i;
                cout << "\n\n Calculate the Simple Interest :\n";
        cout<<" Input the Principle: ";
        cin>>p;
        cout<<" Input the Rate of Interest: ";
        cin>>r;
        cout<<" Input the Time: ";
        cin>>t;

        i=(p*r*t)/100;
        cout<<" The Simple interest for the amount "<<p<<" for "<<t<<"
years @ "<<r<<" % is: "<<i;
       cout << endl;
        return 0;
}
```

Write a program in C++ to enter P, T, R and calculate Compound Interest.

```
#include<iostream>
#include<math.h>

int main()
{
   float p,r,t,cp,ci;
  cout << "\n\n Calculate the Compound Interest :\n";
  cout<<" Input the Principle: ";
 cin>>p;
 cout<<" Input the Rate of Interest: ";
 cin>>r;
cout<<" Input the Time: ";
cin>>t;
 ci=p*pow((1+r/100),t)-p;
 cp=p*pow((1+r/100),t);
 cout<<" The Interest after compounded for the amount "<<p<<" for "<<t<<" years @ "<<r<<" % is: "<<ci;
 cout << endl;
cout<<" The total amount after compounded for the amount "<<p<<" for "<<t<<" years @ "<<r<<" % is: "<<cp;
 cout << endl;
 cout << endl;
 return 0;
```

```
}
```

Write a program in C++ to add two binary numbers

```
#include <iostream>
#include <math.h>
using namespace std;

int main()
{
        long bn1,bn2;
        int i=0, r=0;
        int sum[20];
   cout << "\n\n Addition of two binary numbers:\n";
        cout << "-----------------------------------\n";
        cout << " Input the 1st binary number: ";
        cin>> bn1;
        cout << " Input the 2nd binary number: ";
        cin>> bn2;
  while (bn1 != 0 || bn2 != 0)
  {
   sum[i++] = (int)((bn1 % 10 + bn2 % 10 + r) % 2);
   r = (int)((bn1 % 10 + bn2 % 10 + r) / 2);
   bn1 = bn1 / 10;
   bn2 = bn2 / 10;
  }
  if (r != 0) {
   sum[i++] = r;
  }
  --i;
  cout<<" The sum of two binary numbers is: ";
  while (i >= 0) {
   cout<<(sum[i--]);
  }
   cout<<("\n");
 }
```

Write a C++ program to add all the numbers from 1 to a given number

```
#include <iostream>

int Add_1_to_given_number(int n) {

        int total = 0;

        for (int x = 1; x <= n; x++)
        {
                total += x;
        }
        return total;
}

int main() {

        cout << "\nAdd 1 to 4: " << Add_1_to_given_number(4);
        cout << "\nAdd 1 to 100: " << Add_1_to_given_number(100);
        return 0;
```

```
}
```

Write a C++ program to read an integer n and prints the factorial of n, assume that n = 10

```
#include <iostream>
long long factorial(int num) {
   if (num == 0) {
      return 1;
   }
   else {
      return num * factorial(num-1);
   }
}

int main() {
   int num;
   cin >> num;
   cout << factorial(num) << endl;
}
```

```
#include <iostream>
int main() {
  cout << "Hello World!";
  return 0;
}
```

```
#include <iostream>
using namespace std;

int main() {
  cout << "Hello World!";
  cout << "I am learning C++";
  return 0;
}
```

C++ New Lines

```
#include <iostream>


int main() {
```

```
  cout << "Hello World!" << endl;
  cout << "I am learning C++";
  return 0;
}
```

C++ Multi-line Comments

```
/* The code below will print the words Hello World!
to the screen, and it is amazing */
cout << "Hello World!";
```

Add Variables Together

```
int x = 5;
int y = 6;
int sum = x + y;
cout << sum;
```

Constants

When you do not want others (or yourself) to override existing variable values, use the `const` keyword (this will declare the variable as "constant", which means unchangeable and read-only):

```
const int myNum = 15;  // myNum will always be 15
myNum = 10;  // error: assignment of read-only variable 'myNum'

const int minutesPerHour = 60;
const float PI = 3.14;
```

C++ User Input

```
int x;
cout << "Type a number: "; // Type a number and press enter
cin >> x; // Get user input from the keyboard
cout << "Your number is: " << x; // Display the input value
```

Creating a Simple Calculator

```
int x, y;
```

```
int sum;
cout << "Type a number: ";
cin >> x;
cout << "Type another number: ";
cin >> y;
sum = x + y;
cout << "Sum is: " << sum;
```

C++ String Data Types

String Types

The `string` type is used to store a sequence of characters (text). This is not a built-in type, but it behaves like one in its most basic usage. String values must be surrounded by double quotes:

```
// Include the string library
#include <string>

// Create a string variable
string greeting = "Hello";

// Output string value
cout << greeting;
```

C++ While Loop

The while loop loops through a block of code as long as a specified condition is true:

Syntax

```
while (condition) {
  // code block to be executed
}
```

In the example below, the code in the loop will run, over and over again, as long as a variable (i) is less than 5:

Example

```
int i = 0;
while (i < 5) {
  cout << i << "\n";
  i++;
}
```

The Do/While Loop

The do/while loop is a variant of the while loop. This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.

Syntax

```
do {
  // code block to be executed
}
while (condition);
```

The example below uses a do/while loop. The loop will always be executed at least once, even if the condition is false, because the code block is executed before the condition is tested:

Example

```
int i = 0;
do {
  cout << i << "\n";
  i++;
}
while (i < 5);
```

C++ For Loop

When you know exactly how many times you want to loop through a block of code, use the for loop instead of a while loop:

Syntax

```
for (statement 1; statement 2; statement 3) {
  // code block to be executed
}
```

Statement 1 is executed (one time) before the execution of the code block.

Statement 2 defines the condition for executing the code block.

Statement 3 is executed (every time) after the code block has been executed.

The example below will print the numbers 0 to 4:

Example

```
for (int i = 0; i < 5; i++) {
  cout << i << "\n";
}
```

Another Example

This example will only print even values between 0 and 10:

Example

```
for (int i = 0; i <= 10; i = i + 2) {
```

```
  cout << i << "\n";
}
```

C++ Break and Continue

C++ Break

You have already seen the break statement used in an earlier chapter of this tutorial. It was used to "jump out" of a switch statement.

The break statement can also be used to jump out of a loop.

This example jumps out of the loop when i is equal to 4:

Example

```
for (int i = 0; i < 10; i++) {
  if (i == 4) {
    break;
  }
  cout << i << "\n";
}
```

C++ Continue

The continue statement breaks one iteration (in the loop), if a specified condition occurs, and continues with the next iteration in the loop.

This example skips the value of 4:

Example

```
for (int i = 0; i < 10; i++) {
  if (i == 4) {
    continue;
  }
  cout << i << "\n";
}
```

C++ Arrays

Arrays are used to store multiple values in a single variable, instead of declaring separate variables for each value.

To declare an array, define the variable type, specify the name of the array followed by square brackets and specify the number of elements it should store:

```
string cars[4];
```

We have now declared a variable that holds an array of four strings. To insert values to it, we can use an array literal - place the values in a comma-separated list, inside curly braces:

```
string cars[4] = {"Volvo", "BMW", "Ford", "Mazda"};
```

To create an array of three integers, you could write:

```
int myNum[3] = {10, 20, 30};
```

Access the Elements of an Array

You access an array element by referring to the index number.

This statement accesses the value of the first element in cars:

Example

```
string cars[4] = {"Volvo", "BMW", "Ford", "Mazda"};
cout << cars[0];
// Outputs Volvo
```

Note: Array indexes start with 0: [0] is the first element. [1] is the second element, etc.

Change an Array Element

To change the value of a specific element, refer to the index number:

Example

```
cars[0] = "Opel";
```

Example

```
string cars[4] = {"Volvo", "BMW", "Ford", "Mazda"};
cars[0] = "Opel";
cout << cars[0];
// Now outputs Opel instead of Volvo
```

C++ Arrays and Loops

Loop Through an Array

You can loop through the array elements with the for loop.

The following example outputs all elements in the cars array:

Example

```
string cars[4] = {"Volvo", "BMW", "Ford", "Mazda"};
for(int i = 0; i < 4; i++) {
  cout << cars[i] << "\n";
}
```

The following example outputs the index of each element together with its value:

Example

```
string cars[4] = {"Volvo", "BMW", "Ford", "Mazda"};
for(int i = 0; i < 4; i++) {
```

```
  cout << i << ": " << cars[i] << "\n";
}
```

Omit Array Size

You don't have to specify the size of the array. But if you don't, it will only be as big as the elements that are inserted into it:

```
string cars[] = {"Volvo", "BMW", "Ford"}; // size of array is always 3
```

This is completely fine. However, the problem arises if you want extra space for future elements. Then you have to overwrite the existing values:

```
string cars[] = {"Volvo", "BMW", "Ford"};
string cars[] = {"Volvo", "BMW", "Ford", "Mazda", "Tesla"};
```

If you specify the size however, the array will reserve the extra space:

```
string cars[5] = {"Volvo", "BMW", "Ford"}; // size of array is 5, even though it's only three elements inside it
```

Now you can add a fourth and fifth element without overwriting the others:

```
cars[3] = "Mazda";
cars[4] = "Tesla";
```

C++ References

Creating References

A reference variable is a "reference" to an existing variable, and it is created with the & operator:

```
string food = "Pizza";  // food variable
string &meal = food;    // reference to food
```

Now, we can use either the variable name food or the reference name meal to refer to the food variable:

Example

```
string food = "Pizza";
string &meal = food;

cout << food << "\n";  // Outputs Pizza
cout << meal << "\n";  // Outputs Pizza
```

C++ Memory Address

Memory Address

In the example from the previous page, the & operator was used to create a reference variable. But it can also be used to get the memory address of a variable; which is the location of where the variable is stored on the computer.

When a variable is created in C++, a memory address is assigned to the variable. And when we assign a value to the variable, it is stored in this memory address.

To access it, use the & operator, and the result will represent where the variable is stored:

Example

```
string food = "Pizza";

cout << &food; // Outputs 0x6dfed4
```

Note: The memory address is in hexadecimal form (0x..). Note that you may not get the same result in your program.

And why is it useful to know the memory address?
References and Pointers (which you will learn about in the next chapter) are important in C++, because they give you the ability to manipulate the data in the computer's memory - which can reduce the code and improve the performance.

These two features are one of the things that make C++ stand out from other programming languages, like Python and Java.

C++ Pointers

Creating Pointers

You learned from the previous chapter, that we can get the memory address of a variable by using the & operator:

Example

```
string food = "Pizza"; // A food variable of type string

cout << food;  // Outputs the value of food (Pizza)
cout << &food; // Outputs the memory address of food (0x6dfed4)
```

A pointer however, is a variable that stores the memory address as its value.

A pointer variable points to a data type (like int or string) of the same type, and is created with the * operator. The address of the variable you're working with is assigned to the pointer:

Example

```
string food = "Pizza";  // A food variable of type string
string* ptr = &food;     // A pointer variable, with the name ptr, that stores the address of food

// Output the value of food (Pizza)
cout << food << "\n";
```

```
// Output the memory address of food (0x6dfed4)
cout << &food << "\n";

// Output the memory address of food with the pointer (0x6dfed4)
cout << ptr << "\n";
```

Example explained

Create a pointer variable with the name ptr, that points to a string variable, by using the asterisk sign * (string* ptr). Note that the type of the pointer has to match the type of the variable you're working with.

Use the & operator to store the memory address of the variable called food, and assign it to the pointer.

Now, ptr holds the value of food's memory address.

Tip: There are three ways to declare pointer variables, but the first way is preferred:

```
string* mystring; // Preferred
string *mystring;
string * mystring;
```

C++ Functions

A function is a block of code which only runs when it is called.

You can pass data, known as parameters, into a function.

Functions are used to perform certain actions, and they are important for reusing code: Define the code once, and use it many times.

Create a Function

C++ provides some pre-defined functions, such as main(), which is used to execute code. But you can also create your own functions to perform certain actions.

To create (often referred to as declare) a function, specify the name of the function, followed by parentheses ():

Syntax

```
void myFunction() {
  // code to be executed
}
```

Example Explained

myFunction() is the name of the function

void means that the function does not have a return value. You will learn more about return values later in the next chapter

inside the function (the body), add code that defines what the function should do

Call a Function

Declared functions are not executed immediately. They are "saved for later use", and will be executed later, when they are called.

To call a function, write the function's name followed by two parentheses () and a semicolon ;

In the following example, myFunction() is used to print a text (the action), when it is called:

Example
Inside main, call myFunction():

```
// Create a function
void myFunction() {
  cout << "I just got executed!";
}

int main() {
  myFunction(); // call the function
  return 0;
}

// Outputs "I just got executed!"
```

A function can be called multiple times:

Example

```
void myFunction() {
  cout << "I just got executed!\n";
}

int main() {
  myFunction();
  myFunction();
  myFunction();
  return 0;
}

// I just got executed!
// I just got executed!
// I just got executed!
```

C++ Function Parameters

Parameters and Arguments

Information can be passed to functions as a parameter. Parameters act as variables inside the function.

Parameters are specified after the function name, inside the parentheses. You can add as many parameters as you want, just separate them with a comma:

Syntax

```
void functionName(parameter1, parameter2, parameter3) {
  // code to be executed
}
```

The following example has a function that takes a string called fname as parameter. When the function is called, we pass along a first name, which is used inside the function to print the full name:

Example

```
void myFunction(string fname) {
  cout << fname << " Refsnes\n";
}

int main() {
  myFunction("Liam");
  myFunction("Jenny");
  myFunction("Anja");
  return 0;
}

// Liam Refsnes
// Jenny Refsnes
// Anja Refsnes
```

C++ Default Parameters

Default Parameter Value

You can also use a default parameter value, by using the equals sign (=).

If we call the function without an argument, it uses the default value ("Norway"):

Example

```
void myFunction(string country = "Norway") {
  cout << country << "\n";
}

int main() {
  myFunction("Sweden");
  myFunction("India");
  myFunction();
  myFunction("USA");
  return 0;
}

// Sweden
// India
// Norway
// USA
```

C++ Multiple Parameters

Multiple Parameters

Inside the function, you can add as many parameters as you want:

Example

```
void myFunction(string fname, int age) {
  cout << fname << " Refsnes. " << age << " years old. \n";
}

int main() {
  myFunction("Liam", 3);
  myFunction("Jenny", 14);
  myFunction("Anja", 30);
  return 0;
}

// Liam Refsnes. 3 years old.
// Jenny Refsnes. 14 years old.
// Anja Refsnes. 30 years old.
```

Note that when you are working with multiple parameters, the function call must have the same number of arguments as there are parameters, and the arguments must be passed in the same order.

C++ The Return Keyword

Return Values

The void keyword, used in the previous examples, indicates that the function should not return a value. If you want the function to return a value, you can use a data type (such as int, string, etc.) instead of void, and use the return keyword inside the function:

Example

```
int myFunction(int x) {
  return 5 + x;
}

int main() {
  cout << myFunction(3);
  return 0;
}

// Outputs 8 (5 + 3)
```

This example returns the sum of a function with two parameters:

Example

```
int myFunction(int x, int y) {
  return x + y;
}
```

```
int main() {
  cout << myFunction(5, 3);
  return 0;
}

// Outputs 8 (5 + 3)
```
You can also store the result in a variable:

Example
```
int myFunction(int x, int y) {
  return x + y;
}

int main() {
  int z = myFunction(5, 3);
  cout << z;
  return 0;
}
// Outputs 8 (5 + 3)
```

C++ Functions - Pass By Reference

Pass By Reference

In the examples from the previous page, we used normal variables when we passed parameters to a function. You can also pass a reference to the function. This can be useful when you need to change the value of the arguments:

Example
```
void swapNums(int &x, int &y) {
  int z = x;
  x = y;
  y = z;
}

int main() {
  int firstNum = 10;
  int secondNum = 20;

  cout << "Before swap: " << "\n";
  cout << firstNum << secondNum << "\n";

   // Call the function, which will change the values of firstNum and secondNum
  swapNums(firstNum, secondNum);

  cout << "After swap: " << "\n";
  cout << firstNum << secondNum << "\n";
```

```
  return 0;
}
```

C++ Function Overloading

Function Overloading
With function overloading, multiple functions can have the same name with different parameters:

Example
```
int myFunction(int x)
float myFunction(float x)
double myFunction(double x, double y)
```
Consider the following example, which have two functions that add numbers of different type:

Example
```
int plusFuncInt(int x, int y) {
  return x + y;
}

double plusFuncDouble(double x, double y) {
  return x + y;
}

int main() {
  int myNum1 = plusFuncInt(8, 5);
  double myNum2 = plusFuncDouble(4.3, 6.26);
  cout << "Int: " << myNum1 << "\n";
  cout << "Double: " << myNum2;
  return 0;
}
```
Instead of defining two functions that should do the same thing, it is better to overload one.

In the example below, we overload the plusFunc function to work for both int and double:

Example
```
int plusFunc(int x, int y) {
  return x + y;
}

double plusFunc(double x, double y) {
  return x + y;
}

int main() {
  int myNum1 = plusFunc(8, 5);
  double myNum2 = plusFunc(4.3, 6.26);
  cout << "Int: " << myNum1 << "\n";
```

```
  cout << "Double: " << myNum2;
  return 0;
}
```

Note: Multiple functions can have the same name as long as the number and/or type of parameters are different.

Program to find the ascii value of a character entered from the keyboard

```
#include<iostream>
using namespace std;
int main()
{
      char c;
      cout<<"Enter the character:    ";
      cin >>c;

      cout<<"The ascii value of c is "<<c<<"\t"<<int(c);
}
```