		otherwise, it returns x		

EX:C++ program to Calculate $\sqrt{x^0} + \sqrt{x^2} - \sqrt{x^4} + \sqrt{x^8} - \sqrt{x^{16}}$ ... $(+/-)\sqrt{x^n}$series.

```
#include <iostream>
#include <cmath>
using namespace std;
int main () {
        int n,s=1;
        double x,sum=1,;
        cout<< "enter the values of X ";
        cin>>x;
        cout<<"\n enter the number of quantities ";
        cin>>n;
        for (int i=1;i<n;i++)
            {x=pow(x,2);
             sum+=(s*sqrt(x));
             s*=-1;}
        cout<<"\n the result is: "<<sum;
        return 1;}
```

EX: C++ program to Calculate series $\sqrt{x^0} / \sqrt{y^1} + \sqrt{x^2} / \sqrt{y^3} - \sqrt{x^3} / \sqrt{y^4} + \sqrt{x^4} / \sqrt{y^5} - \sqrt{x^5} / \sqrt{y^6}$ ... $(+/-)\sqrt{x^n} / \sqrt{y^{n+1}}$

```
#include <iostream>
#include <cmath>
using namespace std;
int main () {
        int n,s=1;
        double x,y,sum=0,;
        cout<< "enter the values of X and Y ";
        cin>>x>>y;
        cout<<"\n enter the number of quantities ";
        cin>>n;
        for (int i=0;i<n;i++)
            {sum+=(s*sqrt(pow(x,i))/sqrt(pow(y,i+1)));
             s*=-1;}
        cout<<"\n the result is: "<<sum;
```

```
        return 1;}
```

EX: C++ program to compute the number of upper letters in any string and convert them into lower ones.

```
    #include <iostream>
#include <string>
#include <cctype>

using namespace std;
int main () {
        string st1;
        int c=0;
        cout<<"enter your string: ";
        cin>>st1;
        int len=st1.length();
        for(int i=0;i<len;i++)
           if(isupper(st1[i]))
              {c++;
              st1[i]=tolower(st1[i]);}
        cout<< "\n the number of upper letter is  "<<c;
        cout<< "\n the new string is "<<st1<<"\n";
        return 1;}
```

1.9 User-Defined Functions

Using functions in a program greatly enhances its readability because it reduces the complexity of the **main** function. Also, once you write and properly debug a function, you can use it in the program (or different programs) again and again without having to rewrite the same code repeatedly.

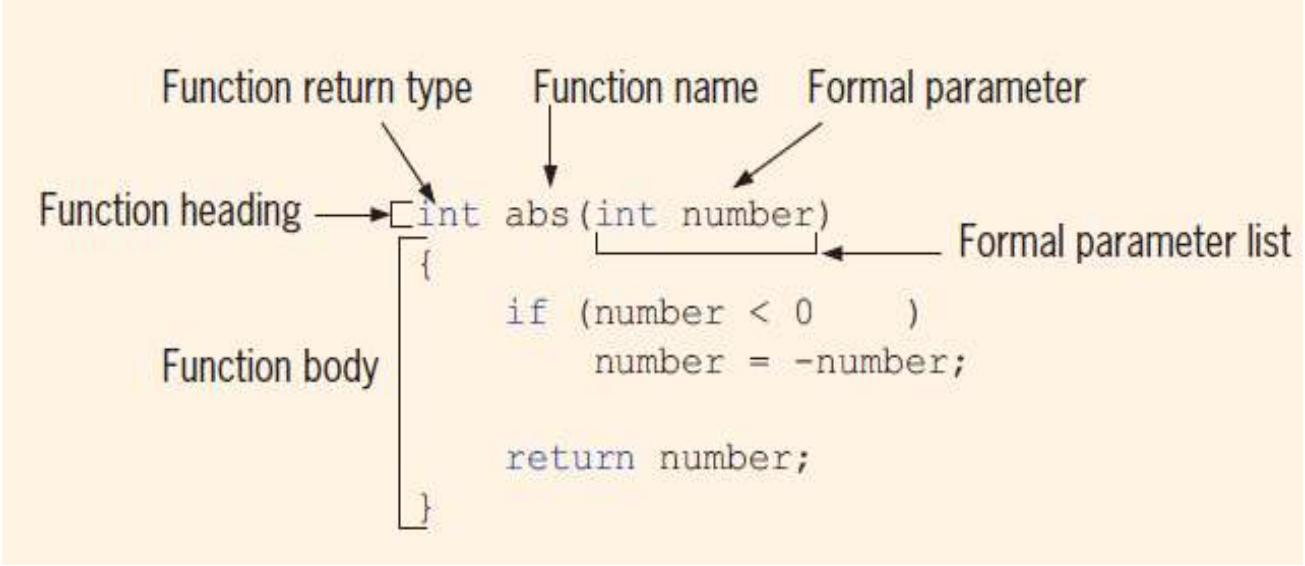
User-defined functions in C++ are classified into two categories:

• **Value-returning functions**—functions that have a return type. These functions return a value of a specific data type using the **return** statement, which we will explain shortly.

• **Void functions**—functions that do not have a return type. These functions do not use a **return** statement to return a value.

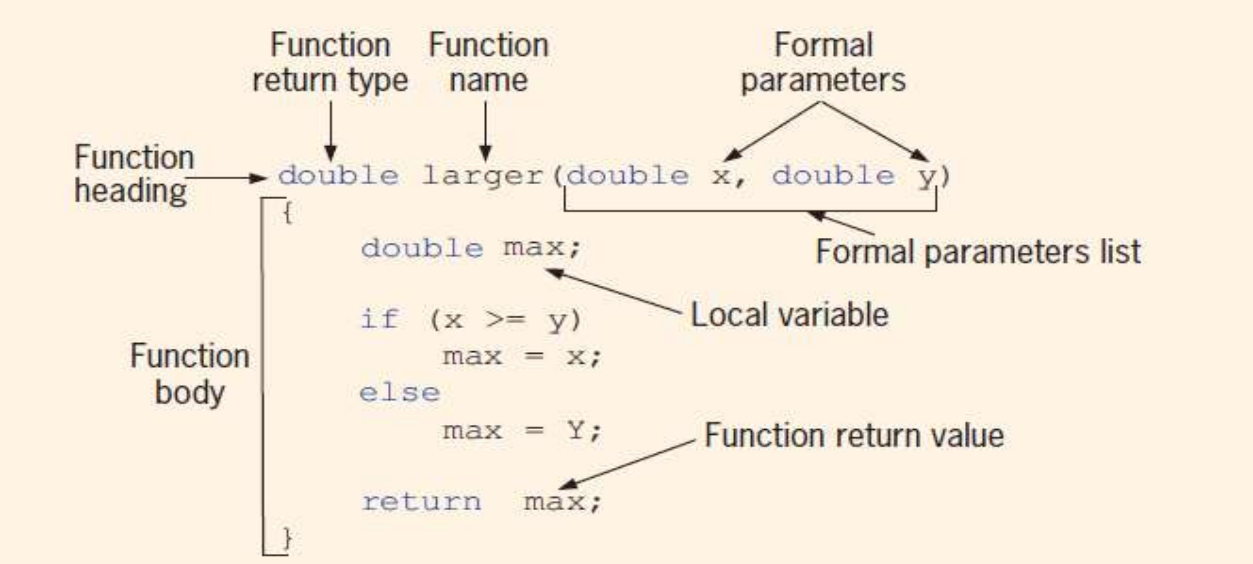
The syntax of a value-returning function is:

```
functionType functionName(dataType identifier, dataType identifier, ...)
{
Statements
return expr;
}
```

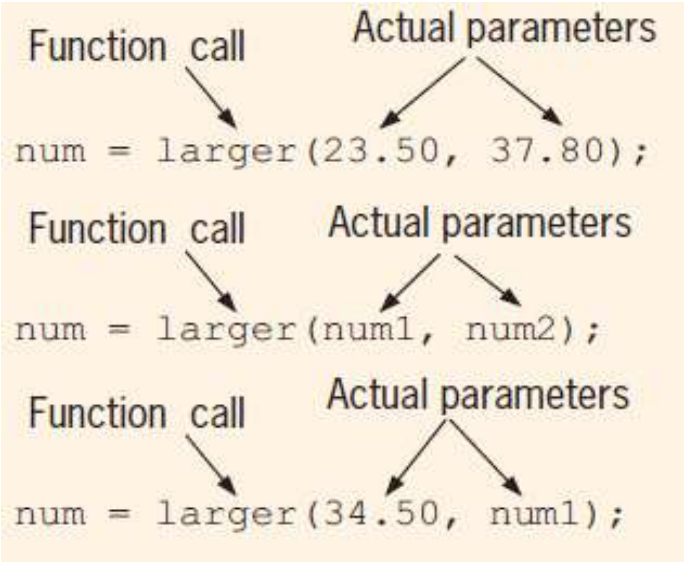


The syntax to call a value-returning function is:

functionName(actual parameter list)
cout<<abs(-500);



Suppose that num, num1, and num2 are double variables. Also suppose that num1 = 45.75 and num2 = 35.50. Figure 6-3 shows various calls to the function larger.



EX: c++ program to find larger among three numbers using previous function larger.

```
#include <iostream>
using namespace std;
double larger(double x, double y)
{
        if (x >= y)
                return x;
        else
```

```
                return y;
}
double compareThree(double x, double y, double z)
{
   return larger(x, larger(y, z));
}
int main () {
double num1,num2,num3;
cout<<"enter three numbers: ";
cin>>num1>>num2>>num3;
cout<<"\n larger number is:"<<compareThree(num1,num2,num3);
return 1;}
```

OR:

```
#include <iostream>
using namespace std;
double larger(double , double ); //function prototype
double compareThree(double , double , double );  //function prototype
int main () {
double num1,num2,num3;

cout<<"enter three numbers: ";
cin>>num1>>num2>>num3;
cout<<"\n larger number is:"<<compareThree(num1,num2,num3);
return 1;}
double larger(double x, double y)
{
        if (x >= y)
                return x;
        else
                return y;
}
double compareThree(double x, double y, double z)
{
   return larger(x, larger(y, z));
}
```

EX: C++ program to reverse any number (3451 converts to 1543)

```
#include <iostream>
using namespace std;
int reversed_number(int num)
{
        int rem,rev=0;
        while(num != 0) {
                rem = num % 10;
                rev = rev * 10 + rem;
                num /= 10;
  }
 return rev;
}
int main() {
  int n;
  cout << "Enter an integer: ";
  cin >> n;
   cout << "Reversed Number = " << reversed_number(n);
  return 0;
}
```

EX: C++ program to compute the number of consonant letters in any string.

```
#include <iostream>
#include<cctype>
#include<string>
using namespace std;
bool vowel(char ch)
{   ch=tolower(ch);
        switch (ch)
        {
                case'e':
                case'u':
                case'i':
                case'o':
                   return true;
                   break;
            default:
               return false;}}
int main() {
                  string st;
                  int len,c=0;
                  cout << "Enter a string: ";
                  cin >> st;
                  len=st.length();
```

```
for(int i=0;i<=len-1;i++)
   if(!vowel(st[i]))
      c++;
cout << "/n number of con-letters is  = " << c<<endl;
return 0;}
```

1.10 Examples of Void functions

EX: C++ program to show the following pattern using two functions one to print space and the second to print stars.

```
#include <iostream>
using namespace std;
void space(int z)
{ for(int i=1;i<=z;i++)
     cout<<" ";}
void stars(int z)
{ for(int i=1;i<=z;i++)
     cout<<"*";}
```

```
int main() {
  int n;
  cout << "Enter number of columns: ";
  cin >> n;
  for(int i=1;i<=n;i++)
     {space(n-i);
      stars(i);
           cout<<endl;}
  return 0;}
```

```
   *
  * *
 * * *
* * * *
```

EX: C++ program to show the following pattern using functions to print space for each line.

```
#include <iostream>
#include <iomanip>
using namespace std;
void line(int start, int end)
{ for(int i=0;i<=end;i++)
     cout<<setw(4)<<start++;}
int main() {
  int n,s=0,e;
  cout << "Enter number of columns: ";
  cin >> n;
  for(int i=1;i<=n;i++)
     {line(s,n);
```

```
 0  1  2  3
 4  5  6  7
 8  9 10 11
12 13 14 15
```

```
      s=s+4;
     cout<<endl;}
 return 0;}
```

EX: C++ program to show the following pattern using functions to print space for each line and the second to print numbers.

```
#include <iostream>
#include <iomanip>
using namespace std;
void space(int z)
{ for(int i=1;i<=z;i++)
     cout<<setw(4)<<" ";}
void line(int start, int end)
{ for(int i=0;i<=end;i++)
     cout<<setw(4)<<start++;}
int main() {
  int n,s,e=0;
  cout << "Enter number of columns: ";
  cin >> n;
  s=n-1;
  for(int i=0;i<n;i++)
     {space(s--);
      line(i+e,e);
      e=e+2;
      cout<<endl;}
  return 0;}
```

```
          0
       1  2  3
    4  5  6  7  8
 9 10 11 12 13 14
```

EX C++ program shows the following pattern using functions to print space for each line.

#include <iostream>
#include <iomanip>
using namespace std;
	int main() {
	  int n;
	  char ch='A';
		A
		B B
		C C C

void space(int z) { for(int i=1;i<=z;i++) cout<<setw(4)<<" ";} void line_C(char c, int num) { for(int i=0;i<num;i++) cout<<setw(4)<<c;}	cout << "Enter number of columns: "; cin >> n; for(int i=1;i<=n;i++) {line_C(ch++,i); cout<<endl;} return 0;}	**D D D D**

1.11 Examples of functions and string

C++ program to swap any string split capital letters from small letters or remove any digits or special characters from a string.

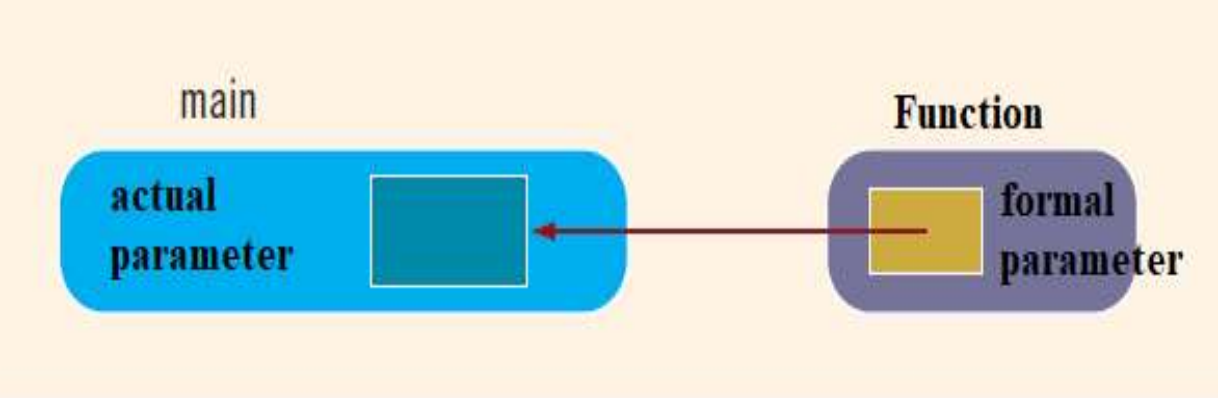
```
#include <iostream>
#include<string>
using namespace std;
string swap(string st)
{   char ch;
        int len=st.length()-1;
        for(int i=0;i<=len/2;i++)
            {ch=st[i];
             st[i]=st[len-i];
             st[len-i]=ch;}
   return st;
}
bool sma_let(char ch)
{
        if(ch>='a' and ch<='z')
          return true;
        else
          return false;
}
bool cap_let(char ch)
{
        if(ch>='A' and ch<='Z')
          return true;
        else
```

```
            return false;
}
void split(string st)
{  string sm="",ca="";
   int len=st.length()-1;
   for(int i=0;i<=len;i++)
          if(sma_let(st[i]))
             sm.append(1, st[i]);
        else if(cap_let(st[i]))
             ca.append(1, st[i]);
   cout<<"small letter string is: "<<sm;
   cout<<"\ncapitall letter string is: "<<ca;}
void remov(string st)
{ string s="ee";
  int len=st.length()-1;
   for(int i=0;i<=len;i++)
      if((sma_let(st[i]))||(cap_let(st[i])))
             s.append(1,st[i]);
   cout<<"new string is: "<<s;}
void showChoices()
{
         cout << "\nEnter--"<<endl;
         cout << "1: To swap a string "<< endl;
         cout << "2: To split capital letters from small letters"<< endl;
         cout << "3: To remove any digits or special characters from a string"<< endl;
         cout << "99: To quit the program." << endl;
}
int main() {
             int choice;
             string st;
                    do
                   {
                   showChoices();
                   cin >> choice;
                   cout << endl;
```

```
switch (choice)
{
case 1:
        cout << "Enter a string: ";
        cin >> st ;
        cout << endl;
   cout<<swap(st);
break;
case 2:
cout << "Enter a string: ";
        cin >> st ;
        cout << endl;
   split(st);
break;
case 3:
cout << "Enter a string: ";
        cin >> st ;
        cout << endl;
   remov(st);
break;
case 99:
    break;
default:
    cout << "Invalid input." << endl;
}
}
while (choice != 99);
return 0;}
```

1.12 Reference variables as parameters

The reference parameter receives the address (memory location) of the actual parameter. Reference parameters can pass one or more values from a function and change the actual parameter's value.



EX: C++ program reads a course score and prints the course grade.

```
#include <iostream>
using namespace std;
void getScore(int& score)
 {

        cout << " Enter course score: ";
        cin >> score;
        cout << endl << "Line 6: Course score is "
        << score << endl;
 }
void printGrade(int cScore)
{
        cout << "Your grade for the course is ";
        if (cScore >= 90)
        cout << "A." << endl;
        else if (cScore >= 80)
        cout << "B." << endl;
        else if(cScore >= 70)
        cout << "C." << endl;
        else if (cScore >= 60)
        cout << "D." << endl;
        else
        cout << "F." << endl;}
int main()
{
```

```
int courseScore;
cout << " Based on the course score, \n"
<< " this program computes the "
<< "course grade." << endl;
getScore(courseScore);
printGrade(courseScore);
return 0;
}
```

EX: C++ program to swap the string.

```
#include <iostream>
#include<string>
using namespace std;
void swap_ch(char& ch1,char& ch2)
 {
        char temp=ch1;
        ch1=ch2;
        ch2=temp; }
void swap_st(string& st)
{   int len=st.length()-1;
   for(int i=0;i<=len/2;i++)
            swap_ch(st[i],st[len-i]);}
int main()
{string st;
 cout<<"Enter the string:";
 cin>>st;
  swap_st(st);
  cout<<"\nthe swapped string is :"<<st<<endl;}
```

1.13 functions and arrays

C++ does not allow functions to return a value of the type array. C++ arrays are always passed by reference.

EX: C++ program to initialize an array, read, print, and find the larger element in an array.

```
#include <iostream>
```

```
using namespace std;
int size = 10;
void initializeArray(int x[],int sizeX)
  {
        for(int i=0;i<=sizeX;i++)
           x[i]=0;
  }



void fillArray(int x[],int sizeX)
  { cout<<"\n enter the elements of array\n";
        for(int i=0;i<=sizeX;i++)
           cin>>x[i];   }
void printArray( int x[],int sizeX)
 {
        for(int i=0;i<=sizeX;i++)
           cout<<"\nx["<<i<<"]="<<x[i]; }
int sumArray( int x[],int sizeX)
 {      int sum=0;
        for(int i=0;i<=sizeX;i++)
                sum+=x[i];
        return sum;   }
int LargestElement( int x[],int sizeX)
 {     int large=x[0];
        for(int i=1;i<=sizeX;i++)
           if(x[i]>large)
             large=x[i];
   return large;}
int main()
{
 int list[10],size=10;
 initializeArray(list,size);
 printArray(list,size);
```

```
 fillArray(list,size);
 printArray(list,size);

 cout<<"\nlarger element in array is: "<<LargestElement(list,size);
 cout<<"\n summation of array:  "<<sumArray(list,size);
}
```

EX: C++ program to swap each string in the array into another array and print them together.

```
#include <iostream>
#include<string>
int size=5;
using namespace std;
void swap_ch(char& ch1,char& ch2)
 {
        char temp=ch1;
        ch1=ch2;
        ch2=temp;
 }
void swap_st(string& st)
{   int len=st.length()-1;
        for(int i=0;i<=len/2;i++)
           swap_ch(st[i],st[len-i]);
}
void swap_arr(string st_ar[], string sw_st[],int sizeX)
{   cout<<"moon";
   string st;
        for(int i=0;i<sizeX;i++)
         {st=st_ar[i];
          swap_st(st);
          sw_st[i]=st;          }  }
```