Functions

A function is a block of code that performs a specific task.

Why are functions used?

If some functionality is performed at multiple places in software, then rather than writing the same code, again and again, we create a function and call it everywhere. This helps reduce code redundancy.
Functions make maintenance of code easy as we have to change at one place if we make future changes to the functionality.
Functions make the code more readable and easy to understand.

The syntax for function declaration is

```
return-type function_name (parameter 1, parameter …… parameter n){  //function_body
}
```

return-type
The return type of a function is the data type of the variable that that function returns.
For eg. if we write a function that adds 2 integers and returns their sum then the return type of this function will be 'int' as we will returning sum that is an integer value.
When a function does not return any value, in that case the return type of the function is 'void'.

function_name
It is the unique name of that function.
It is always recommended to declare a function before it is used.

Parameters
A function can take some parameters as inputs. These parameters are specified along with their data types.
For eg. if we are writing a function to add 2 integers, the parameters would be passed like –

```
int add (int num1, int num2)
```

Main function

The main function is a special function as the computer starts running the code from the beginning of the main function. Main function serves as the entry point for the program.

Examples –

Ques1. Write a program to add 2 numbers using functions.

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

int add(int num1, int num2){
 int sum = num1 + num2;
 return sum;
}

int main()
{
 int a,b;
 cin>>a>>b;

 cout<<add(a,b)<<endl;

 return 0;
}
```

Ques2. Write a program to print a given number using functions.

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

```
void display(int a){
 cout<<a<<endl;
 return;
}

int main()
{
 int a;
 cin>>a;

 print(a);

 return 0;
}
```

C++ Function Without Prototype:

cpp Copy code

```cpp
#include <iostream>

// Function definition
void sayHello() {
    std::cout << "Hello, world!" << std::endl;
}

int main() {
    // Calling the function
    sayHello();
    return 0;
}
```

In this example, the function `sayHello()` is defined before `main()`, so there's no need for a separate prototype declaration.

C++ Function With Prototype:

cpp Copy code

```cpp
#include <iostream>

// Function prototype
void sayHello();

int main() {
    // Calling the function
    sayHello();
    return 0;
}

// Function definition
void sayHello() {
    std::cout << "Hello, world!" << std::endl;
}
```