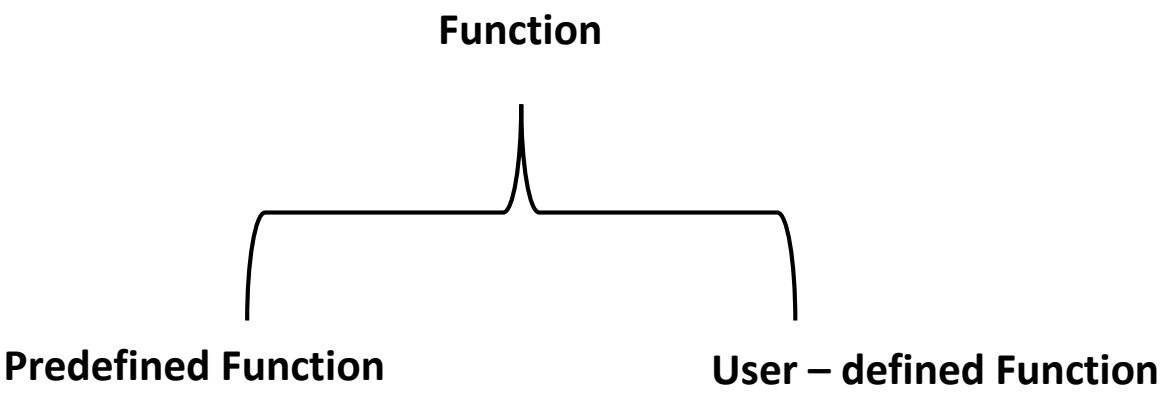
In this lecture you will learn:

- **Function**

Functions are like building blocks. They let you divide complicated programs into manageable pieces. They have other advantages, too:

- While working on one function, you can focus on just that part of the program and construct it, debug it, and perfect it.
- Different people can work on different functions simultaneously.
- If a function is needed in more than one place in a program or in different programs, you can write it once and use it many times.
- Using functions greatly enhances the program's readability because it reduces the complexity of the function main.

There is two type of function:



Predefined Functions

Function	Header File	Purpose	Parameter(s) Type	Result
abs(x)	<cmath>	Returns the absolute value of its argument: abs(-7) = 7	int (double)	int (double)
ceil(x)	<cmath>	Returns the smallest whole number that is not less than x: ceil(56.34) = 57.0	double	double
cos(x)	<cmath>	Returns the cosine of angle: x: cos(0.0) = 1.0	double (radians)	double
exp(x)	<cmath>	Returns e^x, where e = 2.718: exp(1.0) = 2.71828	double	double
fabs(x)	<cmath>	Returns the absolute value of its argument: fabs(-5.67) = 5.67	double	double
floor(x)	<cmath>	Returns the largest whole number that is not greater than x:floor(45.67) = 45.00	double	double
islower(x)	<cctype>	Returns true if x is a lowercase letter; otherwise, it returns false; islower('h') is true	int	int
isupper(x)	<cctype>	Returns true if x is a uppercase letter; otherwise, it returns false; isupper('K') is true	int	int
pow(x, y)	<cmath>	Returns x^y; if x is negative, y must be a whole number: pow(0.16, 0.5) = 0.4	double	double
sqrt(x)	<cmath>	Returns the nonnegative square root of x; x must be nonnegative: sqrt(4.0) = 2.0	double	double
tolower(x)	<cctype>	Returns the lowercase value of x if x is uppercase; otherwise, it returns x	int	int

EXAMPLE 1: The Square Root Function sqrt ( )

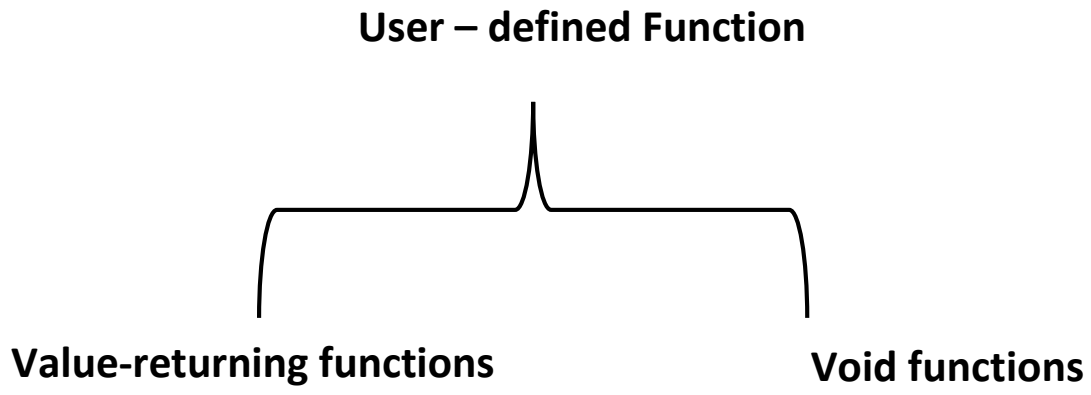
```
#include <iostream>
#include <math.h>
// Test-driver for the sqrt function:
main0
{
for (int i = 0; i < 6; i++)
Cout << "\t" << i << "\t" << sqrt(i) << endl;
}
```

Home work: by using predefined function prove the following Trigonometry:

sin2x = 2 sinxcosx try x = 0 to 2 incremented by 0.2

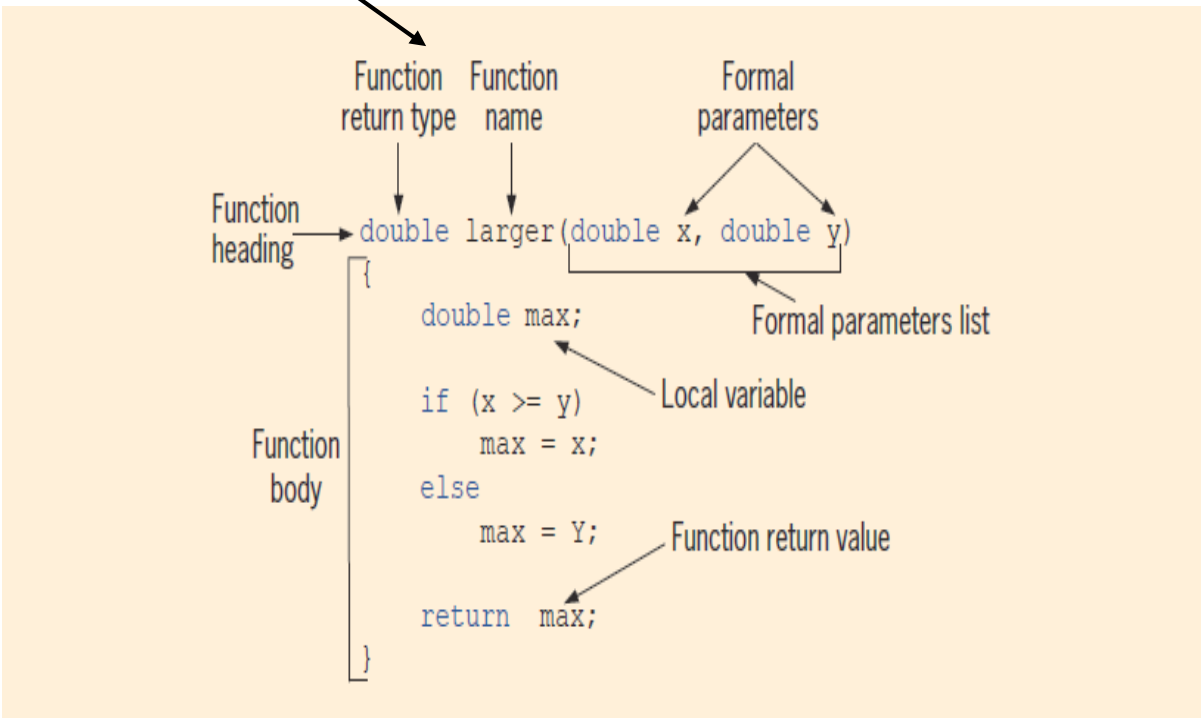
User-Defined Functions

Because C++ does not provide every function that you will ever need and designers cannot possibly know a user's specific needs, you must learn to write your own functions.

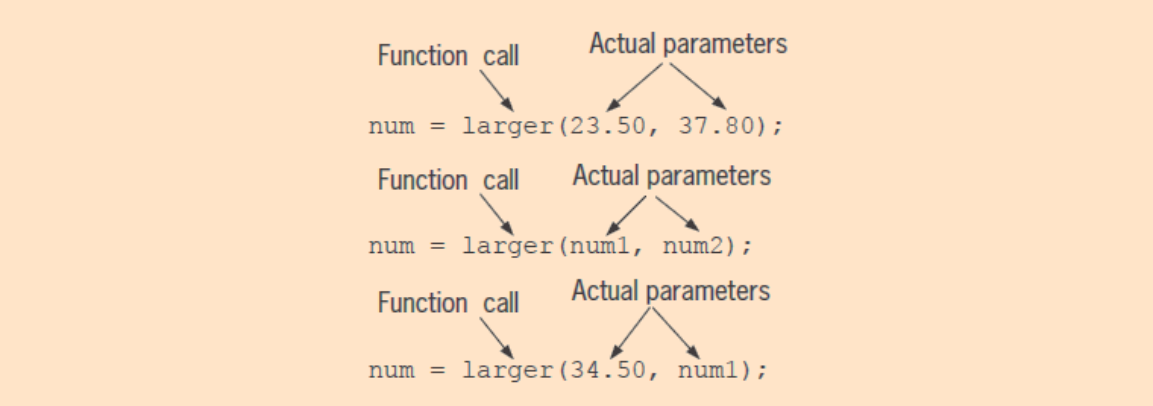


Terminology:

Inside function



Inside main



Example 2: cube ( ) Function

Here is a simple example of a user-defined function:

```
// Returns the cube of the given integer:

int cube(int x)

{

return x*x*x;

}
```

The function returns the cube of the integer passed to it. Thus cube (2) would return 8.

Here is a complete program, consisting of our `cube` function followed by a test driver:

```
// Returns the cube of the given integer:

int cube(int x)

{

return x*x*x;

}

// Test driver for the cube function:

Main( )

{

int n = 1;

while (n != 0) {

cin >> n;

cout << cube(n) << endl;

}

}
```

Example 3: The factorial ( ) Function

The factorial of a positive integer y2 is the number n! Obtained by multiplying n by all the positive integers less than n:

n! = (n) (n - 1) ... (3) (2) (l)

For example, 5! = (5) (4) (3) (2) (l) = 120.

Here is an implementation of the factorial function:

```
int factorial(int n)
{
if (n < 0) return 0;
int f = 1;
while (n > 1)
f *= n--;
return f;
}
```

Example 4: The Permutation Function

A permutation is an arrangement of elements taken from a finite set. The permutation function **P(n,k)** gives the number of different permutations of any k items taken from a set of n items. One way to compute this function is by the formula

$$\boldsymbol{p(n,k) = \frac{n!}{(n-k)!}}$$

For example:

$$\boldsymbol{p(5,2) = \frac{51}{(5-2)!} = \frac{5!}{3!} = \frac{120}{6} = 20}$$

```
// Returns P(n,k), the number of permutations of k from n:
int perm(int n, int k)
{
if (n < 0 II k < 0 II k > n) return 0;
return factorial(n)/factorial(n-k);
}
```

Here is a test driver for the `perm ( )` *function:*

```
int perm(int,int);
main ()
{
for (int i = -1; i < 8; i++) {
for (int j = -1; j <= i+l; j++)
Cout << " "<< perm(i,j);
Cout << endl;
}
}
```

References:

[1] Deitel, P. & Deitel, R., "C++ How to Program", Eighth Edition, Pearson Education Inc., 2012.

[2] Stefan B., " C++ Primer Plus ", Sixth Edition, Pearson Education Inc., 2012.