# SPM 2102
# PROGRAMMING LANGUAGE 1

## Functions (C++ Programming)

By

NORAH MD NOOR

# Function - OutLine

- Definition
- Pre-defined functions
- User-defined functions
- Local vs Global variables
- Arguments/parameters

# Function – Intro…

- **What is function?**
  - Function is a self-contained routines within a larger program that carry out specific tasks.
  - One entity in programming which have a set of command to carry out specific task
  - Sub-routine
  - We can call these function every time we need (in programming sequences) – reusable
  - Sub-routine sub to sub-routine

# Function – Intro…

- **Function definition**
  - Every sub-routine, module
  - Main routine – main function () (Starting of routine)

# Function – Intro…

- **Why we need function?**
  - Organize code in program
  - More manageable?
  - Code are easier to maintain?

# Function – Intro…

- When we need function?
  - When you need to repeat the same process over and over in a program.
  - The function can be called many times but appears in the code once.

# Function Syntax

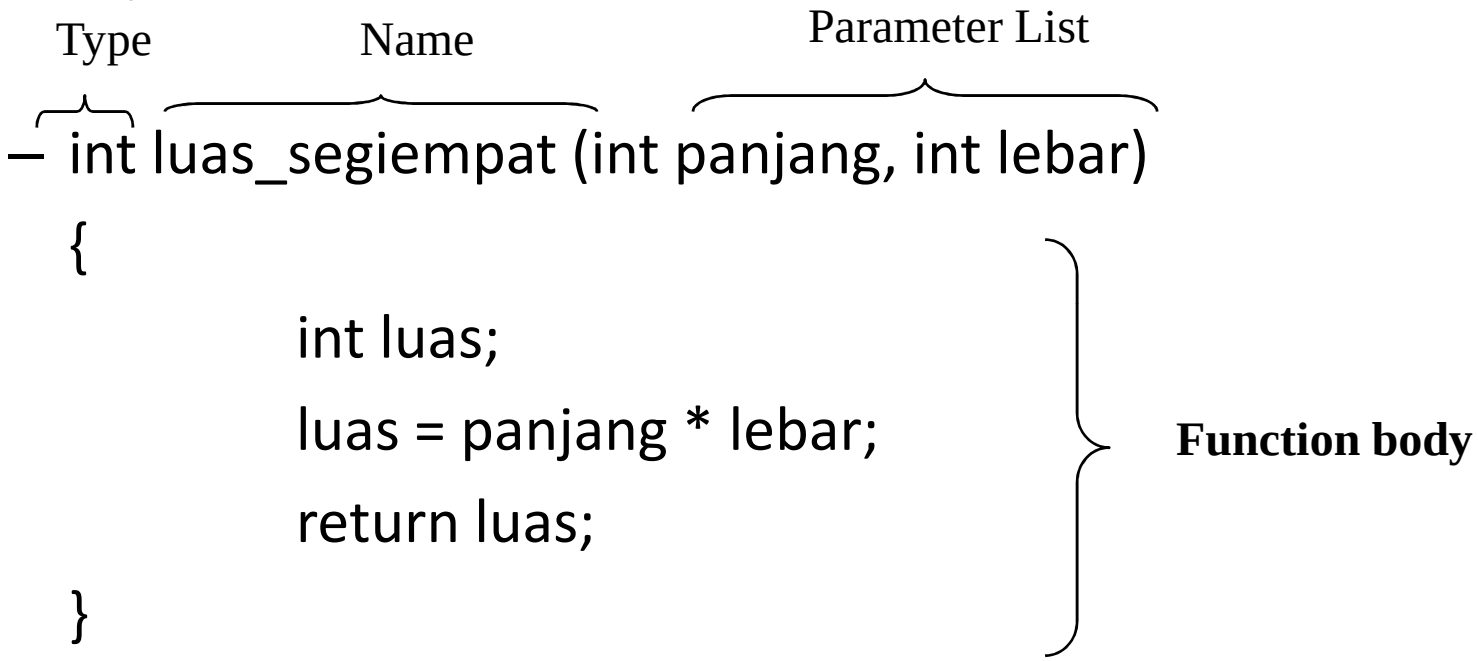
- Function define using following syntax :

```
Function_type function_name (Parameter List)
{
    …function body
}
```

- *Eg : float total_number(void)*

# Function Syntax

- Example:

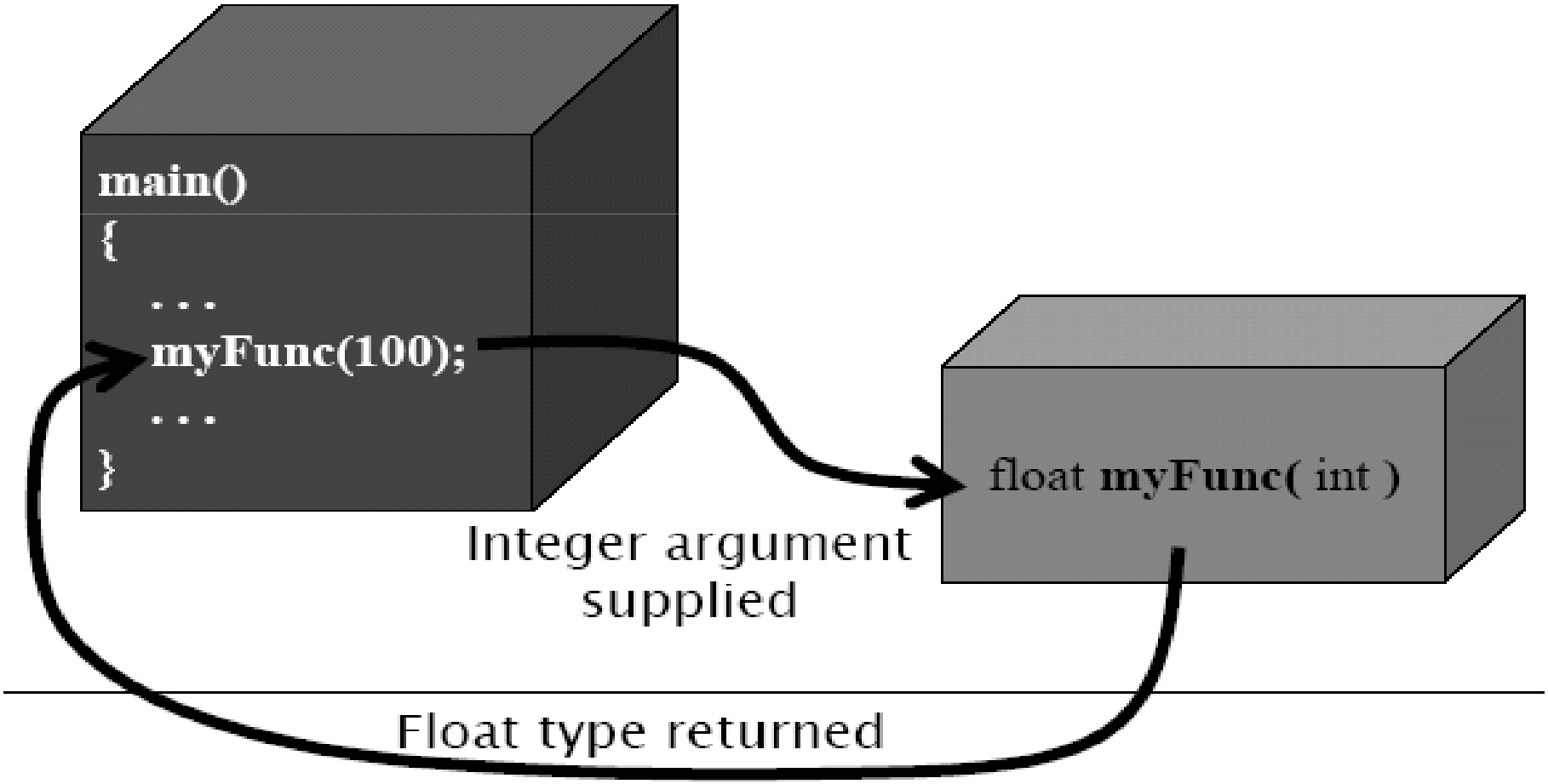


# Function - Elements

- **Three main elements in function :**
  - Define the function
  - Called the function
  - Prototype of Function / Declaration

# Function – Elements

- Function Call



# Function - Categories

**C++ functions can be divided into 2 categories** :

- Pre-defined (standard function) / library function
  - Which is the definitions have been written and it is ready to be used.
  - User needs to include pre-defined header file (*i.e. math.h, time.h*)

- User-defined
  - Function that been created by the user.
  - This functions need to be declared and defined by the user.

# Function - PreDefined

- Pre-Defined
  - FunctionName ( value )

- For example:
  - *math.h*
    sqrt( 4)
    floor (4.3)

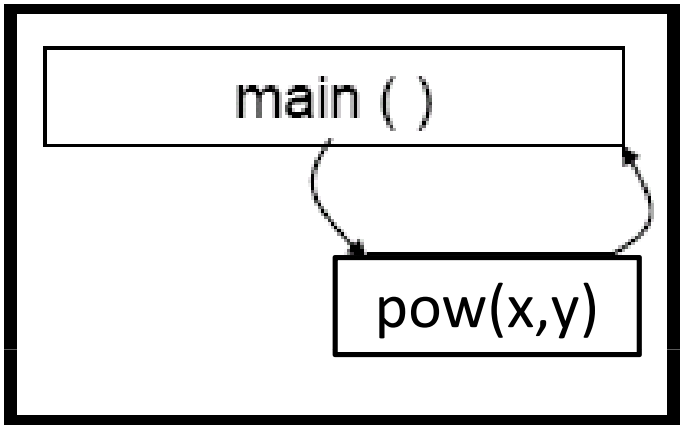
  - *time.h*
    localtime( )

# Function - PreDefined

- The Standard Math Library
  - fabs : absolute value of floating point number
  - floor : largest integral value not greater than x
  - ceil : smallest integral value not less than x
  - fmod : floating-point remainder function
  - log : logaritma
  - pow : Raise a number by a power.
  - sin : The sine of an integer.
  - sqrt : Square root of a number.
  - tan : Tangent.
  - tanh : Hyperbolic tangent.
  - Etc…

# Example of Function pre-defined

```
#include<conio.h>
#include<iostream.h>
#include<math.h>
void main()
{
double no1 = 3 , no2 = 4;
cout <<no1<<"indeks" <<no2<<" = "<<pow(no1,no2);
getch();
}
```



# Function – User Define

- Function Prototype Examples

*long **FindArea**(long **length**, long **width**);*

*// returns long, has two parameters*

*void **PrintMessage**(int **messageNumber**);*

*// returns void, has one parameter*

*int **GetChoice**();*

*// returns int, has no parameters*

# Function – User Define

- **Function Definition Examples**

```
long FindArea(long len, long w)
{
return len * w;
}
void PrintMessage(int whichMsg)
{
if (whichMsg == 0)
cout << "Hello.\n";
if (whichMsg == 1)
cout << "Goodbye.\n";
if (whichMsg > 2)
cout << "I'm confused.\n";
```

```
#include <math.h>    //declare as math function
#include <iostream.h>
#include <conio.h>
int indeks_2( int ); // Function prototype

void main()
{
int y;
for ( int x = 1; x <= 10 ; x++ )
cout << indeks_2( x )<<"\n";// Function call
cin>>y;
cout<<y;
}

int indeks_2( int x ) // Function definition
{ return x + x; }
```

# Local vs Global Variable

- **Local variables**
  - Variables declared in a function (or any functions) are local to the function.
  - Variables declared in **main()** are local to **main()**

- **Global variables**
  - Variables declared before/outside **main()** are global variables.
  - They are known throughout the program.

# Local vs Global Variable

- **Example:**

```
Global variables
long int x, y, z;
void main()
{
. . .
}
int myFunction( void )
{
long int x, y, z; //Local variables
}
```

# Argument OR Parameters

- Arguments of a function are called formal parameters
- Parameters used when calling a function are called actual parameters
- Actual parameters must match the formal
- Parameters in number, type & order

# Argument OR Parameters

```
#include<iostream.h>
#include<conio.h>
int luas_segiempat (int, int);
void main()
{
cout<<" Luas SegiEmpat ";
cout<<luas_segiempat (4,8);
getch();
}
int luas_segiempat (int panjang, int lebar)
{
int luas;
luas = panjang*lebar;
return luas;
}
```

# Argument OR Parameters

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

int luas_segiempat(int, int);

void main(){
 cout<<" luas segiempat lebar 4, panjang 8 adalah: ";
 cout<<luas_segiempat(4, 8);

 getch();
}

int luas_segiempat (int panjang, int lebar)
   {
      int luas;
      luas = panjang * lebar;
      return luas;
   }
```

Actual parameter

Formal parameter

# That's all for today