SLIIT

Discover Your Future

IT1050- Object Oriented Concepts

Functions in C++

Agenda

- Standard functions in C++
- User defined functions
  - Pass by value
  - Pass by reference

Standard Functions in C++

- <cmath>
- sqrt(x)
- log(x)
- log10(x)
- pow(x, y)
- exp(x)

- <iomanip>
- setw(n)
- setprecision(n)
- setiosflags()

- <cstring>
- strcpy(string1,string2)
- strcmp(string1,string2)
- strcat(string1,string2)
- strlen(string1)

Implementing Functions

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

void printMessage();

int main()
{
   printMessage();
   return 0;
} //main
```

Called Function

Function Prototype

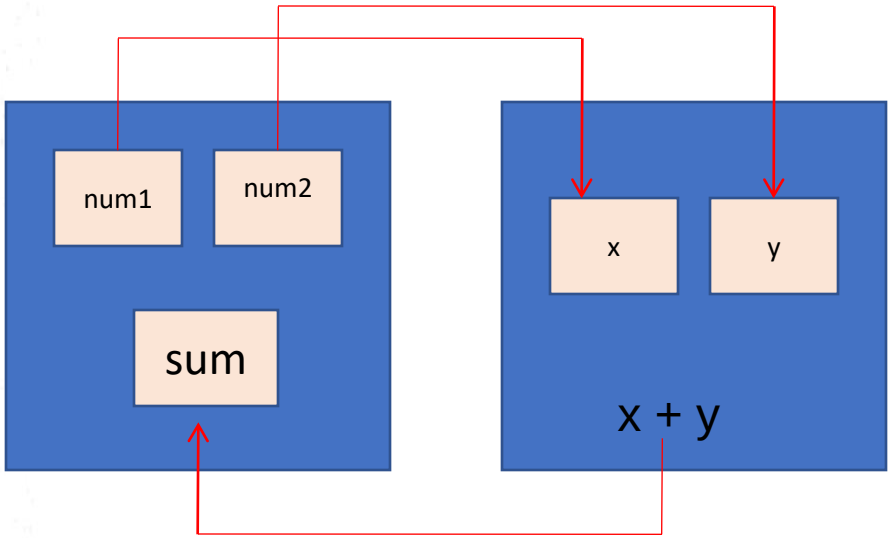
Calling Function

```
void printMessage()
{
   cout << "**********" << endl;
   cout << "* Hello  *" << endl;
   cout << "**********" << endl;
} //printMessage
```

Example-01

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

int findSum(int x, int y);

int main()
{
    int num1, num2,sum;
    cout<<"Input two numbers :";
    cin >> num1 >> num2;
    sum = findSum(num1, num2 );
     cout<<"Sum is : "<<sum <<endl;
    return 0;
} //main
```

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



Example-02

- Define the procedural abstraction for a function which calculates elapsed time in hours for a trip, given distance in miles, and mph.

elapsedTime(Distance, MPH) -> Time

- The elapsedTime function takes an integer for the distance and a float mph and produces the elapsed time for the trip.

Implementing a function

- Function Heading

```
return_type function_name(parameter_list)
```

```
Eg :  float  CalculateAvg ( int num1, int num2 )
```

- The Parameter List

  -Comma separated list of parameter declarations

  -Each parameter is declared separately

- Return statement

  -is what the function uses to specify the value to be returned

```
return expression;
```

```
Eg : return avg;
Eg : return ( num1+num2) /2.0f;
```

ElapsedTime() function

```
float elapsedTime(int Distance, float MPH)
{
  if(MPH > 0.0)
       return(Distance/MPH);
  else
       cout << "MPH is not greater than 0!";
  return 0.0;
}
```

Invoking Functions

- Functions can be invoked by using their name followed by the argument list, which is enclosed in parentheses, anywhere a value of the same type as the return type can be used.

```
Eg :  avg = calculateAvg( 10, 20 );
```

- Functions which do not return values (sometimes referred to as procedures or void functions) can be invoked by using the function name and its argument list as a statement in the program. If a function returns a value and is invoked in this manner the value is discarded.

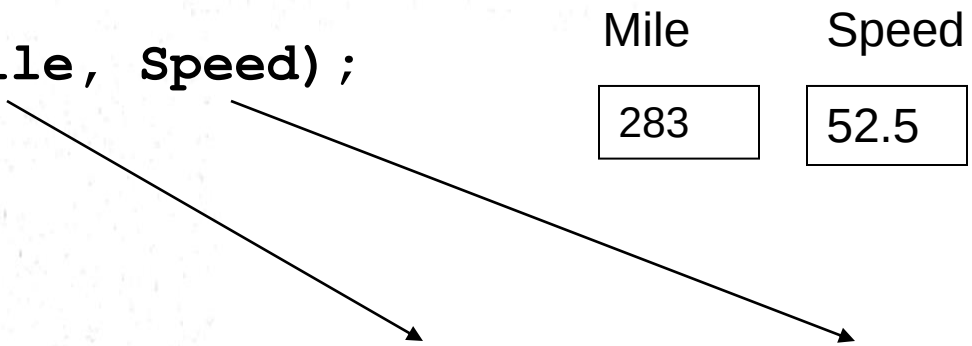
```
Eg :  printMessage();
```

Program using ElapsedTime()

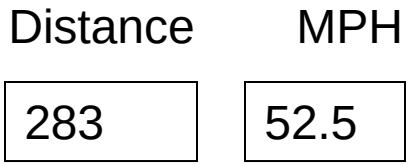
```
#include <iostream>
Using namespace std;
float elapsedTime(int Distance, float MPH);
int main()
{
   int Miles;
   float Speed, Time;
   cout << "Please type the distance in miles:";
   cin >> Miles;
   cout << "Please type the speed as miles per hour:";
   cin >> Speed;
   time = elapsedTime(Miles,Speed);
   cout << "The elapse time for a trip of " << Miles << "miles," <<
   endl << "at a rate of " << Speed << "MPH is" << Time << "hours.";

   return 0;
} //main
```

Invoking ElapsedTime()

```
Time = elapsedTime(Mile, Speed);
```




```
float elapsedTime(int Distance, float MPH)
{
      if(MPH > 0.0)
            return(Distance/MPH);
      else
            cout << "MPH is not greater than 0!"
      return 0.0;
} //elapsedTime
```



Sample output for Example-02

Please type the distance in miles: 283

Please type the speed as miles per hour: 52.5

The elapsed time for a trip of 283 miles,

at a rate of 52.5 MPH is 5.39 hours.

Arguments and Parameters

- The term argument refers to the values passed into the function. The arguments appear in the invocation of the function.

```
Time = elapsedTime(Mile, Speed);
```

- The term parameter refers to the variables declared in the function heading and used by the function to hold the information passed to it.

```
float elapsedTime(int Distance, float MPH)
```

Arguments and Parameters

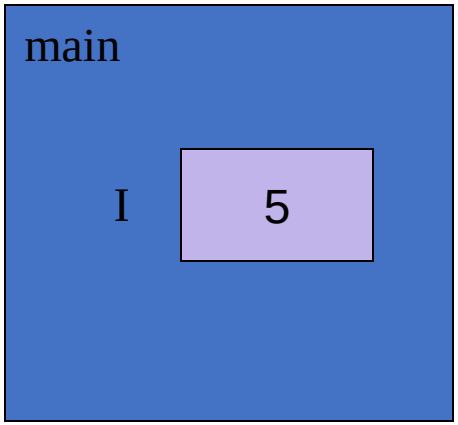
- The first argument matches with the first parameter, the second argument matches the second parameter, and so forth.
- The argument and the parameter do not have to have the same name.
- The argument may be a constant, a variable, or an expression, when using pass by value.
- The parameter must be a variable.

Parameter Passing - Pass by Value

- Pass by value is a method of passing information to a function whereby the parameter receives a copy of the value of the argument.
- Any changes that the function makes to the parameter when pass by value is used are made to the copy and not to the original argument.

Example-03 - Pass by Value

```
void change(int X);


int main()
{  int I=5;
   cout << "First time I is "<< I <<endl;
   change(I);
   cout << "Next time I is "<< I << endl;
} //main


void change(int X)
{  cout << "Entering function X is "<< X;
   X = 7;
   cout << "Leaving function X is "<< X;
} //change
```

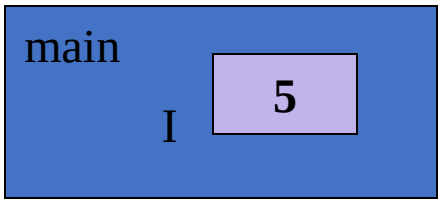


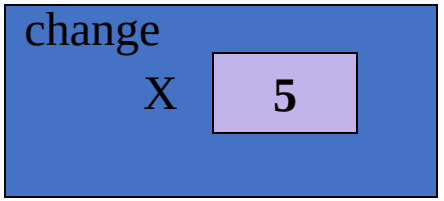
Output

First time I is 5

Example-03

```
void change(int X);

int main()
{   int I=5;
    cout << "First time I is "<< I <<endl;
    change(I);
    cout << "Next time I is "<< I << endl;
} //main

void change(int X)
{   cout << "Entering function X is "<< X;
    X = 7;
    cout << "Leaving function X is "<< X;
} //change
```





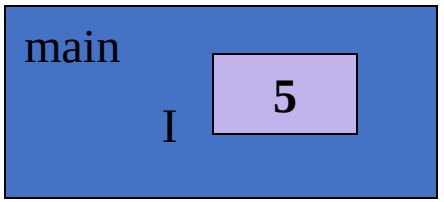
Output

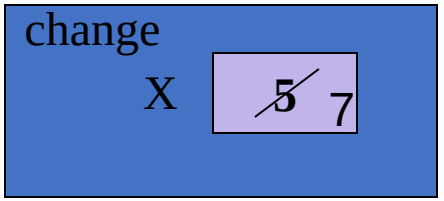
First time I is 5

Entering function X is 5

Example-03

```
void change(int X);

int main()
{   int I=5;
    cout << "First time I is "<< I <<endl;
    change(I);
    cout << "Next time I is "<< I << endl;
} //main


void change(int X)
{   cout << "Entering function X is "<< X;
    X = 7;
    cout << "Leaving function X is "<< X;
} //change
```





Output

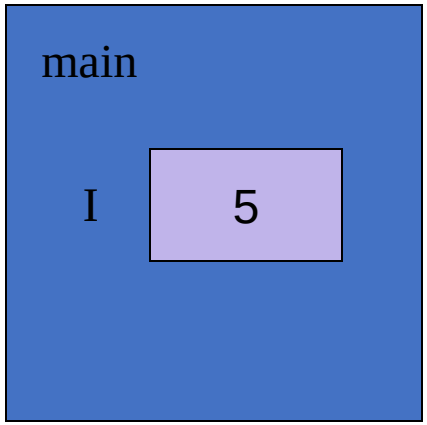
First time I is 5

Entering function X is 5

Leaving function X is 7

Example-03

```
void change(int X);


int main()
{   int I=5;
    cout << "First time I is "<< I <<endl;
    change(I);
    cout << "Next time I is "<< I << endl;
} //main


void change(int X)
{   cout << "Entering function X is "<< X;
    X = 7;
    cout << "Leaving function X is "<< X;
} //change
```



Output

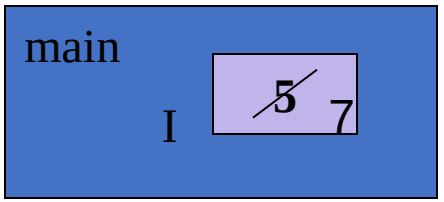
First time I is 5
Entering function X is 5
Leaving function X is 7
Next time I is 5

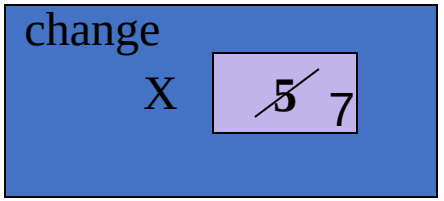
Parameter Passing – Pass By Reference

- Pass by reference or call by reference is the technique of making a variable accessible to a function by passing its address. Because the function has direct access to the argument, through its address, the function may modify the argument.
- In C++, pass by reference is obtained by placing an & immediately following the data type for the parameter.

Example-04 - Pass by Reference

```
void change(int &X);


int main()
{   int I=5;
    cout << "First time I is "<< I <<endl;
    change(I);
    cout << "Next time I is "<< I << endl;
} //main


void change(int &X)
{  cout << "Entering function X is "<< X;
    X = 7;
    cout << "Leaving function X is "<< X;
} //change
```





Output

First time I is 5
Entering function X is 5
Leaving function X is 7
Next time I is 7

When To Use Pass By Reference

- Use pass by reference whenever the function returns a value or multiple values through the parameter list.
- Use pass by reference for two way communication, i.e. the function modifies a parameter.
- Use pass by reference to reduce storage needs when passing large objects.
- Input and Output streams must be passed to a function by reference.

Example-05

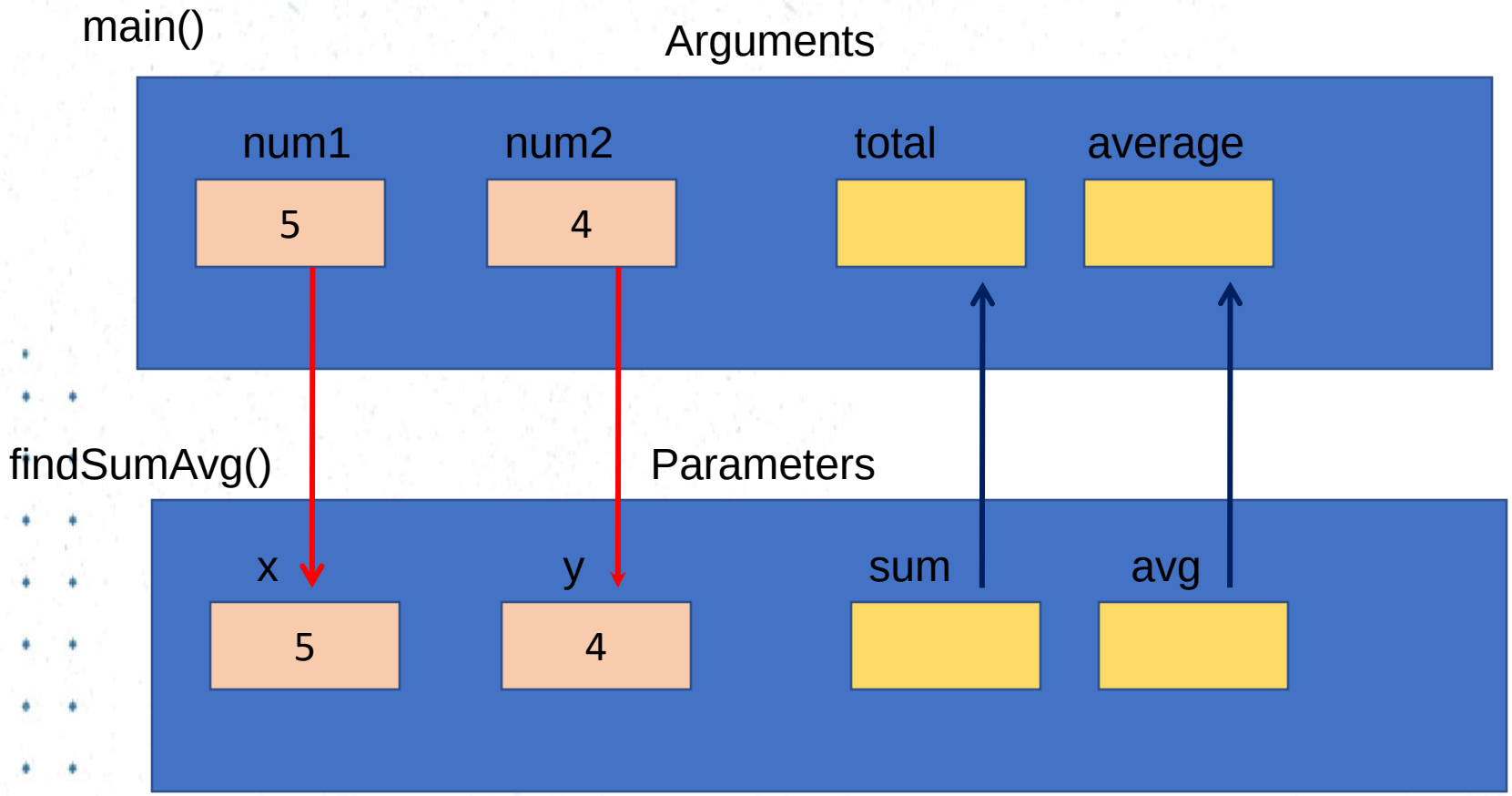
- Write a program to input two numbers from the keyboard and find the sum and average by using a function.

```
void findSumAvg(int x, int y, int &sum, float &avg);
```

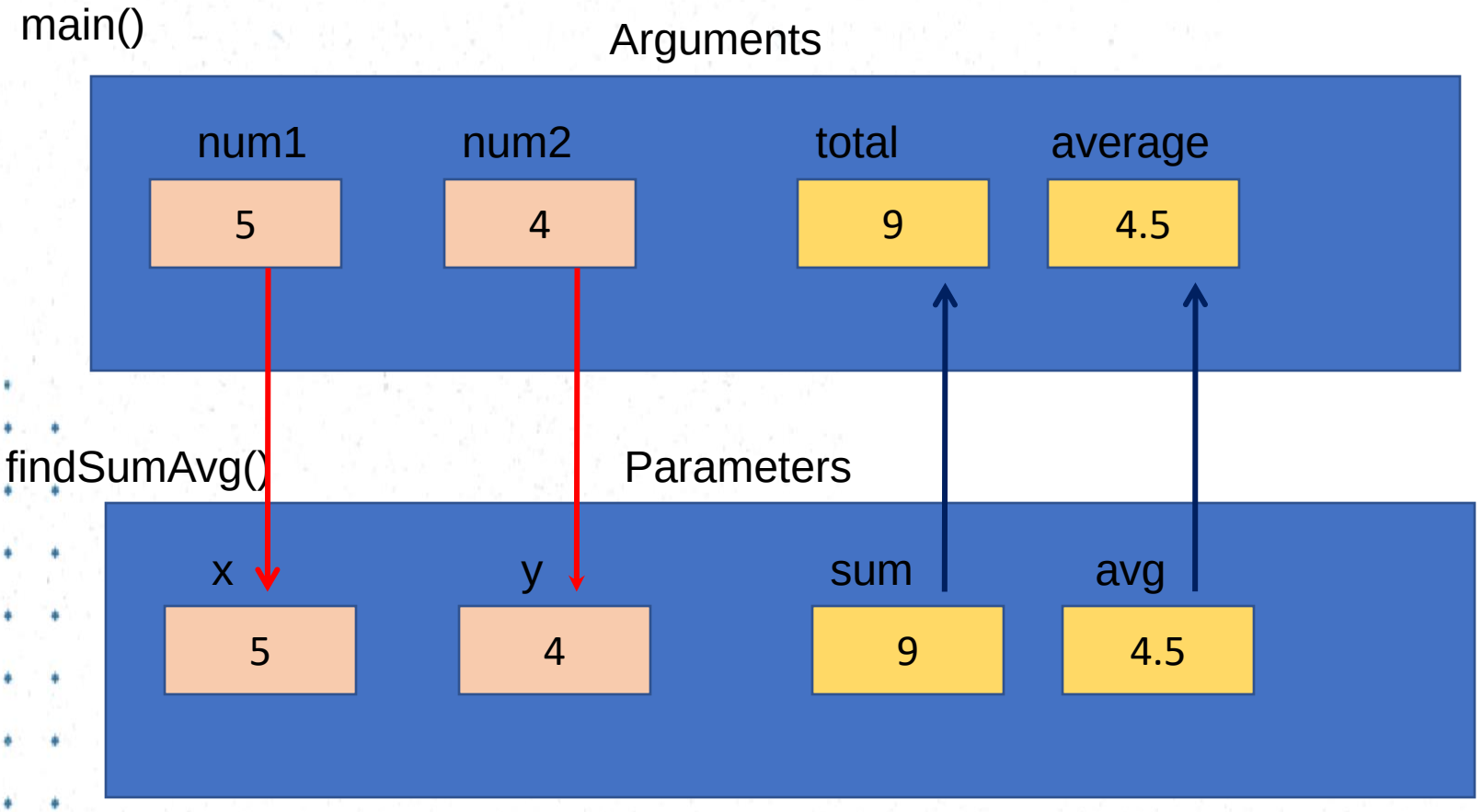
```
void findSumAvg(int x, int y, int &sum, float &avg);

int main()
{
  int num1, num2,total;
  float average;

  cout<<"Input two numbers :";
  cin>>num1 >> num2;

  findSumAvg(num1, num2, total, average);

  cout<<"Sum is : "<< total << endl;
  cout<<"Average is : "<< average <<endl;

  return 0;
}
```

```
void findSumAvg(int x, int y, int &sum, float &avg)
{
   sum = x + y;
   avg = sum / 2.0;
}
```

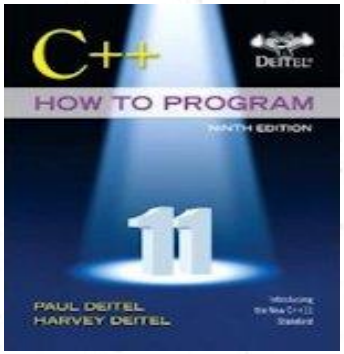
When calling findSumAvg()



After calling findSumAvg()



Reference



Chapter 06

Deitel & Deitel's (2016), C++ How to Program, 9th Edition

Thank you……

anjalie.g@sliit.lk