Lecture (10)
" Programming Essentials in C++ “
Function

Presented by : Dr. Nehal El Azaly,
Dr. Mohamed AbdEl Fattah

Function Parameters and Arguments

- Information can be passed to functions as a parameter. Parameters act as variables inside the function.

```
void functionName(parameter1, parameter2, parameter3) {
  // code to be executed
}
```

Example

```
void myFunction(string fname) {
  cout << fname << " Refsnes\n";
}

int main() {
  myFunction("Liam");
  myFunction("Jenny");
  myFunction("Anja");
  return 0;
}

// Liam Refsnes
// Jenny Refsnes
// Anja Refsnes
```

Default Parameter Value

- You can also use a default parameter value, by using the equals sign (=).

```
void myFunction(string country = "Norway") {
  cout << country << "\n";
}

int main() {
  myFunction("Sweden");
  myFunction("India");
  myFunction();
  myFunction("USA");
  return 0;
}

// Sweden
// India
// Norway
// USA
```

Multiple Parameters

```
void myFunction(string fname, int age) {
  cout << fname << " Refsnes. " << age << " years old. \n";
}

int main() {
  myFunction("Liam", 3);
  myFunction("Jenny", 14);
  myFunction("Anja", 30);
  return 0;
}

// Liam Refsnes. 3 years old.
// Jenny Refsnes. 14 years old.
// Anja Refsnes. 30 years old.
```

Function Return Values

- The void keyword, indicates that the function should not return a value.

- To return a value, you can use a data type (such as int, string, etc.) instead of void, and use the return keyword inside the function

Example of Function Declaration and Definition

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

int main() {
  int z = myFunction(5, 3);
  cout << z;
  return 0;
}
// Outputs 8 (5 + 3)
```

Example of Function Declaration and Definition

main.cpp

```cpp

#include <iostream>
using namespace std;

int myFunction(int x, int y) {
    return x + y;
}

int main() {
    int z = myFunction(5, 3);
    cout << z; // Outputs 8 (5 + 3)
    return 0;
}

```

```
8
```

Pass Array

```
void myFunction(int myNumbers[5]) {
  for (int i = 0; i < 5; i++) {
    cout << myNumbers[i] << "\n";
  }
}

int main() {
  int myNumbers[5] = {10, 20, 30, 40, 50};
  myFunction(myNumbers);
  return 0;
}
```

Example of Function Declaration and Definition

Run Debug Stop Share Save { } Beautify

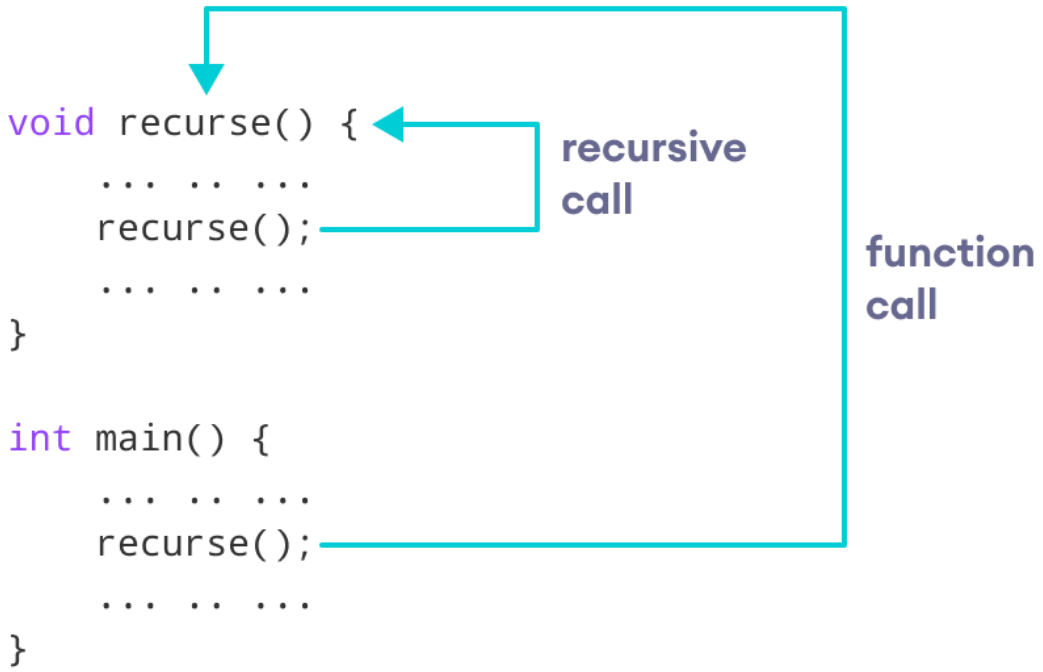
main.cpp

```cpp

#include <iostream>
using namespace std;

void myFunction(int myNumbers[5]) {
    for (int i = 0; i < 5; i++) {
        cout << myNumbers[i] << "\n";
    }
}

int main() {
    int myNumbers[5] = {10, 20, 30, 40, 50};
    myFunction(myNumbers);
    return 0;
}
```

```
10
20
30
40
50
```

Recursion

- Recursion is the technique of making a function call itself.
- This technique provides a way to break complicated problems down into simple problems which are easier to solve



Example

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

int sum(int k) {
  if (k > 0) {
    return k + sum(k - 1);
  } else {
    return 0;
  }
}

int main() {
  int result = sum(5);
  cout << result;
  return 0;
}
```

10 + sum(9)

10 + (9 + sum(8))

10 + (9 + (8 + sum(7)))

...

10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1 + sum(0)

10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1 + 0

main.cpp

```cpp

#include <iostream>
using namespace std;

int factorial(int);

int main() {
    int n, result;

    cout << "Enter a non-negative number: ";
    cin >> n;

    result = factorial(n);
    cout << "Factorial of " << n << " = " << result;

    return 0;
}

int factorial(int n) {
    if (n > 1) {
        return n * factorial(n - 1);
    } else {
        return 1;
    }
}
```

input

```
Enter a non-negative number: 5
Factorial of 5 = 120

...Program finished with exit code 0
Press ENTER to exit console.
```

• GDB


Thank
you