บทที่ 5
ฟังก์ชัน

รายวิชา การเขียนโปรแกรมคอมพิวเตอร์

ดร.นิฏฐิตา เชิดชู

ฟังก์ชัน คือ อะไร และทำไมต้องเขียนฟังก์ชัน

- ฟังก์ชัน เป็นการแบ่งการเขียนโปรแกรมที่มีความซับซ้อนออกเป็นโปรแกรมย่อยๆ
- ตรวจสอบข้อผิดพลาดได้ง่าย
- สามารถเรียกใช้งานซ้ำได้ง่าย
- แก้ไขได้ง่าย

```
int main()
{
   คำสั่งที่ 1;
   คำสั่งที่ 2;
   คำสั่งที่ 3;
   คำสั่งที่ 4;
   คำสั่งที่ 5;
   คำสั่งที่ 6;
   คำสั่งที่ 7;
   คำสั่งที่ 8;
   คำสั่งที่ 9;
   คำสั่งที่ 10;
   คำสั่งที่ 11;
   คำสั่งที่ 12;
   คำสั่งที่ 13;
   คำสั่งที่ 14
   คำสั่งที่ 15;
   คำสั่งที่ 16;
   คำสั่งที่ 17;
   คำสั่งที่ 18;
}
```

```
int main()
{
   function1();
   function2();
   function3();
}
```

main function

```
void function1 ()
{
   คำสั่งที่ 1;
   คำสั่งที่ 2;
   ...
   คำสั่งที่ 5;
}
```

function 1

```
void function2 ()
{
   คำสั่งที่ 6;
   คำสั่งที่ 7;
   ...
   คำสั่งที่ 10;
}
```

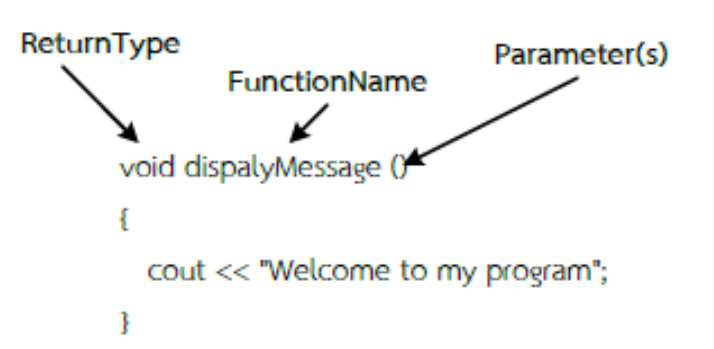
function2

```
void function3 ()
{
   คำสั่งที่ 11;
   คำสั่งที่ 12;
   ...
   คำสั่งที่ 18;
}
```

function 3

5.1 การประกาศ และเรียกใช้งานฟังก์ชัน

```
ReturnType FunctionName (type1 param1, type2 param2, ... )
{
        // function body
        return ReturnValue;
}
```



ฟังก์ชันที่ไม่มีการคืนค่ากลับ

- void
- ไม่มีคำสั่ง return;

```
void displayMessage()
{
        cout << "Welcome to my program";
}
```

ตัวอย่างที่ 5.1

```
1  #include <iostream>
2   using namespace std;
3
4   void displayMessage()
5   {
6       cout << "print out from displayMessage function.\n";
7   }
8
9   int main()
10  {
11      cout << "print out from main function.\n";
12      displayMessage();
13      cout << "print out from main function again!\n";
14      return 0;
15  }
```

ผลลัพธ์ของการรันโปรแกรม

```
print out from main function.
print out from displayMessage function.
print out from main function again!
```

5.2 Function Prototype

- บอกข้อมูลเกี่ยวกับฟังก์ชันให้กับ compiler
- ใช้เพื่อประกาศการใช้งานฟังก์ชัน
- เขียนไว้ส่วนหัวของโปรแกรม

ตัวอย่างที่ 5.2

```
1  #include <iostream>
2   using namespace std;
3
4   // Function Prototype //
5   void displayMessage();
6
7   int main()
8   {
9       cout << "print out from main function.\n";
10      displayMessage();
11      cout << "print out from main function again!\n";
12      return 0;
13  }
14
15  void displayMessage()
16  {
17      cout << "print out from displayMessage function.\n";
18  }
```

ผลลัพธ์ของการรันโปรแกรม

```
print out from main function.
print out from displayMessage function.
print out from main function again!
```

5.3 การส่งค่าให้กับฟังก์ชัน

- ค่าที่ส่งให้กับฟังก์ชัน -> อาร์กิวเมนต์ (argument)
- ตัวแปรที่ใช้รับค่าในฟังก์ชัน -> พารามิเตอร์ (parameter)

```
dispalyIntNumber (10);                // การเรียกใช้ฟังก์ชัน

void dispalyIntNumber (int value)   // ฟังก์ชันเฮดเดอร์
{
   cout << "The number is " << value << endl;
}
```

- การส่งอาร์กิวเมนต์
  - Pass by value
  - Pass by reference

Pass by value

- คัดลอกค่า argument -> parameter
- การเปลี่ยนค่า parameter ภายในฟังก์ชัน ไม่ทำให้ค่าใน argument เปลี่ยน

ตัวอย่างที่ 5.3

```
1  #include <iostream>
2  using namespace std;
3
4  // Function Prototype //
5  void showSum(int num1, int num2, int num3);
6
7  int main()
8  {
9      int value1, value2, value3;
10
11     cout << "Input 3 integers and show sum value\n";
12     cout << "Input first number? ";
13     cin >> value1;
14     cout << "Input second number? ";
15     cin >> value2;
16     cout << "Input third number? ";
17     cin >> value3;
18     showSum(value1, value2, value3);
19     return 0;
20 }
21
22 void showSum(int num1, int num2, int num3)
23 {
24     int sum;
25
26     sum = num1 + num2 + num3;
27     cout << "The sum is " << sum << endl;
28 }
```

ผลลัพธ์ของการรันโปรแกรม

```
Input 3 integers and show sum value
Input first number? 5[Enter]
Input second number? 10[Enter]
Input third number? 4[Enter]
The sum is 19
```

Pass by reference

- ส่งค่าหน่วยความจำ (แทนการก๊อปปี้ค่า) ไปใส่ใน parameter
- การดำเนินการภายในฟังก์ชัน -> เปลี่ยนค่าใน argument ได้

ตัวอย่างที่ 5.4

```
1  #include <iostream>
2   using namespace std;
3
4   // Function Prototype //
5   void doubleNum(int &refValue);
6
7   int main()
8   {
9       int value = 4;
10
11      cout << "In main, the value is " << value << endl;
12      cout << "calling doubleNum() ..." << endl;
13      doubleNum(value);
14      cout << "Back in main, the value is " << value << endl;
15      return 0;
16  }
17
18  void doubleNum(int &refValue)
19  {
20      refValue *= 2;
21  }
```

ผลลัพธ์ของการรันโปรแกรม

```
In main, the value is 4
calling doubleNum() ...
Back in main, the value is 8
```

การคืนค่าของฟังก์ชัน

- ส่งค่าที่ฟังก์ชันประมวลผลได้กลับสู่ฟังก์ชันหลัก
- คืนค่าได้เพียง 1 ค่าเท่านั้น
- ต้องมีการบอกชนิดของข้อมูลที่จะใช้รับค่า
- ใช้คำสั่ง return

ตัวอย่างที่ 5.5

```
1  #include <iostream>
2   using namespace std;
3
4   // Function Prototype //
5   int findSum(int num1, int num2, int num3);
6
7   int main()
8   {
9       int value1, value2, value3;
10      int result;
11
12      cout << "Input 3 integers and show sum value\n";
13      cout << "Input first number? ";
14      cin >> value1;
15      cout << "Input second number? ";
16      cin >> value2;
17      cout << "Input third number? ";
18      cin >> value3;
19      result = findSum(value1, value2, value3);
20      cout << "The result is " << result << endl;
21      return 0;
22  }
23
24  int findSum(int num1, int num2, int num3)
25  {
26      int sum;
27
28      sum = num1 + num2 + num3;
29      return sum;
30  }
```

ผลลัพธ์ของการรันโปรแกรม

```
Input 3 integers and show sum value
Input first number? 5[Enter]
Input second number? 10[Enter]
Input third number? 9[Enter]
The result is 24
```

5.4 Local & Global Variables

- Local variable
  - เรียกใช้งานได้เฉพาะภายในฟังก์ชันเท่านั้น
  - และเรียกใช้งานได้เมื่อมีการเรียกใช้ฟังก์ชันเท่านั้น
  - เมื่อฟังก์ชันทำงานเสร็จ ตัวแปรจะไม่สามารถถูกอ้างถึงได้
  - ในฟังก์ชันต่างกัน สามารถมีตัวแปร local variable ชื่อเดียวกันได้

ตัวอย่างที่ 5.6

```
1  #include <iostream>
2  using namespace std;
3
4  // Function Prototype //
5  void anotherFunction();
6
7  int main()
8  {
9      int num = 10;        //Local variable in main
10
11     cout << "In main, num has value " << num << endl;
12     anotherFunction();
13     cout << "Back in main, num has value " << num << endl;
14     return 0;
15 }
16
17 void anotherFunction()
18 {
19     int num = 5;         //Local variable in anotherFunction
20
21     cout << "In anotherFunction, num has value " << num << endl;
22 }
```

ผลลัพธ์ของการรันโปรแกรม

```
In main, num has value 10
In anotherFunction, num has value 5
Back in main, num has value 10
```

5.4 Local & Global Variables

- Global variable
  - เรียกใช้งานได้ภายในฟังก์ชัน main() และฟังก์ชันอื่นๆ
  - การประกาศตัวแปรต้องทำไว้นอกฟังก์ชัน main()

ตัวอย่างที่ 5.6

```
1  #include <iostream>
2  using namespace std;
3
4  // Function Prototype //
5  void anotherFunction();
6  // Global Variable    //
7  int num;
8
9  int main()
10 {
11     num = 10;
12     cout << "In main, num has value " << num << endl;
13     anotherFunction();
14     cout << "Back in main, num has value " << num << endl;
15     return 0;
16 }
17
18 void anotherFunction()
19 {
20     cout << "In anotherFunction, num has value " << num << endl;
21     num = 5;
22     cout << "Now, it is changed to " << num << endl;
23
24 }
```

ผลลัพธ์ของการรันโปรแกรม

```
In main, num has value 10
In anotherFunction, num has value 10
Now, it is changed to 5
Back in main, num has value 5
```

5.5 Local Static Variable

- โดยปกติแล้วตัวแปร local จะถูกทำลายเมื่อฟังก์ชันทำงานเสร็จ
- หากต้องการเก็บค่าตัวแปรดังกล่าวไว้ในการเรียกใช้ฟังก์ชันครั้งต่อไป ให้ใช้ static variable
- การใช้งาน จะเรียกใช้ได้เฉพาะภายในฟังก์ชันที่ประกาศไว้เท่านั้น เรียกนอกฟังก์ชันไม่ได้ !!!

ตัวอย่างที่ 5.7

```
1   #include <iostream>
2   using namespace std;
3
4   // Function Prototype //
5   void showStatic();
6
7   int main()
8   {
9       int i;
10
11      for(i=0; i<10; i++)
12      {
13          showStatic();
14      }
15      return 0;
16  }
17
18  void showStatic()
19  {
20      static int numCalls = 0;     //Static local variable
21
22      cout << "This function has been called "
23           << ++numCalls << " times. " << endl;
24  }
```

ผลลัพธ์ของการรันโปรแกรม

```
This function has been called 1 times.
This function has been called 2 times.
This function has been called 3 times.
This function has been called 4 times.
This function has been called 5 times.
This function has been called 6 times.
This function has been called 7 times.
This function has been called 8 times.
This function has been called 9 times.
This function has been called 10 times.
```

5.6 Overload Function

- ฟังก์ชันชื่อเดียวกัน แต่มี จ.น. parameter หรือชนิดของ parameter ที่แตกต่างกัน
- เวลาใช้งาน compiler จะเลือกอันที่เหมาะสมให้

ตัวอย่างที่ 5.8

```
1  #include <iostream>
2  #include <iomanip>
3  using namespace std;
4
5  // Function Prototype //
6  int findSquareArea (int width);
7  double findSquareArea (double width);
8
9  int main()
10 {
11     int valueInt;
12     double valueDouble;
13
14     cout << "Enter an integer and a floating-point value: ";
15     cin >> valueInt >> valueDouble;
16
17     cout << "Here are their squares: ";
18     cout << fixed << showpoint << setprecision(2);
19     cout << findSquareArea(valueInt);
20     cout << " and ";
21     cout << findSquareArea(valueDouble) << endl;
22     return 0;
23 }
24
25 int findSquareArea(int width)
26 {
27     return width * width;
28 }
29
30 double findSquareArea(double width)
31 {
32     return width * width;
33 }
```

ผลลัพธ์ของการรันโปรแกรม

```
Enter an integer and a floating-point value: 15 25.5[Enter]
Here are their squares: 225 and 650.25
```

5.7 Exit Function

- ใช้สั่งให้โปรแกรมหยุดทำงานก่อนจบโปรแกรม
- มักใช้เพื่อทดสอบความถูกต้องของการทำงานของโปรแกรม
- EXIT_SUCCESS -> 0
- EXIT_FAILURE -> 1

ตัวอย่างที่ 5.9

```
1  #include <iostream>
2  #include <cstdlib>
3  using namespace std;
4
5  // Function Prototype //
6  void quitProgram();
7
8  int main()
9  {
10     cout << "In main, Calling quitProgram function.\n";
11     quitProgram();
12     cout << "Back in main again.";
13     return 0;
14 }
15
16
17
18 void quitProgram()
19 {
20     cout << "This program terminates with exit function.\n";
21     exit(EXIT_SUCCESS);
22     cout << "This message will never be displayed\n";
23     cout << "because the program has already terminated.";
24 }
```

ผลลัพธ์ของการรันโปรแกรม

```
In main, Calling quitProgram function.
This program terminates with exit function.
```