Computer Programming

#10

Functions

function ทางคณิตศาสตร์

เช่น **f(x, y) = x + y** จะทำการคำนวณหาค่าของผลรวมของ **x** และ **y** ดังนั้นถ้าต้องการทราบค่าของ **11** บวก **8** เราสามารถใช้ฟังก์ชัน **f(x, y)** ได้ เช่น กำหนดให้ **x = 11** และ **y = 8** จะได้ค่าของ **f(x, y)** ซึ่งเป็นผลลัพธ์เท่ากับ **19**

แนวคิดของการเขียน **function** ในโปรแกรมก็คล้ายกัน

ตัวอย่าง function definition

```
double  energy (int  m, double  v)
{
    double  ke;
    ke = 0.5*m*v*v;
    return  ke;
}
```

function definition

(การนิยาม function): ประกอบไปด้วย

ส่วนหัว (header) ของ function คล้ายกับการประกาศตัวแปร จะประกอบด้วย

- ชนิดของการส่งค่ากลับของ function
- ชื่อของ function
- ชนิดของค่าที่ส่งไปให้ function

รูปแบบทั่วไปสำหรับ function ที่มี 2 ค่าที่ส่งไปให้ function (arguments)

```
ftype  fname (type1  name1, type2  name2)
```

- ftype คือ ชนิดของการส่งค่ากลับของ function
- fname คือ ชื่อของ function
- type1, type 2 คือ ชนิดของ argument (ค่าที่ส่งไปให้ function)
- name1, name 2 คือ ชื่อของ argument (ค่าที่ส่งไปให้ function)

เช่น

```
double  energy (int  m, double  v)
```

body ของ function ซึ่งอยู่ในวงเล็บปีกกา จะมีคำสั่งต่าง ๆ ที่ต้องการให้ function นั้นทำงาน

รูปแบบทั่วไปของ function definition ที่มี 2 arguments คือ

```
ftype  fname (type1  name1, type2  name2)
{
    …
    …
}
```

function call

คือ (การเรียกใช้ function): คือคำสั่งที่ใช้ในการย้ายการทำงาน จาก function หนึ่งไปอีก function หนึ่ง

เช่น ใน main function มีการเรียกใช้ function energy

```
.
.
energy(mass, velocity);
.
.
```

ดังนั้นรูปแบบทั่วไปของ function call ที่มี 2 arguments คือ

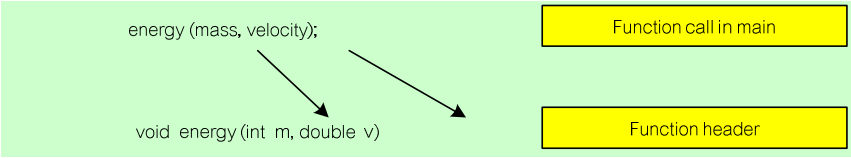
fname(argv1, argv2);

- fname: ชื่อของ function
- argv1, argv2: arguments ของ function ซึ่งมักจะเป็นตัวแปร หรือค่าคงที่

หมายเหตุ:

- ในเรียกใช้ function จะต้องมี function definition ก่อน function ที่เรียกใช้เสมอ
- ชื่อของตัวแปรจาก function ที่เรียกใช้และ function ที่ถูกเรียกใช้ อาจจะเหมือนกันหรือไม่เหมือนกันก็ได้ เช่น มีตัวแปร mass ใน main ก็อาจจะตั้งชื่อ mass ที่ header ใน function energy ก็ได้

- จำนวน ลำดับ และชนิด ของ arguments ใน function call และ header ของ function definition จะต้องสอดคล้องกัน เนื่องจากจะต้องมีการส่งค่าไปยัง function เช่น



จะมีการส่งค่า mass ใน main ไป m ใน energy และค่า velocity ใน main ไป v ใน energy จะเห็นว่า

- จำนวนของ arguments คือ 2
- ลำดับของ arguments เริ่มจาก mass ตามมาด้วย velocity
- ชนิดของ arguments คือ int และ double

การส่งค่ากลับจาก function

- ชนิดของค่าที่ส่งกลับจะต้องตรงกับชนิดของการส่งค่ากลับของ function ที่ประกาศไว้ที่ function definition

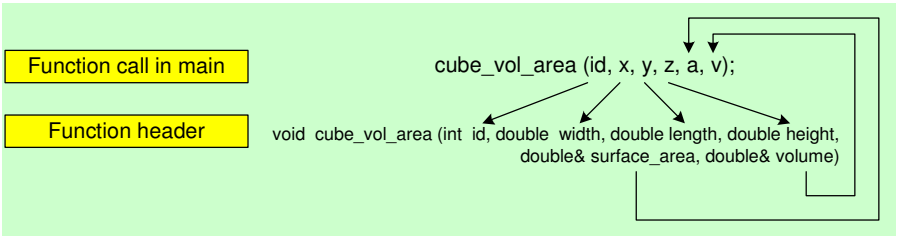
- จะต้องมีคำสั่ง “return” ใน body ของ function ยกเว้น function ชนิด void ไม่จำเป็นจะต้องมี คำสั่ง “return” ก็ได้

Pass by value

การส่งค่าจากตัวอย่างข้างต้น เรียกว่า pass by value ซึ่งค่าของ mass และ velocity ใน main จะถูก copy ไปให้ตัวแปร m และ v ใน energy ตามลำดับ เมื่อตัวแปร m และ v ใน energy มีการเปลี่ยนแปลงค่า ก็จะไม่มีการเปลี่ยนแปลงค่าของ mass และ velocity ใน main

Pass by reference

- สัญลักษณ์ & ใช้ใน function definition เพื่อให้ function สามารถเปลี่ยนแปลงค่าของ argument ที่ส่งมาจาก function อื่นได้ (เช่น main function) เช่น



ในการเปลี่ยนแปลงค่า surface_area และ volume ใน function cube_vol_area จะทำให้ a และ v ใน main function เปลี่ยนไปด้วยตามค่าของ surface_area และ volume ตามลำดับ

ตัวอย่าง Source code (pass by reference)

```
#include <iostream>
using namespace std;
void cube_vol_area (int id, double width, double length, double
  height, double& surface_area, double& volume)
{
  cout<< “For cube id = ” <<id<<endl;
  surface_area = 2*width*height + 2*length*height
                          + 2*width*length;
  volume = width*length*height;
}
```

```
int  main( )
{
  int id=5;
  double a, v;
  double x = 6.3, y = 7.2, z = 1.5;
  cube_vol_area (id, x, y, z, a, v);
  cout<< “cube surface area = ” <<a
          << “ cube volume = ” <<v<<endl;
return 0;
}
```

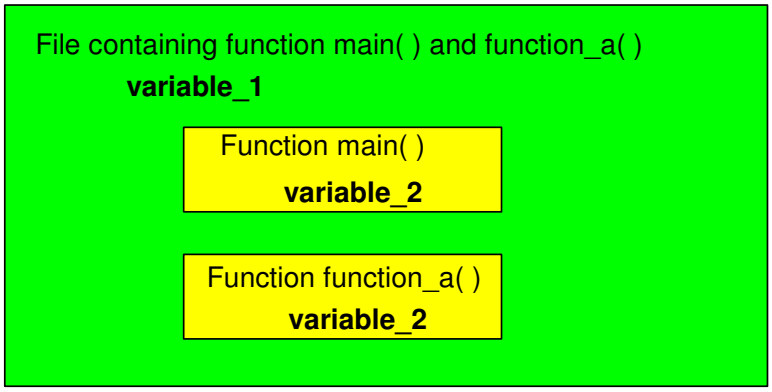
เมื่อ **run program** จะได้ผลลัพธ์ดังนี้

For cube id = 5

cube surface area = 132.22 cube volume = 68.04

Scope

คือพื้นที่หรือขอบเขตที่มีการดำเนินการ (active) เช่น



- File scope: variable_1 เป็นแบบ file scope หรือ global variable คือถูกประกาศอยู่นอก main function และ function อื่น ๆ ดังนั้น function ทุก function สามารถใช้และเปลี่ยนค่าของตัวแปรนี้ได้

- Function scope: variable_2 เป็นแบบ function scope หรือ local variable คือถูกประกาศอยู่ใน function จะใช้อยู่ใน function ที่ประกาศเท่านั้น ซึ่ง variable_2 ใน main function และ function_a จะไม่เหมือนกันคือถ้าเปลี่ยนค่า variable_2 ใน main function variable_2 ใน function_a ก็จะไม่เปลี่ยน

- Block scope: มีการประกาศตัวแปรใน block (เปิดด้วย { และปิดด้วย }) ซึ่งตัวแปรก็สามารถใช้ได้ใน block เท่านั้น เช่น

```
for ( . . . )
{
    int  n;
    .
    .
}
```

ตัวแปร n ใน for loop สามารถใช้ได้ใน for loop เท่านั้น

LAB

- ให้ save file ชื่อ **“lab10_รหัสประจำตัวนิสิต.cpp”**
  เช่น lab10_1234567.cpp
- ให้เขียน comment บนสุดของโปรแกรมดังต่อไปนี้

```
// Name:   First name   Last name
// ID:     Student ID
// 305171 lab 10
```

1. ให้เขียน **function** ในโปรแกรมเพื่อหาค่า

$$y = 2.7x^5 - 3.2x^3 + x - 10.54$$

โดยรับค่า **x** ที่ **main function** แล้วส่งค่า **x** ไปให้ **function** ข้างต้น ให้ส่งค่า **y** กลับจาก **function** และแสดงค่า **y** ออกทางหน้าจอภาพโดยใช้คำสั่งใน **main function**

2. ให้เขียน **function** ในโปรแกรมเพื่อรับค่าจำนวนเต็ม **3** ค่ามาจาก **function** อื่น เช่น **main function** แล้วนำมาหาค่าสูงสุด จากนั้นให้ส่งค่าสูงสุดกลับ