บทที่ 4
การเขียนโปรแกรมแบบวนรอบ

รายวิชา การโปรแกรมคอมพิวเตอร์

ดร.นิฏฐิตา เชิดชู

4.1 ตัวดำเนินการสำหรับการเพิ่มและลดค่า

- การเพิ่ม หรือลดค่าทีละ 1
  - x = x + 1;
  - x += 1;
  - x++;
  - ++x;
  - y = y – 1;
  - y -=1;
  - y--;
  - --y;
- การเพิ่ม หรือลดค่าทีละมากกว่า 1
  - X = x + 5;
  - X += 5;
  - y = y-10;
  - y -= 10;

++ และ --

- หาก ++ หรือ - - เขียนไว้ก่อนตัวแปร เช่น ++x หรือ - -y
  - ให้ทำการเพิ่มค่า x หรือ ลดค่า y ไป 1 ค่า ก่อนจึงทำชุดคำสั่ง
- หาก ++ หรือ - - เขียนไว้หลังตัวแปร เช่น x++ หรือ y- -
  - ให้ทำชุดคำสั่งก่อนแล้วจึงค่อยทำการเพิ่มหรือลดค่า

ตัวอย่างที่ 4.1

```
1  #include <iostream>
2   using namespace std;
3   int main()
4   {
5       int num = 0;
6       int x, y;
7       x = ++num;
8       y = --num;
9       cout << "x = " << x << endl;
10      cout << "y = " << y << endl;
11      return 0;   }
```

ผลลัพธ์ของการรันโปรแกรม

```
x = 1
y = 0
```

ตัวอย่างที่ 4.2

```
1  #include <iostream>
2
3   using namespace std;
4   int main()
5   {
6       int num = 0;
7       int x, y;
8       x = num++;
9       y = num--;
10      cout << "x = " << x << endl;
11      cout << "y = " << y << endl;
12      return 0;
13  }
```

ผลลัพธ์ของการรันโปรแกรม

```
x = 0
y = 1
```

++ และ - -

ตัวอย่างที่ 4.3

```
1  #include <iostream>
2   using namespace std;
3   int main()
4   {
5       int a = 10;
6       int b = 20;
7       int c = a + b++;
8       cout << "c = " << c << endl;
9       cout << "b = " << b << endl;
10      return 0;
11  }
```

ผลลัพธ์ของการรันโปรแกรม

```
c = 30
b = 21
```

ตัวอย่างที่ 4.4

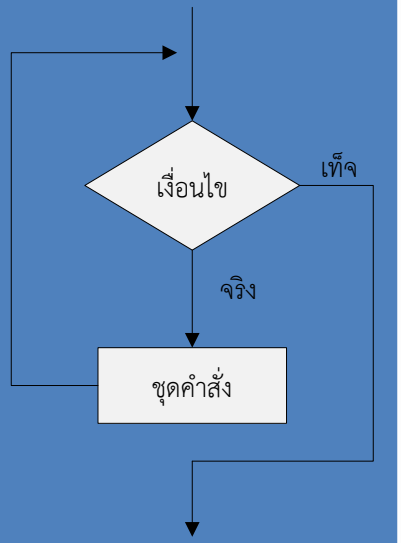
```
1  #include <iostream>
2   using namespace std;
3   int main()
4   {
5       int student = 39;
6       if(student++ >= 40)
7       {
8           cout << "No. of students = " << student << endl;
9           cout << "The number of student id greater than 40.";
10          cout << "The class is too small.\n";
11      }
12      else
13      {
14          cout << "No. of students = " << student << endl;
15          cout << "Welcome to the class !";
16      }
17      return 0;
18  }
```

ผลลัพธ์ของการรันโปรแกรม

```
No. of students = 40
Welcome to the class
```

4.2 การเขียนโปรแกรมแบบวนรอบแบบ while

```
while(เงื่อนไข)
{
    //ชุดคำสั่งที่ต้องการให้โปรแกรมทำงาน
    คำสั่งที่ 1;
    คำสั่งที่ 2;
    .
    .
    .
    .
    คำสั่งที่ n;
}
```



ตัวอย่างที่ 4.5

```
1  #include <iostream>
2   using namespace std;
3   int main()
4   {
5       int num = 5;
6       while (num >= 0)
7       {
8           cout << num-- << " ";
9       }
10      return 0;
11  }
```

ตัวอย่างที่ 4.5 (ต่อ)

ผลลัพธ์ของการรันโปรแกรม

```
5 4 3 2 1 0
```

ข้อควรระวังในการเขียนโปรแกรมวนลูป

- การเกิดการวนรอบอนันต์
- เงื่อนไขไม่มีวันเป็นเท็จ -> ไม่สามารถออกจากการวนรอบได้

ตัวอย่างที่ 4.6

```
1  #include <iostream>
2   using namespace std;
3   int main()
4   {
5       int num = 5;
6       while (num >= 0)
7       {
8           cout << num++ << " ";
9       }
10      return 0;
11  }
```

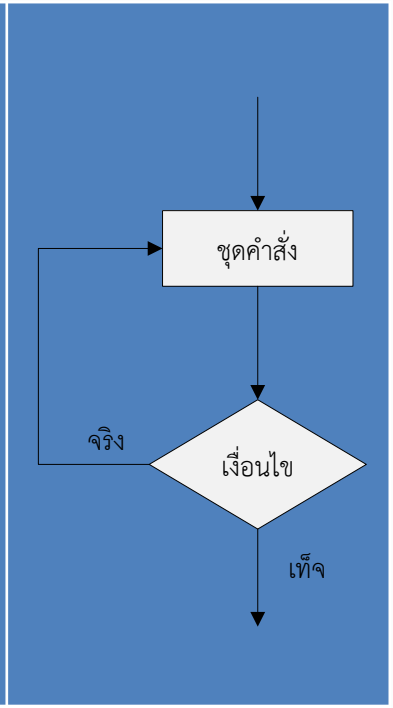
ผลลัพธ์ของการรันโปรแกรม

```
5 6 7 8 9 10 11 12 13 14 15 16 ...
```

4.3 การเขียนโปรแกรมวนรอบแบบ do-while

- เริ่มทำงานชุดคำสั่งในลูปก่อน 1 รอบ
- แล้วจึงทำการทดสอบเงื่อนไข
- หากเป็นจริง ก็จะทำงานชุดคำสั่งต่อไป
- หากเป็นเท็จ ก็จะออกจากลูป

```
do
{
    //ชุดคำสั่งที่ต้องการให้โปรแกรมทำงาน
    คำสั่งที่ 1;
    คำสั่งที่ 2;
    .
    .
    .
    .
    คำสั่งที่ n;
}
while(เงื่อนไข);
```



ตัวอย่างที่ 4.7

```
1  #include <iostream>
2   using namespace std;
3   int main()
4   {
5       int num = 5;
6       do
7       {
8           num++;
9           cout << "num = " << num << endl;
10      }
11      while (num < 5);
12      cout << "program finished !";
13      return 0;
14  }
```

ผลลัพธ์ของการรันโปรแกรม

```
num = 6
```

ตัวอย่างที่ 4.8

```
1 #include <iostream>
2  using namespace std;
3  int main()
4  {
5      double width, height, area;
6      char user_input;
7      do
8      {
9          cout << "Calculate rectangle area\n";
10         cout << "enter width : ";
11         cin >> width;
12         cout << "enter height : ";
13         cin >> height;
14         area  = width * height;
15         cout << "area = " << area << endl;
16         cout << "Do you want to calculate again ? (Y/N)";
17         cin >> user_input;
18     }
19     while(user_input == 'y' || user_input == 'Y');
20     cout << "program finished !";
21     return 0;
22 }
```

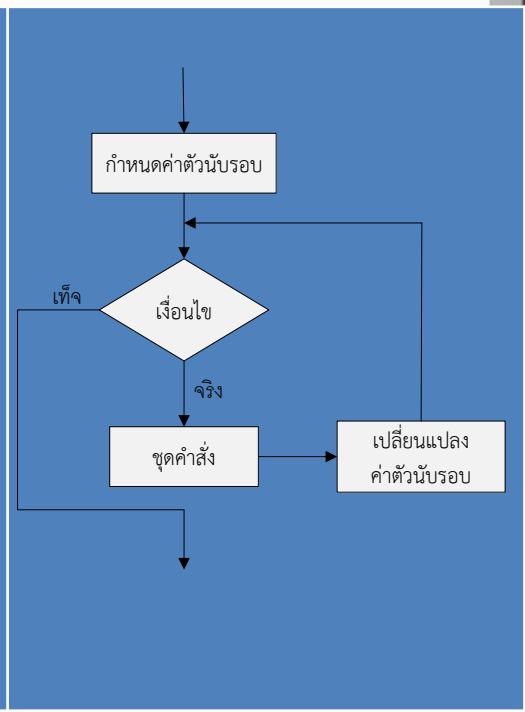
ผลลัพธ์ของการรันโปรแกรม

```
Calculate rectangle area
enter width : 5.0
enter height : 3.5
area = 17.5
Do you want to calculate again ? (Y/N)Y
Calculate rectangle area
enter width : 10
enter height : 5
area = 50
Do you want to calculate again ? (Y/N)N
program finished !
```

4.4 การเขียนโปรแกรมวนรอบแบบ for

- นิยมใช้กับการทำงานที่รู้จำนวนรอบของการวนรอบที่แน่นอน
- คำสั่ง for ประกอบไปด้วย 3 ส่วน
  - การกำหนดค่าเริ่มต้นในการนับรอบการทำงาน
  - เงื่อนไขที่กำหนดการทำงาน
  - การเปลี่ยนแปลงค่าของตัวแปรที่ใช้นับรอบ

```
for(ค่าเริ่มต้น; เงื่อนไข; การเปลี่ยนค่าการนับรอบ)
{
    //ชุดคำสั่งที่ต้องการให้โปรแกรมทำงาน
    คำสั่งที่ 1;
    คำสั่งที่ 2;
    .
    .
    .
    .
    คำสั่งที่ n;
}
```



ตัวอย่างที่ 4.9

```
1 #include <iostream>
2  using namespace std;
3  int main()
4  {
5      int num;
6      cout << "multiplication table" << endl;
7      cout << "--------------------" << endl;
8      for(num = 1; num <=12; num++)
9      {
10         cout << "2 x " << num << " = " << 2*num <<endl;
11     }
12     return 0;
13 }
```

ผลลัพธ์ของการรันโปรแกรม

```
multiplication table
--------------------
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
2 x 11 = 22
2 x 12 = 24
```

การเขียนโปรแกรม ตัวอย่างที่ 4.9 ด้วย while

ตัวอย่างที่ 4.10

```
1  #include <iostream>
2   using namespace std;
3   int main()
4   {
5       int num = 1;
6       cout << "multiplication table" << endl;
7       cout << "--------------------" << endl;
8       while(num <= 12)
9       {
10          cout << "2 x " << num << " = " << 2*num <<endl;
11          num++;
12      }
13      return 0;
14  }
```

4.5 การเขียนโปรแกรมวนรอบแบบ nested loop

- ลูปซ้อนลูป
- ทำงานในลูปในเสร็จก่อน จึงจะเริ่มทำงานลูปนอก

ตัวอย่างที่ 4.11

```
1  #include <iostream>
2   using namespace std;
3   int main()
4   {
5       int i, j;
6
7       for (i = 1; i <= 8; i++)
8       {
9           cout << "Row:" << i << "\t";
10          for(j = 1; j <= i; j++)
11          {
12              cout << "*";
13          }
14          cout << endl;
15      }
16      return 0;
17  }
```

ลูปนอก

ลูปใน

ผลลัพธ์ของการรันโปรแกรม

```
Row:1    *
Row:2    **
Row:3    ***
Row:4    ****
Row:5    *****
Row:6    ******
Row:7    *******
Row:8    ********
```

4.6 คำสั่ง break

- ใช้หยุดการทำงานการวนรอบ
- เมื่อเจอ break; โปรแกรมจะหยุดการทำงานในคำสั่งที่เหลืออยู่ในลูป
- กระโดดออกมาทำงานในคำสั่งที่อยู่ท้ายลูป
- สำหรับ nest loop คำสั่ง break; จะมีผลกับเฉพาะลูปที่ใช้คำสั่งเท่านั้น

ตัวอย่างที่ 4.12

```
1  #include <iostream>
2   using namespace std;
3   int main()
4   {
5       int i, j;
6
7       for (i = 1; i <= 8; i++)
8       {
9           cout << "Row:" << i << "\t";
10          for(j = 1; j <= i; j++)
11          {
12              if(j > 5)
13                  break;
14
15              cout << "*";
16          }
17          cout << endl;
18      }
19      return 0;
20  }
```

ลูปนอก

ลูปใน

ผลลัพธ์ของการรันโปรแกรม

```
Row:1    *
Row:2    **
Row:3    ***
Row:4    ****
Row:5    *****
Row:6    *****
Row:7    *****
Row:8    *****
```

4.7 คำสั่ง continue

- ใช้งานกับลูป
- เมื่อเห็น continue; จะสิ้นสุดการทำงานในรอบลูปนั้นทันที
- While – ทดสอบเงื่อนไขด้านบน
- Do-while – ทดสอบเงื่อนไขด้านล่าง
- For – ทำการเพิ่มหรือลดค่า index
- Nest loop คำสั่ง continue; จะมีผลกับลูปที่มีการใช้คำสั่งเท่านั้น

ตัวอย่างที่ 4.13

```
 1 #include <iostream>
 2  using namespace std;
 3  int main()
 4  {
 5      int i, j;
 6
 7      for (i = 1; i <= 8; i++)
 8      {
 9          cout << "Row:" << i << "\t";
10          for(j = 1; j <= i; j++)
11          {
12              if(j < 5)
13                  continue;
14
15              cout << j << " ";
16          }
17          cout << endl;
18      }
19      return 0;
20  }
```

ลูปนอก

ลูปใน

ผลลัพธ์ของการรันโปรแกรม

```
Row:1
Row:2
Row:3
Row:4
Row:5    5
Row:6    5 6
Row:7    5 6 7
Row:8    5 6 7 8
```