112 學年度　第一學期　　課號：I4390

進階程式開發技術

程式作業報告

程式作業 **01**

姓名：李信均

座號：I4B18

(1) Write an application to compute the week day number from its week day name. For example, the week day number of the week day name "Monday" is 1. In your program, the week day name is represented by a string. Your program should input the week day name, using **switch**-statement on String to find the week day number of its week day name, and output both the week day name and its week day number. Invalid week day name should output the message: "**Error: invalid week day name**". (25%)

主要想法說明:

Import Scanner 並讓使用者輸入 weekday 字串，

```
1  package HW1;
2
3
4  import java.util.Scanner;
5
6  public class weekday {
7
8      public static void main(String[] args) {
9          // TODO Auto-generated method stub
10         Scanner scanner = new Scanner(System.in);
11         System.out.print("Input weekday(String) : ");
12         String weekdayName = scanner.nextLine();
```

呼叫 function 回傳對應的 weekday number，function 使用的是 switch case 判斷，如果使用者輸入的字串不為有效的 weekday，回傳 0，

```
14         int weekdayNumber = getWeekdayNumber(weekdayName.toLowerCase());
```

```
24      public static int getWeekdayNumber(String weekdayName)
25              int weekdayNumber;
26              switch (weekdayName) {
27                  case "monday":
28                      weekdayNumber = 1;
29                      break;
30                  case "tuesday":
31                      weekdayNumber = 2;
32                      break;
33                  case "wednesday":
34                      weekdayNumber = 3;
35                      break;
36                  case "thursday":
37                      weekdayNumber = 4;
38                      break;
39                  case "friday":
40                      weekdayNumber = 5;
41                      break;
42                  case "saturday":
43                      weekdayNumber = 6;
44                      break;
45                  case "sunday":
46                      weekdayNumber = 7;
47                      break;
48                  default:
49                      // Invalid weekday name
50                      weekdayNumber = 0;
51                      break;
52              }
53              return weekdayNumber;
54          }
55
56  }
```

回到 main function，判斷 weekday number，輸出 weekday & weekday number 或者是 error，並關閉 scanner 結束程式。

```
16          if (weekdayNumber != 0) {
17              System.out.println("Output: " + weekdayName + " " + weekdayNumber);
18          } else {
19              System.out.println("Error: invalid week day name");
20          }
21          scanner.close();
22      }
```

輸出:

正確輸入

```
<terminated> weekday [Java Application] D:\Java JDK17\bin\javaw.
Input weekday(String) : monday
Output: monday 1
```

錯誤輸入

```
Input weekday(String) : aa
Error: invalid week day name
```

(2) Write an application that print a pattern of ***k*** rows. Your program must read in a number ***k***. Then, output the pattern with ***k*** rows. The ***i***-th row has 2(***k*** - ***i***) spaces before the first number. The ***i***-th row starts from **1** and contains numbers that increased by **1** until number *i* and then decrease by 1 until number 1. The numbers in each row are separated by one spaces. For example, if ***k*** = 4, output the pattern as follows: (25%)

```
   1
  1 2 1
 1 2 3 2 1
1 2 3 4 3 2 1
```

主要想法說明:

Import Scanner 並讓使用者輸入 rows 數量，並將其存為 k，

```
1  package HW1;
2
3
4  import java.util.Scanner;
5
6  public class triangle {
7
8      public static void main(String[] args) {
9          // TODO Auto-generated method stub
10         Scanner scanner = new Scanner(System.in);
11         System.out.print("Input the number of rows: ");
12         int k = scanner.nextInt();
```

使用兩層迴圈輸出三角形，

一開始輸出總行數減當前行數的空格數量，因為每個值之間多一個空格排版較好看，因此這裡每次輸出兩個空格，

第二步輸出從 1 開始到當前行數的值，

第三部輸出從當前行數-1 到 1 的值

```
14         for (int i = 1; i <= k; i++) {
15             for (int j = 1; j <= (k - i); j++) {
16                 //space in front
17                 System.out.print("  ");
18             }
19
20             for (int j = 1; j <= i; j++) {
21                 //numerical increase
22                 //1 -> maximum
23                 System.out.print(j + " ");
24             }
25
26             for (int j = i - 1; j >= 1; j--) {
27                 //numerical decrease
28                 //maximum-1 -> 1
29                 System.out.print(j + " ");
30             }
31
32             System.out.println();
33         }
34         scanner.close();
35     }
36 }
```

輸出:

Rows = 4

```
Input the number of rows: 4
      1
    1 2 1
  1 2 3 2 1
1 2 3 4 3 2 1
```

Rows = 6

```
Input the number of rows: 6
          1
        1 2 1
      1 2 3 2 1
    1 2 3 4 3 2 1
  1 2 3 4 5 4 3 2 1
1 2 3 4 5 6 5 4 3 2 1
```

(3) Value π can be approximated by using the following series: (25%)

$$\pi = 4(1 - 1/3 + 1/5 - 1/7 + 1/9 - 1/11 + \ldots + 1/(2k-1) - 1/(2k+1))$$

Write program that output the π value for k = 10000. 20000, … , and 100000.

主要想法說明:

使用兩層迴圈，第一層為設定 k 的最大值，k = 10000. 20000, . . . , and 100000，

第二層為讓 k 從 1 開始，判斷 k 是否為偶數，

如果是則讓 pi-(1/(2*k)-1)，如果否則讓 pi-(1/(2*k)-1)，

第二層結束後讓 pi*4 輸出，並繼續第一層將 pi 值初始化、k 的最大值加 10000

```java
package HW1;


public class pi {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        for (int num = 1; num <= 10; num++) {
            int max = num * 10000;
            double pi = 0;
            for(double k = 1; k <= max; k++) {
                if ( k % 2 == 0) {
                    pi -= 1/((2*k)-1);
                }
                else {
                    pi += 1/((2*k)-1);
                }
            }
            pi = pi * 4;
            System.out.println("k = " + max + ", pi = " + pi);
        }
    }
}
```

輸出:

隨著 k 值越大，越接近 pi。

```
k = 10000, pi = 3.1414926535900345
k = 20000, pi = 3.1415426535898248
k = 30000, pi = 3.141559320256462
k = 40000, pi = 3.1415676535897985
k = 50000, pi = 3.1415726535897814
k = 60000, pi = 3.141575986923102
k = 70000, pi = 3.141578367875482
k = 80000, pi = 3.1415801535897496
k = 90000, pi = 3.1415815424786238
k = 100000, pi = 3.1415826535897198
```

(4) Given a set of points (x,y) in the two dimensional xy-plane, for each point ***p*** find the point that closest to point ***p***. Your program should input the set of points, compute each point's closest point, and print out each point and its closest point. (25%)

主要想法說明:

先宣告一個 class point，member 有 x, y，function 有 distance()、toString()，目的為輸出與計算點和點間的距離，計算距離使用數學公式(x-a)^2+(y-b)^2 開根號，

```
13    public point(int x, int y) {
14        this.x = x;
15        this.y = y;
16    }
17
18    //compute the distance of point a and b.
19    public double distance(point p) {
20        int disX = (this.x - p.x)*(this.x - p.x);
21        int disY = (this.y - p.y)*(this.y - p.y);
22        return Math.sqrt(disX + disY);
23    }
24
25    //output
26    @Override
27    public String toString() {
28        return "(" + this.x + ", " + this.y + ")";
29    }
30 }
```

這個二維的平面圖範圍為 100*100，所有點都在 x < 100, y < 100

使用 List 存放 point，這些 point 用 random 的方式產生五個並輸出，

```
34    public static void main(String[] args) {
35        // TODO Auto-generated method stub
36
37        //create the set to store points
38        List<point> points = new ArrayList<>();
39        System.out.println("other points");
40
41        //random 5 points
42        for (int i = 0; i < 5; i++) {
43            point p = new point(new Random().nextInt(100), new Random().nextInt(100));
44            while ( points.indexOf(p) != -1) {
45                p = new point(new Random().nextInt(100), new Random().nextInt(100));
46            }
47            points.add(p);
48            System.out.println(p.toString());
49        }
```

前面有 import Scanner，這裡讓使用者輸入 x, y 並輸出，當作目標點，

```
50          Scanner scanner = new Scanner(System.in);
51          System.out.print("Input x(<100): ");
52          int x = scanner.nextInt();
53          while (x >= 100) {
54              x = scanner.nextInt();
55          }
56          System.out.print("Input y(<100): ");
57          int y = scanner.nextInt();
58          while (y >= 100) {
59              y = scanner.nextInt();
60          }
61          point target = new point(x, y);
62          System.out.println("target points");
63          System.out.println(target.toString());
```

宣告一數為 closest = int 最大值和一點 maxPoint，
呼叫 point function distance()，比較每個點的距離並輸出點和點的距離，
如果比 closest 小則讓 closest = distance，maxPoint = this point，
結束後輸出這個最近的點。

```
64          double closest = Integer.MAX_VALUE;
65          point maxPoint = new point(0, 0);
66          for ( point p:points) {
67              System.out.println(p.toString() + " to " + target.toString() + " = " + p.distance(target));
68              if ( p.distance(target) < closest) {
69                  maxPoint = p;
70                  closest = p.distance(target);
71              }
72          }
73          System.out.println("closest point is " + maxPoint);
74          scanner.close();
75      }
76
77  }
```

輸出：

X = 67, Y = 28，選擇第一個點最近的檢查

```
other points
(66, 27)
(51, 46)
(85, 41)
(55, 19)
(96, 74)
Input x(<100): 67
Input y(<100): 28
target points
(67, 28)
(66, 27) to (67, 28) = 1.4142135623730951
(51, 46) to (67, 28) = 24.08318915758459
(85, 41) to (67, 28) = 22.20360331117452
(55, 19) to (67, 28) = 15.0
(96, 74) to (67, 28) = 54.378304497290095
closest point is (66, 27)
```

X = 25, Y = 32，選擇隨機的檢查

```
other points
(0, 65)
(32, 55)
(89, 90)
(46, 18)
(49, 51)
Input x(<100): 25
Input y(<100): 32
target points
(25, 32)
(0, 65) to (25, 32) = 41.400483088968905
(32, 55) to (25, 32) = 24.041630560342615
(89, 90) to (25, 32) = 86.37129152675674
(46, 18) to (25, 32) = 25.238858928247925
(49, 51) to (25, 32) = 30.610455730027933
closest point is (32, 55)
```