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 XX

姓名：李信均

座號：I4B18

(1) Design a program that sorts elements in an array. This program must be designed with the following requirements: **(40 分)**

- Write a **generic** method, called **selectionSort**. The generic **selectionSort** method is based on the ***selection sort*** algorithm which sorts elements in an array.
- Write a **generic** method to output the elements of an array.
- Write a main program that inputs array, sorts array, and outputs an **Integer** array and a **Double** array.

Hint: Use **<T extends Comparable<T>>** in the type-parameter section for this method, so that you can use method **compareTo** to compare the objects of the type **T** represents.

進入 main function，宣告一個長度為 5 的整數陣列和 scanner

```
29    public static void main(String[] args) {
30        Scanner scan = new Scanner(System.in);
31
32        Integer[] intArr = new Integer[5];
```

輸入五個整數

```
33        int element = 0;
34        int index = 0;
35        System.out.println("\nEnter 5 integer");
36        while(index < 5) {
37            System.out.print((index+1)+": ");
38            element = scan.nextInt();
39            intArr[index] = element;
40            index++;
41        }
```

將排序前的鎮列印出，這裡使用 function 來印出

```
43        System.out.print("\nBefore: ");
44        printArray(intArr);
```

印出的 function 使用的型態泛型，且宣告一個泛型的陣列以處理 double 和 integer

```
25    public static <T> void printArray(T[] arr) {
26        System.out.println(Arrays.toString(arr));
27    }
```

開始進行 selectionSort

```
46        doSelectionSort(intArr);
```

selectionSort 使用泛型且繼承 comparable，為的是比較數字，因為 double 和 integer 的父類別都是 Number，並宣告一個泛型的陣列

```
8●    public static <T extends Comparable<T>> void doSelectionSort(T[] arr) {
9         int n = arr.length;
```

利用 compareTo 實作 selectionSort 的演算法，當 compareTo 回傳的值<0 代表小於，=0 代表等於，>0 代表大於

```
11          for (int i = 0; i < n - 1; i++) {
12              int minIndex = i;
13              for (int j = i + 1; j < n; j++) {
14                  if (arr[j].compareTo(arr[minIndex]) < 0) {
15                      minIndex = j;
16                  }
17              }
18
19              T temp = arr[minIndex];
20              arr[minIndex] = arr[i];
21              arr[i] = temp;
22          }
23      }
```

印出排序後的結果

```
47          System.out.print("\nAfter: ");
48          printArray(intArr);
```

Double 的陣列一樣先宣告長度為 5 的陣列並輸入

```
50          Double[] doubleArr = new Double[5];
51          Double dElement = 0.0;
52          index = 0;
53          System.out.println("\nEnter 5 double");
54          while(index < 5) {
55              System.out.print((index+1)+": ");
56              dElement = scan.nextDouble();
57              doubleArr[index] = dElement;
58              index++;
59          }
```

接著印出排序前並實作 selectionSort 最後再印出結果，實作上跟整數陣列大同小異。

```
61          System.out.print("\nBefore: ");
62          printArray(doubleArr);
63
64          doSelectionSort(doubleArr);
65          System.out.print("\nAfter: ");
66          printArray(doubleArr);
67
68          scan.close();
69      }
70
71  }
```

輸出：

整數陣列

```
Enter 5 integer
1: 123
2: 32423
3: 123124
4: 423421
5: 124124

Before: [123, 32423, 123124, 423421, 124124]

After: [123, 32423, 123124, 124124, 423421]
```

Double 陣列

```
Enter 5 double
1: 123.213
2: 214.234
3: 4235.12
4: 3251.423
5: 123.222

Before: [123.213, 214.234, 4235.12, 3251.423, 123.222]

After: [123.213, 123.222, 214.234, 3251.423, 4235.12]
```

(2) Design a program that can insert ***k*** numbers in order into a **LinkedList** object. Your program should input the value of ***k*** and ***k*** numbers, compute the total sum and the average of those numbers, and output the ***k*** numbers and the total sum, and average of those numbers. Your program must use **Iterator** object to access the numbers in the **LinkedList** object. Test your program for **Integer** and **Double** numbers. **(40 分)**

進入 main function，輸入 k(list 的大小)

```
 9●     public static void main(String[] args) {
10         Scanner scan = new Scanner(System.in);
11
12         System.out.print("Enter k: ");
13         int k = scan.nextInt();
```

宣告 linkedList，型態為 double 和 integer 的父類別 Number

```
15         LinkedList<Number> numbersList = new LinkedList<>();
```

輸入 k 個數字，考慮到最後算出總合和平均都是用 double 表示，這裡不管輸入的是否為整數都直接存為 double

```
17         for (int i = 0; i < k; i++) {
18             System.out.print((i + 1) + ": ");
19             numbersList.add(scan.nextDouble());
20         }
```

使用 iterator 遍歷 numberList，並用 hasNext 判斷是否還存在下個數字，使用 iterator 算出總合

```
22         double sum = 0;
23         Iterator<Number> iterator = numbersList.iterator();
24         while (iterator.hasNext()) {
25             sum += iterator.next().doubleValue();
26         }
```

用總和/k 算出平均並輸出

```
27         double average = sum / k;
28
29         System.out.println("LinkedList: " + numbersList);
30         System.out.println("Sum: " + sum);
31         System.out.println("Average: " + average);
32
33         scan.close();
34     }
```

輸出：

```
Enter k: 3
1: 1
2: 5
3: 8.4
LinkedList: [1.0, 5.0, 8.4]
Sum: 14.4
Average: 4.8
```

(3) What is the following class converted to after **type erasure**?

```
public class Pair<K, V> {
  public Pair(K key, V value) {
     this.key = key;
    this.value = value;
  }

  public K getKey(); { return key; }
  public V getValue(); { return value; }

   public void setKey(K key)    { this.key = key; }
   public void setValue(V value) { this.value = value; }

   private K key;
   private V value;
 }
```

將 K、V 抹除全部換為 Object 即可

```
3 public class pair {
4     public pair(Object key, Object value) {
5         this.key = key;
6         this.value = value;
7     }
8
9     public Object getKey() {
0         return key;
1     }
2
3     public Object getValue() {
4         return value;
5     }
6
7     public void setKey(Object key) {
8         this.key = key;
9     }
0
1     public void setValue(Object value) {
2         this.value = value;
3     }
4
5     private Object key;
6     private Object value;
7 }
```

(4) Given the following classes: **(10 分)**

```
class Shape { /* ... */ }
class Circle extends Shape { /* ... */ }
class Rectangle extends Shape { /* ... */ }

class Node<T> { /* ... */ }
```

Will the following code compile? If not, why?

```
Node<Circle> nc = new Node<>();
Node<Shape> ns = nc;
```

這個將會導致無法執行，即使 Circle 為 Shape 的子類別，但

Node<Circle>並不是 Node<Shape>的子類別，因為 Node 宣告是泛型。