



# Object Oriented Programming

Generics and ArrayLists

Lab 12

# ● Content

- ArrayLists
- Generics

CSLAB

# ● ArrayList Class

- **An ArrayList is a dynamic data structure. This means**
  - items can be added and removed from the list.
  - The length can grow or shrink as items are added and removed
- **ArrayList is a class in the standard Java libraries**
  - `Java.util.ArrayList`.

CSLAB

## ● ArrayList vs Arrays

- A normal array in java is a static data structure, because you stuck with the initial size of your array.
- An **ArrayList** serves the same purpose as an array, except that an **ArrayList** can change length while the program is running

CSLAB

# ● ArrayList Class

- The class **ArrayList** is implemented using an array as a private instance variable

```
public class ArrayList<E> extends AbstractList<E>
    implements List<E>, RandomAccess, Cloneable, java.io.Serializable
{
    private static final long serialVersionUID = 8683452581122892189L;

    /**
     * The array buffer into which the elements of the ArrayList are stored.
     * The capacity of the ArrayList is the length of this array buffer.
     */
    private transient Object[] elementData;


    /**
```

- When this hidden array is full, a new larger hidden array is created and the data is transferred to this new array

CSLAB

# ● ArrayList Class

## • Things to know about ArrayLists.

- The **base type** of an ArrayList must be a **class type** (or other reference type):
- it cannot store primitive types. 
- Primitive types must be boxed into their Wrapper class before they can be used.
- ArrayLists are less efficient than Arrays.

CSLAB

## ● Using ArrayLists

- To set up an ArrayList, you first have to
  - import the package from the `java.util` library:
    - `import java.util.ArrayList;`
  - Specify the base type
    - `ArrayList<BaseType> aList = new ArrayList<BaseType>();`
- It is possible to specify the initial size of the ArrayList.
  - `ArrayList<String> aList = new ArrayList<String>(20);`

초기 size 지정 ⇒ 값이 들어가기  
전에 늘어난다

CSLAB

# ● ArrayList Notation

- With regular Arrays it is easy to access items at specific indices of the array using [ ].
  - String str = myArray[i];*  
*myArray[j] = "Hello";*
- ArrayLists do not provide the convenient [ ] notation. Instead the get(index), set(index, object) and add(object) methods may be used.

– *String str = myArrayList.get(i);*  
*myArrayList.set(j, "Hello");*

*vote[] arr = new Vote[20];*

*null | null | null | | - ...*

초기배열에  
생성값으로  
쓰임새를  
⇒ 값 수정가능.

☐ Note: set cannot be used to initialize an item. It can only replace existing items.



# ● ArrayList Notation

- To insert items into the ArrayList for the first time you can use the **.add()** method.

- *myArrayList.add("Goodbye");*  
*myArrayList.add("cruel");*  
*myArrayList.add("world.");*

☐ This adds items sequentially into the ArrayList i.e. 0,1,2...

Goodbye	cruel	world			
0	1	2	3	4	...

CSLAB

# ArrayList Notation

- The `add()` method is overloaded and can accept another parameter. This allows you to insert at a specific index.

`add(index, object);`

```
myArrayList.add("Goodbye");
```

```
myArrayList.add("world");
```

```
myArrayList.add(1, "cruel.");
```

☐ The items from index 1 are shifted to allow for the word "cruel" to be inserted.

⇒ 1 인덱스 값이 존재하면 right shift 후, cruel add.

Goodbye	world				
0	1	2	3	4	...



Goodbye	cruel	world			
0	1	2	3	4	...

## ● ArrayList Notation

- Return the number of elements stored in the ArrayList by using the .size() method. *ArrayList은 length 대신*

– `System.out.println( myArrayList.size() );`



3

Goodbye	cruel	world			
0	1	2	3	4	...

CSLAB

# ● ArrayList Class

## Display 14.1 Some Methods in the Class ArrayList

### CONSTRUCTORS

```
public ArrayList<Base_Type>(int initialCapacity)
```

Creates an empty ArrayList with the specified *Base\_Type* and initial capacity.

```
public ArrayList<Base_Type>()
```

Creates an empty ArrayList with the specified *Base\_Type* and an initial capacity of 10.

(continued)

CSLAB 3

# ● ArrayList Class

## Display 14.1 Some Methods in the Class ArrayList

### ARRAYLIKE METHODS

```
public Base_Type set( int index, Base_Type newElement)
```

Sets the element at the specified index to newElement. Returns the element previously at that position, but the method is often used as if it were a void method. If you draw an analogy between the ArrayList and an array *a*, this statement is analogous to setting *a*[index] to the value newElement. The index must be a value greater than or equal to 0 and less than the current size of the ArrayList. Throws an IndexOutOfBoundsException if the index is not in this range.

```
public Base_Type get(int index)
```

Returns the element at the specified index. This statement is analogous to returning *a*[index] for an array *a*. The index must be a value greater than or equal to 0 and less than the current size of the ArrayList. Throws IndexOutOfBoundsException if the index is not in this range.

(continued)

3



# ● ArrayList Class

## Display 14.1 Some Methods in the Class ArrayList

### METHODS TO ADD ELEMENTS

```
public boolean add(Base_Type newElement)
```

Adds the specified element to the end of the calling ArrayList and increases the ArrayList's size by one. The capacity of the ArrayList is increased if that is required. Returns `true` if the add was successful. (The return type is `boolean`, but the method is typically used as if it were a void method.)

```
public void add( int index, Base_Type newElement)
```

Inserts `newElement` as an element in the calling ArrayList at the specified index. Each element in the ArrayList with an index greater or equal to `index` is shifted upward to have an index that is one greater than the value it had previously. The index must be a value greater than or equal to 0 and less than *or equal* to the current size of the ArrayList. Throws `IndexOutOfBoundsException` if the index is not in this range. Note that you can use this method to add an element after the last element. The capacity of the ArrayList is increased if that is required.

(continued)

B

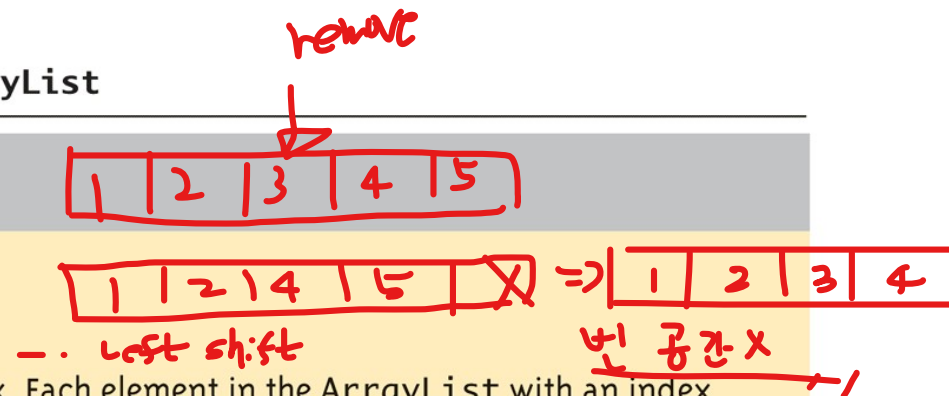
# ArrayList Class

## Display 14.1 Some Methods in the Class ArrayList

### METHODS TO REMOVE ELEMENTS

```
public Base_Type remove(int index)
```

Deletes and returns the element at the specified index. Each element in the ArrayList with an index greater than index is decreased to have an index that is one less than the value it had previously. The index must be a value greater than or equal to 0 and less than the current size of the ArrayList. Throws `IndexOutOfBoundsException` if the index is not in this range. Often used as if it were a void method.



(continued)

CSLAB

# ● ArrayList Class

## Display 14.1 Some Methods in the Class ArrayList

```
protected void removeRange(int fromIndex, int toIndex)
```

Deletes all the element with indices  $i$  such that  $\text{fromIndex} \leq i < \text{toIndex}$ . Element with indices greater than or equal to  $\text{toIndex}$  are decreased appropriately.

```
public boolean remove(Object theElement)
```

Removes one occurrence of `theElement` from the calling `ArrayList`. If `theElement` is found in the `ArrayList`, then each element in the `ArrayList` with an index greater than the removed element's index is decreased to have an index that is one less than the value it had previously. Returns `true` if `theElement` was found (and removed). Returns `false` if `theElement` was not found in the calling `ArrayList`.

```
public void clear()
```

Removes all elements from the calling `ArrayList` and sets the `ArrayList`'s size to zero.

(continued)

CSLAB



# ● ArrayList Class

## Display 14.1 Some Methods in the Class ArrayList

### SEARCH METHODS

```
public boolean contains(Object target)
```

Returns true if the calling ArrayList contains target; otherwise, returns false. Uses the method equals of the object target to test for equality with any element in the calling ArrayList.

```
public int indexOf(Object target)
```

Returns the index of the first element that is equal to target. Uses the method equals of the object target to test for equality. Returns -1 if target is not found.

CSLAB

# • ArrayList Class

## Display 14.1 Some Methods in the Class ArrayList

### MEMORY MANAGEMENT (SIZE AND CAPACITY)

```
public boolean isEmpty()
```

Returns true if the calling ArrayList is empty (that is, has size 0); otherwise, returns false.

```
public int size()
```

Returns the number of elements in the calling ArrayList.

# CSLAB

# ● Golf Score Program (Part 1 of 6)

## Display 14.3 Golf Score Program

```
1  import java.util.ArrayList;
2  import java.util.Scanner;

3  public class GolfScores
4  {
5      /**
6       Shows differences between each of a list of golf scores and their average.
7       */
8      public static void main(String[] args)
9      {
10         ArrayList<Double> score = new ArrayList<Double>();
11         System.out.println("This program reads golf scores and shows");
12         System.out.println("how much each differs from the average.");

13         System.out.println("Enter golf scores:");
14         fillArrayList(score);
15         showDifference(score);
16     }
```

*Parameters of type ArrayList<Double>() are handled just like any other class parameter.*

(continued)

CULAB

## ● Golf Score Program (Part 2 of 6)

Display 14.3 Golf Score Program

```
17    /**
18     Reads values into the array a.
19     */
20    public static void fillArrayList(ArrayList<Double> a)
21    {
22        System.out.println("Enter a list of nonnegative numbers.");
23        System.out.println("Mark the end of the list with a negative number.");
24        Scanner keyboard = new Scanner(System.in);
```

(continued)

CSLAB

# ● Golf Score Program (Part 3 of 6)

Display 14.3 Golf Score Program

```
25  primitive double next;  
26  int index = 0;  
27  next = keyboard.nextDouble();  
28  while (next >= 0)  
29  {  
30      a.add(next);  
31      next = keyboard.nextDouble();  
32  }  
33  }  
34  /**  
35   Returns the average of numbers in a.  
36   */  
37  public static double computeAverage(ArrayList<Double> a)  
38  {  
39      double total = 0;  
40      for (Double element : a)  
41          total = total + element;
```

*Because of automatic boxing, we can treat values of type `double` as if their type were `Double`.*

*A for-each loop is the nicest way to cycle through all the elements in an `ArrayList`.*

(continued)

LAB

## ● Golf Score Program (Part 4 of 6)

Display 14.3 Golf Score Program

```
42     int numberOfScores = a.size();
43     if (numberOfScores > 0)
44     {
45         return (total/numberOfScores);
46     }
47     else
48     {
49         System.out.println("ERROR: Trying to average 0 numbers.");
50         System.out.println("computeAverage returns 0.");
51         return 0;
52     }
53 }
```

(continued)

CSLAB

# ● Golf Score Program (Part 5 of 6)

## Display 14.3 Golf Score Program

```
54     /**
55      * Gives screen output showing how much each of the elements
56      * in a differ from their average.
57      */
58     public static void showDifference(ArrayList<Double> a)
59     {
60         double average = computeAverage(a);
61         System.out.println("Average of the " + a.size()
62                             + " scores = " + average);
63         System.out.println("The scores are:");
64         for (Double element : a)
65             System.out.println(element + " differs from average by "
66                                 + (element - average));
67     }
68 }
```

(continued)

CSLAB



# ● Golf Score Program (Part 6 of 6)

## Display 14.3 Golf Score Program

### SAMPLE DIALOGUE

This program reads golf scores and shows how much each differs from the average.  
Enter golf scores:  
Enter a list of nonnegative numbers.  
Mark the end of the list with a negative number.  
**69 74 68 -1**  
Average of the 3 scores = 70.3333  
The scores are:  
69.0 differs from average by -1.33333  
74.0 differs from average by 3.66667  
68.0 differs from average by -2.33333

CSLAB



## ● For each loop

- As with arrays, the For-each loop can be used to cycle through (iterate) all the elements in an collection (like an ArrayList)
  - *for(variable : collection) {  
    Statements;  
}*

CSLAB

# ● For each loop

```
import java.util.ArrayList;

public class ArrayListTest {
    public static void main(String[] args){
        //create the ArrayList
        ArrayList<Integer> aList = new ArrayList<Integer>();

        //Load objects into the list
        for(int i = 0; i < 20; i++){
            aList.add(new Integer(i));
        }
        System.out.println("Loaded integers into list.");

        //using the the for-each loop data can be retrieved
        for (Integer itger:aList){
            System.out.println(itger.intValue());
        }
    }
}
```

Loaded integers into list.

0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19

for each 는 size 를 알아야 사용 가능.  
초기값 설정에서 size 를 보려고 노력,

Why is the **for-each** loop **not** used here?

# ● ArrayList Iterator

- An iterator is an object that is used with a collection to provide **sequential access** to the collection's elements
- This access allows examination and possible modification of the elements

CSLAB

# • ArrayList Iterator

- Java provides an `Iterator<T>` interface
  - Any object of any class that satisfies the `Iterator<T>` interface is an `Iterator<T>`
- An `Iterator<T>` does not stand on its own
- It must be associated with some collection object using the method `iterator` {usually inner classes of collections}

CSLAB

# ● ArrayList Iterator

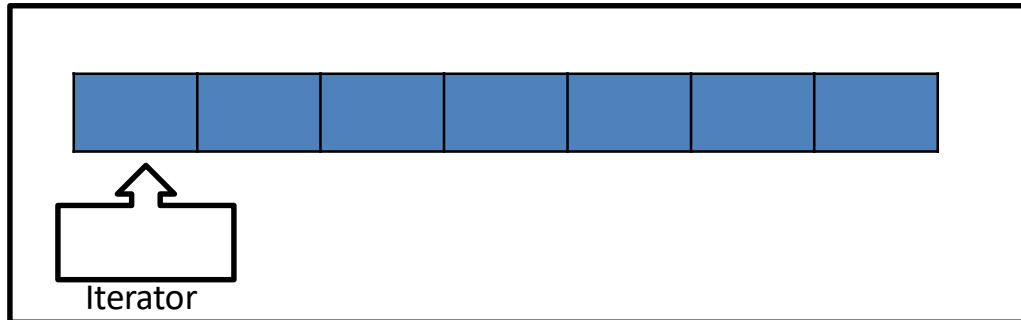
- Iterators know how to iterate the internal data structure.
  - For ArrayLists the iterator knows how to iterate the list.
- To use an Iterator you must obtain it from the ArrayList.
- *Iterator iterator = myArrayList.iterator();*



☐ Obtain an iterator over the ArrayList named myArrayList

# CSLAB

# ArrayList Iterator



- Iterators provide methods for traversing the internal structure:
  - *Object next()*
    - Returns the next item from the internal structure
  - *boolean hasNext()* *next value 유무*
    - Returns true if there are more items in the structure
  - *void remove()*
    - Removes the current item from the structure.

CSLAB

# ArrayList Iterator

```

1 import java.util.ArrayList;
2 import java.util.Iterator;
3
4 public class ArrayListTest {
5
6     public static void main(String[] args) {
7         // TODO Auto-generated method stub
8         ArrayList<Integer> aList = new ArrayList<Integer>();
9
10        for(int i=0; i<20; i++)
11            aList.add(new Integer(i));
12
13        System.out.println("Loaded integers into list");
14
15        Iterator iterator = aList.iterator();
16        while(iterator.hasNext()) {
17            Integer itger = (Integer) iterator.next();
18            System.out.println(itger.intValue());
19        }
20    }
21 }

```

Loaded integers into list

0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19

return class

```

1 import java.util.ArrayList;
2 import java.util.Iterator;
3
4 public class ArrayListTest {
5
6     public static void main(String[] args) {
7         // TODO Auto-generated method stub
8         ArrayList<Integer> aList = new ArrayList<Integer>();
9
10        for(int i=0; i<20; i++)
11            aList.add(new Integer(i));
12
13        System.out.println("Loaded integers into list");
14
15        Iterator<Integer> iter;
16        for(iter = aList.iterator(); iter.hasNext(); ) {
17            Integer itger = iter.next();
18            System.out.println(itger.intValue());
19        }
20    }
21 }
22 }

```

generic 으로 선언  
=> type casting 필요

# Iterators

```
public class GenericManager <T>{  
  
    private Collection<T> cList;  
  
    public GenericManager(){  
        cList = new ArrayList<T>();  
    }  
  
    public void add(T item){  
    }  
  
    public String find(T item){  
        String toReturn = "";  
        for(Iterator<T>myIterator = cList.iterator(); myIterator.hasNext();){  
            T listContact = myIterator.next();  
            if (listContact.equals(item))  
                toReturn +=listContact+"\n";  
        }  
        return toReturn;  
    }  
}
```

A look at our Generic Manager implemented with an [ArrayList](#). We are using an Iterator to iterate the list.

- ☐ Notice the structure of the for loop is different.
- ☐ This for loop structure is used specifically with iterators to prevent [ConcurrentModificationException](#).

동기화없이도  
여러번 처리 가능.  
수동

동시

수동

LAB



## ● Generics 코드로 다른 type도 사용되게끔.

- It would be nice if we could write a single sort method that could sort the elements in an Integer array, a String array or an array of any type that supports ordering.
- Using Java **Generic** concept we might write a generic method for sorting an array of objects, then invoke the generic method with Integer arrays, Double arrays, String arrays and so on, to sort the array elements.
- A way to **re-use** the same code with **different inputs**

CSLAB

# ● Parameterized Classes and Generics

- The class ArrayList is a *parameterized class*
- It has a parameter, denoted by Base Type, that can be replaced by any reference type to obtain a class for ArrayLists with the specified base type

*ArrayList<BaseType> aList = new ArrayList<BaseType>();*

- Starting with version 5.0, Java allows class definitions with parameters for types
  - These classes that have type parameters are called *parameterized class* or *generic definitions*, or simply, generics

CSLAB

# ● Generics Class

- A class definition with a type parameter is called a generic class
  - The type parameter is included in angular brackets after the class name in the class definition heading
  - The type parameter can be used like other types used in the definition of a class

```
Public class myClass<T> {  
    // the generic type is used in here  
}
```

- The type plugged in for a type parameter must always be a reference type
  - It cannot be a primitive type such as **int**, **double**, or **char**

CSLAB

# ● Defining Generic Class

Display 14.4 A Class Definition with a Type Parameter

```
1 public class Sample<T>
2 {
3     private T data;
4
5     public void setData(T newData)
6     {
7         data = newData;
8     }
9
10    public T getData()
11    {
12        return data;
13    }
14 }
```

*T is a parameter for a type.*

☐ Constructor does not have a parameter

CSLAB

## ● Using Generic Class

- To use a generic class you must
  - Plug the type of the parameter.
  - This is done by using < >.
- e.g. if we would like to create a sample object from previous slide we can do this.
  - `Sample<String> object = new Sample<String>();`

CSLAB

## ● Generic Class

- It is possible to use multiple generic parameters in a single class.  
This may be done by listing the parameters.

인스턴스는 받을 수 있지만

특정 타입에 대한 연산은 필요로  
하니까 쓸때,

```
public class myClass <T, U> {
```

*// the generic types are used in here*

```
}
```

CSLAB

## ● Bounds with Generic Class

제한

- You can have Java enforce this restriction on the possible types that can be plugged in for T.
- With a regular Type we can simply
  - extend a base Type
  - implement an interface.
- **Generics** can be made to conform by using the keyword **extends**.

When **extends** is used the generic becomes bound to a particular

- interface ↳ 여기서 상속은 모든 부모의 type은 받든지 아닌,
- parent class extend [class name]  
↳ 여기서 명시된 class의 parameter type으로 받는다.

## ● Bounds with Generic Class

```
Public class myClass <T extends Comparable>{
    // this forces T to be only Comparable types
}
```

- You may also bind the generic to multiple types however as with regular types you can

- Be bound to multiple interfaces
- Be bound to only one class

```
Public class myClass <T extends Employee & Comparable & Iterable>{
    // this forces T to be only Comparable types and
    // T must be a descendant (sub-class) of Employee
    // The extended class (Employee) must be listed first.
}
```

상속된 class 지워  
그걸 class만

다중 interface 가능.

class 먼저 나열함 그 후 interface



## • Generic Methods

- When a generic class is defined, the type parameter can be used in the definitions of the methods for that generic class
- In addition, a generic method can be defined that has its own type parameter that is not the type parameter of any class
  - A generic method can be a member of an ordinary class or a member of a generic class that has some other type parameter
  - The type parameter of a generic method is local to that method, not to the class

CSLAB

## ● Generic Methods

- The type parameter must be placed (in angular brackets) after all the modifiers, and before the returned type

*public static <T> T genMethod(T[] a)*

☐ Accepts T array as a parameter and returns type T

- When one of these generic methods is invoked, the method name is prefaced with the type to be plugged in, enclosed in angular brackets

*String s = NonGenClass.<String>genMethod(c);*

CSLAB

## ● Generics

```
public class Utility{  
    //...  
  
    public static <T> T getMidpoint(T[] a){  
        return a[a.length/2];  
    }  
  
    public static <T> T getFirst(T[] a){  
        return a[0];  
    }  
    //...  
}  
  
String midString = Utility.<String>getMidpoint(b);  
double firstNumber = Utility.<Double>getFirst(c);
```

# ● Self-Test

- 프로젝트 명: Project12\_1
  - `git commit -m "Project12_1"`
- 고객과의 상담 정보를 관리하는 Contact management 프로그램을 작성할 것
- Contact 클래스를 작성할 것. 이 클래스는 Comparable을 implements하며, String name, String telNum, String email 3개의 instance variable을 갖는다.
- 두 개의 생성자를 만들 것
  - 첫번째 생성자는 모든 instance variable을 인자로 받는다.
  - 두번째 생성자는 name을 인자로 받으며 나머지 변수는 null로 설정한다.
- 매개변수를 적절히 사용하여 accessor와 mutator 메소드를 작성할 것 (`getTelNum()`, `setTelNum()`, `getEmail()`, `setEmail()`, `getName()`, `setName()`)
- name과 telNum, email이 포함된 문자열을 반환하는 `toString()` (작성되어 있음)
- name이 같은지 비교하는 `equals()` 메소드를 작성할 것
- name을 알파벳 오름차순으로 비교하는 `compareTo()` 메소드 (작성되어 있음)

## ● Self-Test (Contd)

- GenericManager 클래스를 작성할 것. 이 클래스는 generic 매개변수 T를 갖는다. generic 매개변수 타입 T는 Comparable로 bound 된다.
- 이 클래스는 ArrayList cList라는 instance variable 갖는다. 이 ArrayList의 base type은 T이다.
- 매개변수를 받지 않는 생성자를 만들 것. 생성자는 새로운 ArrayList 객체를 만든다.
- add() 메소드를 작성할 것. 이 메소드는 generic type을 매개변수로 가지며 이를 ArrayList에 추가한다. 반환 형은 void이다.
- CompareTo() 메소드를 통해 ArrayList를 정렬하는 sort() 메소드 (작성되어 있음)
- find() 메소드를 작성할 것. 이 메소드는 generic type을 매개변수로 가지며 매개변수와 동일한 모든 객체의 문자열을 반환한다.
- toString() 메소드를 작성할 것. 이 메소드는 ArrayList의 각 항목을 모두 거치며 ArrayList 안에 있는 모든 객체의 문자열을 반환한다.

CSLAB

# Self-Test (Contd)

```
*****Contact Management System*****
```

- (1) Add a contact:
- (2) Display all contacts:
- (3) Search for contacts
- (4) Sort the contacts
- (5) Exit the program

```
*****
```

1

Please enter the name:

IronMan

Please enter the tell#:

01000000000

Please enter the email:

Rich@gmail.com

```
*****Contact Management System*****
```

- (1) Add a contact:
- (2) Display all contacts:
- (3) Search for contacts
- (4) Sort the contacts
- (5) Exit the program

```
*****
```

1

Please enter the name:

Thor

Please enter the tell#:

01011111111

Please enter the email:

KingofAsgard@gmail.com

```
*****Contact Management System*****
```

- (1) Add a contact:
- (2) Display all contacts:
- (3) Search for contacts
- (4) Sort the contacts
- (5) Exit the program

```
*****
```

1

Please enter the name:

DoctorStrange

Please enter the tell#:

01022222222

Please enter the email:

Dormammu@gmail.com

```
*****Contact Management System*****
```

- (1) Add a contact:
- (2) Display all contacts:
- (3) Search for contacts
- (4) Sort the contacts
- (5) Exit the program

```
*****
```

1

Please enter the name:

Thanos

Please enter the tell#:

01033333333

Please enter the email:

Balance@gmail.com

CSLAB

# ● Self-Test (Contd)

```
*****Contact Management System*****
```

- (1) Add a contact:
- (2) Display all contacts:
- (3) Search for contacts
- (4) Sort the contacts
- (5) Exit the program

```
*****
```

```
2
*****Contact List*****
```

```
Name: IronMan   telNum: 01000000000   email: Rich@gmail.com
Name: Thor      telNum: 01011111111   email: KingofAsgard@gmail.com
Name: DoctorStrange telNum: 01022222222   email: Dormammu@gmail.com
Name: Thanos    telNum: 01033333333   email: Balance@gmail.com
```

```
*****
```

```
*****Contact Management System*****
```

- (1) Add a contact:
- (2) Display all contacts:
- (3) Search for contacts
- (4) Sort the contacts
- (5) Exit the program

```
*****
```

```
3
```

```
Please enter the name to search for:|
```

```
IronMan
```

```
*****Search Results*****
```

```
Name: IronMan   telNum: 01000000000   email: Rich@gmail.com
```

```
*****
```

```
*****Contact Management System*****
```

- (1) Add a contact:
- (2) Display all contacts:
- (3) Search for contacts
- (4) Sort the contacts
- (5) Exit the program

```
*****
```

```
4
```

```
*****Contact Management System*****
```

- (1) Add a contact:
- (2) Display all contacts:
- (3) Search for contacts
- (4) Sort the contacts
- (5) Exit the program

```
*****
```

```
2
```

```
*****Contact List*****
```

```
Name: DoctorStrange telNum: 01022222222   email: Dormammu@gmail.com
Name: IronMan        telNum: 01000000000   email: Rich@gmail.com
Name: Thanos         telNum: 01033333333   email: Balance@gmail.com
Name: Thor           telNum: 01011111111   email: KingofAsgard@gmail.com
```

```
*****
```

```
*****Contact Management System*****
```

- (1) Add a contact:
- (2) Display all contacts:
- (3) Search for contacts
- (4) Sort the contacts
- (5) Exit the program

```
*****
```

```
5
```

```
Exiting Program... Goodbye.
```

