National University of Computer & Emerging Sciences, Karachi
Computer Science Department
Fall 2022, Lab Manual – 02

Course Code: CL-2001	**Course : Data Structures - Lab**
Instructor(s) :	**Abeer Gauher, Sobia Iftikhar**

LAB - 2

Writing & using dynamic safe arrays & Basic

Recursion IN JAVA

Arrays

There are three behavioral differences between an array in C++ and a similar array in Java.

1. A Java array is aware of its size while a C++ array is not
2. Java detects and notifies a programmer when an array is indexed out of bounds C++ does not
3. In C++ it is possible to create an array of fully constructed objects in a single operation; Java requires multiple operations to create an array of objects.

```
int[]   scores = new int[10];

for (int i = 0; i < scores.length; i++)
        . . . scores[i] . . .;
```

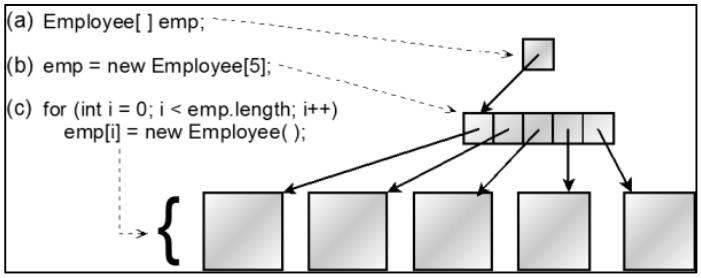
Java

```
const int size = 10;
int scores[size];
for (int i = 0; i < size; i++)
        . . . scores[i] . . .;
```

C++

Java arrays. *Representing the size of an array with a variable makes it easier to maintain the code: if a programmer changes the "10" in either example, the size of the array and the associated code automatically change throughout the program without additional effort. In a Java program, the size of the array is stored with the array itself in the* ***length*** *instance field by the instantiation operation, but C++ requires a separate variable.*

Creating an array of objects in Java



1. Defines the array variable, which only allocates enough memory to hold an address
2. The array is created with the "new" operator; new always returns an address or pointer; steps (a) and (b) can often be combined
3. Each object is created or instantiated one at a time with the "new" operator and the address of each object is stored in the array

Option 1: dataType[] arr;
Option 2: dataType []arr;
Option 3: dataType arr[];

Multidimensional Array

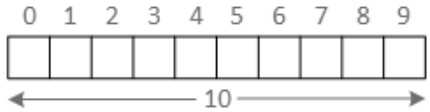
In such case, data is stored in row and column based index (also known as matrix form).

int[][] a = **new int**[3][4]; It is a 2-dimensional array, that can hold a maximum of 12 elements,

```
// create a 3d array
    int[][][] test = {
      {
        {1, -2, 3},
        {2, 3, 4}
      },
      {
        {-4, -5, 6, 9},
        {1},
        {2, 3}
      }
    };
    for (int[][] array2D: test) {
      for (int[] array1D: array2D) {
        for(int item: array1D) {
          System.out.println(item);
        }
      }
    }
```

Bounds Checking

Arrays, in both the C++ and Java programming languages, are zero-indexed, meaning that legal index values range from 0 to size - 1 (where size is the maximum number of elements in the array).



An array with a size or length of 10. The first element is located at index location 0 and the last element at index 9. Using an index value less than 0 or greater than 9 with this array is an *indexing out of bounds* error. C++ does not detect such errors, but the Java virtual machine (JVM) does and aborts the program by throwing an IndexOutOfBoundsException.

In c++ needs to write the code for bound checking

```
void set(int pos, Element val){      //set method
        if (pos<0 || pos>=size){
                cout<<"Boundary Error\n";
         }
         else{
                Array[pos] = val;
         }
}
```

```
public class Example {

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

        int array1[] = new int[10];
        for (int i = 0; i <= 10; i++)   // index out of bounds
            array1[i] = 25;
    }

}
```

Problems Javadoc Declaration Console Progress Properties Model Validation References

<terminated> Example [Java Application] C:\Program Files\Java\jre1.8.0_281\bin\javaw.exe (Aug 17, 2022 1:39:18 PM – 1:39:19 PM)

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 10
        at Example.main(Example.java:12)
```

Array List (Dynamic Array)

In Java, we need to declare the size of an array before we can use it. Once the size of an array is declared, it's hard to change it. To handle this issue, we can use the array list class. It allows us to create resizable arrays.

Creating an ArrayList: Required Library [`java.util.ArrayList`]

```
ArrayList<Type> arrayList= new ArrayList<>();
```

Example:

```
import java.util.ArrayList;
public class Example {
        public static void main(String[] args) {
                // TODO Auto-generated method stub
                        System.out.println("hey from Ds");
                        ArrayList<String> languages= new ArrayList<>();
                        languages.add("ALI");
                    languages.add("Ahmed");
                    languages.add("Asif");
                    System.out.println("ArrayList: " + languages);
        }
```

Array List Operation

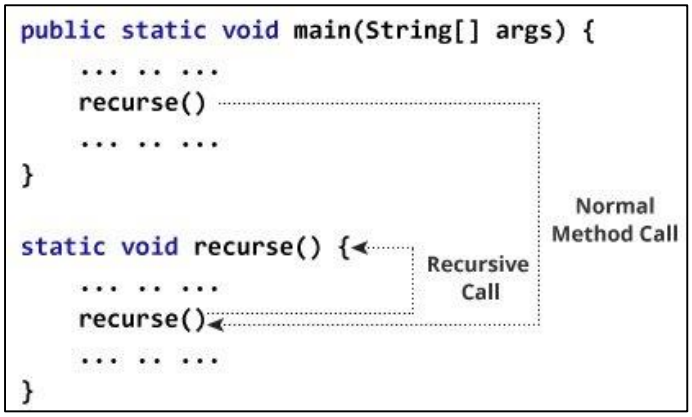
Basic Operation:

Add elements	add(value); to add a single element
Access elements	Get(index); to access the element
Change elements	Set(index, value); to change the element of array
Remove elements	Remove(index); to delete the value from array
Check empty array list	IsEmpty();
Return the position of speechified value	indexOf();
Ensure the capacity. resize	ensureCapacity();

Recursion

In Java, a method that calls itself is known as a recursive method. And, this process is known as recursion.

In order to stop the recursive call, we need to provide some conditions inside the method (Base Condition). Otherwise, the method will be called infinitely



Example

```
class Factorial {

   static int factorial( int n ) {
      if (n != 0)  // termination condition
         return n * factorial(n-1); // recursive call
      else
         return 1;
   }
```

```
   public static void main(String[] args) {
      int number = 4, result;
      result = factorial(number);
      System.out.println(number + " factorial = "
+ result);
   }
}
```

Task 01

Create a program that has two arraylists of Strings (you can decide if you define these in your code directly or if the user should enter the strings).
Now create a new arraylist from these two in the following way:

- take as first value for the new list the first value from the first list
- take as next value for the new list the first value from the second list
- take as next value for the new list the second value from the first list …

until all values from both lists are in the new list. (start with two lists of the same size => if you have a solution, extend it by handling lists of different sizes => as soon as one source list has no value anymore, just add all remaining values from the other list)
Example: Given two lists of same size, list 1 being [“a”, “b”, “c”] list 2 being [“x”, “y”, “z”] resulting list: [“a”,”x”,”b”,”y”,”c”,”z”]
Example: Given two lists of different size list 1 being [“a”, “b”, “c”, “x”] list 2 being [“y”, “z”] [“a”,”y”,”b”,”z”,”c”,”x”]

Task 02

A teacher has a class of 10 students. Each student is numbered from 1 to 10. The test scores are stored in a 2D array where each column holds the grades for each test. You can take the number of tests as your choice. The teacher enters the student number, test number and the program prints the grade for the particular test. You can use your own grade chart.

Task 03

Write a java program to rearrange a given sorted array of positive integers . Note: In the final array, first element should be maximum value, second minimum value, third second maximum value , fourth second minimum value, fifth third maximum and so on.

Task 04:

a. Write a Java Program to Find Reverse of a Number Using Recursion. 1654 rev = 4561
b. Suppose you're given an integer array A of n integers. You want to figure out the number of duplicates (an integer that appears more than once) in Array.
    a. Suppose Array [] a = {2,4,6,5,3,2,6}
c. Write a recursive function that returns the sum of the digits of a positive integer. For example: SumOfDigits(int x) when x=123 will return 1+2+3=6.

Task 05

- Let the user choose how many words they want to input
- Read that many words
- Let the user choose a number bigger than 0
- Remove all words from the ArrayList that are shorter than the given number.

  Example Input:
  4 // four words
  Ananas
  Zimtzitroneneis
  Apple
  Zartbitterschokolade
  6 // remove all shorter than six

  Output:
  Zimtzitroneneis
  Zartbitterschokolade