RIZAL TECHNOLOGICAL UNIVERSITY
COLLEGE OF ENGINEERING ARCHITECTURE AND TECHNOLOGY

DATA STRUCTURE AND ALGORITHM

ARRAY

- Operations in Array
- Arrays in Java
- Dimensionality of Arrays

Objectives:

- Define what array is.
- Enumerate operations on arrays
- Define the basics of arrays in Java
- Specify the dimensionalities of arrays

Let's define

a **list** is an ordered set consisting of a variable number of elements or a collection of related records to which additions and deletions may be made, if applicable.

a **linear list**

- is a finite sequence of simple data items or records.
- is commonly represented by an array

It is either empty or it can be written as:

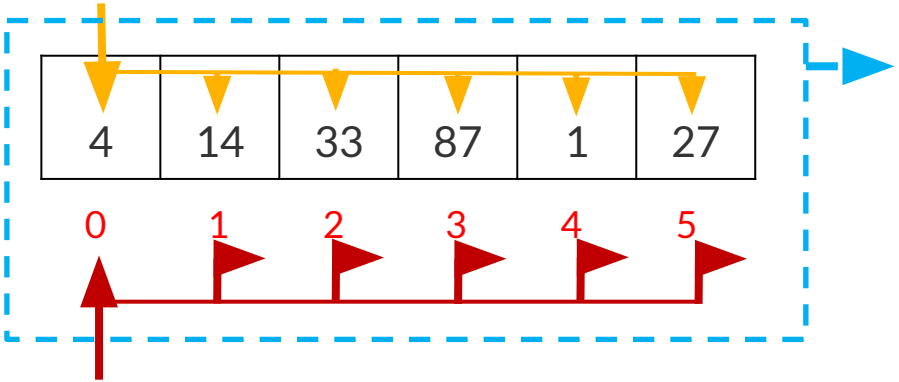
$(a_1, a_2, a_3, \ldots, a_n)$

where a_1 to a_n are elements of some set S

ARRAY

- is the most commonly used data storage, built into most programming languages.
- ordered collection of data items of the same type referred to collectively by a single name

Element individual data / items in an array



Length size of an array; must be specified by an int value and not long or short

Index each variable or cell in an array

Operations in Array

0	04
1	07
2	11
3	14
4	17
5	
6	
7	
8	
10	
11	

Insert

- add items to the indicated place/cell
- the process is very fast because it only includes one step
- every data inserted is placed in the first vacant cell

Search

- another process that the algorithm carries out
- moves methodically downwards to search for the item

Delete

- the algorithm must first locate the item to delete it
- the deleted item causes hole in the array

Hole – one or more cells that have filled cells above them

Arrays in Java

- are dynamically allocated
- since arrays are objects in Java, we can find the length using the object property *length*
- the variables in the array are ordered and each have an index beginning from 0
- Java array can be also be used as a static field, a local variable or a method parameter
- Array can contain *primitives* (int, char, etc.) as well as *object* (or non-primitive) references of a class depending on the definition of the array

Arrays in Java: Creating, Initializing, and Accessing

An array declaration has two components:

- **element_type** – declares the element type (data type) of the array
- **array_name** – reference to an array

```
//CREATING

element_type  array_name[ ];
OR
element_type[ ]  array_name;
```

```
// INSTANTIATING

array_name = new element_type [length];
```

Example: Arrays in Java

```
int intA[]; //declare a reference to an array

intA = new int[5];  //initialize the array
```

Or, you can also write in single-statement approach:

```
int intA[] = new int[5];  //declaration and instantiation
```

Another alternative syntax would be:

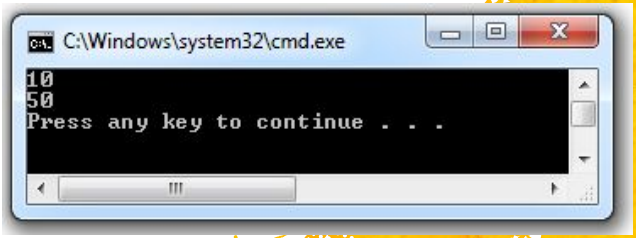
```
int[] intA = new int[5];  //declaration and instantiation
```

Example: Arrays in Java

Each element in the array is accessed via its **index**.

```java
class arrBasic
{
    public static void main(String args[])
    {

        //declaring array literal
        int[] intA = new int[5];  //declaration and instantiation

        intA[0]=10; //initialization
        intA[1]=70;
        intA[2]=40;
        intA[3]=50;
        intA[4]=30;

        System.out.println(intA[0]); //display element at index 0
        System.out.println(intA[3]); //display element at index 3
    }
}
```

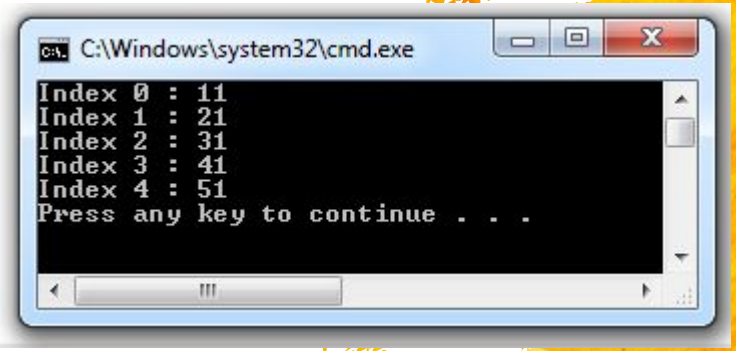
```java
class arrBasic
{
    public static void main(String args[])
    {

        //declaring array literal
        int[] intA = new int[] {10, 70, 40, 50, 30};  //without array length


        System.out.println(intA[0]); //display element at index 0
        System.out.println(intA[3]); //display element at index 3
    }
}
```



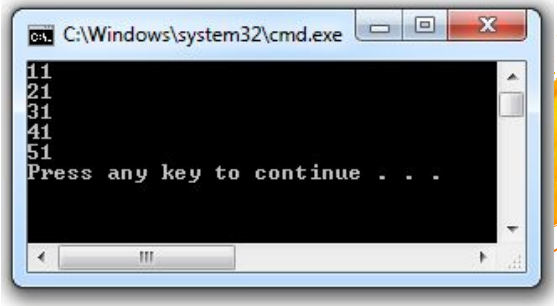
Example: Arrays in Java

Each element in the array is accessed via its for loop.

```
class arrBasicLoop
{
    public static void main(String args[])
    {

        int a[]=new int[] {11, 21, 31, 41, 51};

        //printing array
        for(int i=0;i<a.length;i++)//length is the property of array
        System.out.print("Index " + i + " : " + a[i] + " " + '\n');

    }
}
```



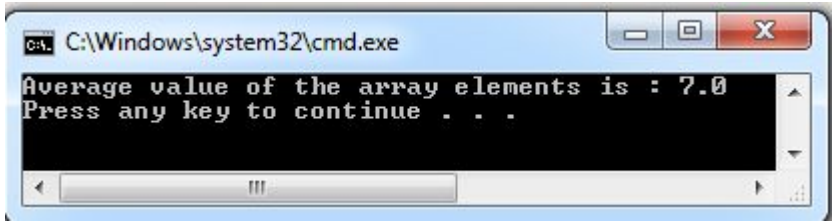

```
class arrBasicLoop
{
    public static void main(String args[])
    {

        int a[]=new int[]{11, 21, 31, 41, 51} ;

        for (int i:a) {
            System.out.print(i + " " + '\n');
        }
    }
}
```



Example: Arrays in Java

Calculate the average value of all array elements

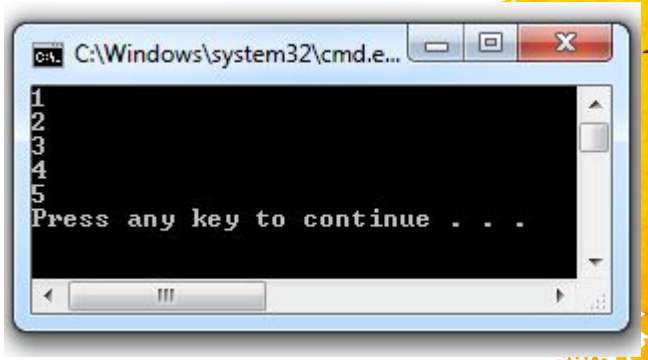
```
public class arrAve
{
    public static void main(String[] args) {

          int[] numbers = new int[]{20, 30, 25, 35, -16, 60, -100};
            //calculate sum of all array elements

        int sum = 0;

        for(int i=0; i < numbers.length ; i++)
            sum = sum + numbers[i];
            //calculate average value

        double average = sum / numbers.length;

            System.out.println("Average value of the array elements is : " + average);
    }
}
```



Example: Arrays in Java

Arrays.sort

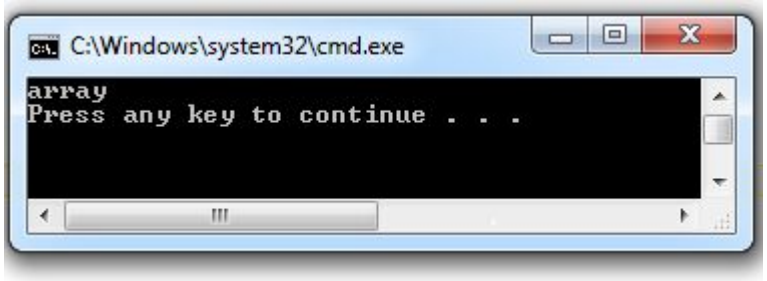
```
import java.util.Arrays;

public class courseSort {

    public static void main(String[]args) {

    int[] course = {3, 2, 4, 1, 5};
    Arrays.sort(course);

        for (int i=0; i<course.length; i++) {
            System.out.println(course[i]);
        }
    }
}
```



Example: Arrays in Java

arraycopy

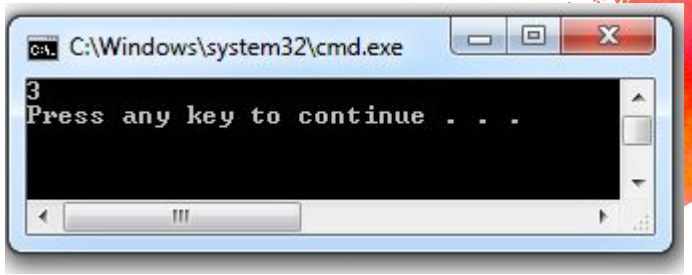
```
import java.util.*;

class arrCopy {

    public static void main(String[]args) {

    char[] copyFrom = {'e', 'g', 'a', 'r', 'r', 'a', 'y', 'i'};

    char[] copyTo = new char[14];

    System.arraycopy(copyFrom, 2, copyTo, 0,5);
    System.out.println(new String(copyTo));

    }
}
```



Example: Arrays in Java

Finding the minimum number among elements.

```
class arrMin
{
    static void min(int arr[])
    {
        int min = arr[0];

        for (int i=0; i<arr.length; i++)
            if (min>arr[i])
            min = arr[i];

        System.out.println(min);
    }

    public static void main(String[] args)
    {
        int a[] = {33, 3, 8, 9};
        min (a);
    }
}
```



Example: Arrays in Java

```java
class Student
{
    public int studnum;
    public String studname;
    Student(int studnum, String studname)
    {
        this.studnum = studnum;
        this.studname = studname;
    }
}
```

```java
// Elements of the array are objects of a class Student.
public class StudentInfo
{
    public static void main (String[] args)
    {
        // declares an Array of integers.
        Student[] arr;

        // allocating memory for 5 objects of type Student.
        arr = new Student[5];

        // initialize the first elements of the array
        arr[0] = new Student(2017201,"Matthew");

        // initialize the second elements of the array
        arr[1] = new Student(2017404,"Mark");

        // so on...
        arr[2] = new Student(2017585,"Luke");
        arr[3] = new Student(2017990,"John");
        arr[4] = new Student(2017304,"Paul");

        // accessing the elements of the specified array
        for (int i = 0; i < arr.length; i++)
            System.out.println("Element at " + i + " : " +
                        arr[i].studnum +" "+ arr[i].studname);
    }
}
```

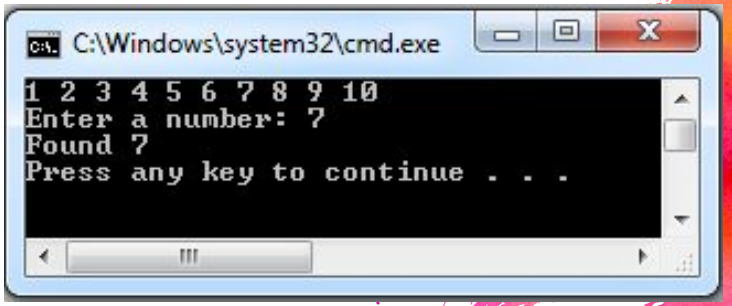
C:\Windows\system32\cmd.exe

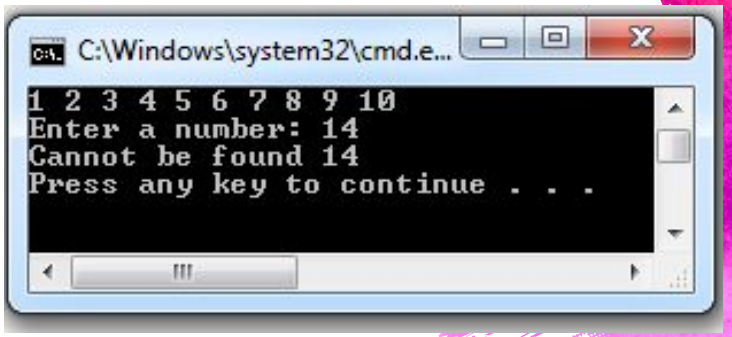
```
Element at 0 : 2017201 Matthew
Element at 1 : 2017404 Mark
Element at 2 : 2017585 Luke
Element at 3 : 2017990 John
Element at 4 : 2017304 Paul
Press any key to continue . . .
```

Example: Arrays in Java

The program prompts the user to input a number to search for and prints a message “Found [search key]” if the number exists in array elements else “Not Found [search key]”

```java
import java.util.Scanner;

 public class arrForCantF {

    public static void main(String[]args){

    int[]arr;
    arr = new int[] {1,2,3,4,5,6,7,8,9,10};
    int nItems = 0;
    int j;
    int searchId;
    nItems = 10;

    Scanner input = new Scanner(System.in);

    for (j=0; j<nItems; j++)
        System.out.print(arr[j]+ " ");
        System.out.println(" ");

        System.out.print("Enter a number:" + " ");
        searchId = input.nextInt();

    for (j=0; j<nItems; j++)
        if (arr[j] == searchId)
            break;
        if(j==nItems)
            System.out.println("Cant Find" + " " + searchId);
        else
            System.out.println("Found" + " " + searchId);
    }
 }
```





Dimensionality of Arrays

- an integer from 1-n called dimensioned variables

- One-dimensional Array (1D)
  arraynamc [j]

- *Two-dimcnsional Array (2D)*
  arraynamc [i] [j]

- *Multi-dimcnsional Array*
  arraynamc [i] [j] [k] .....

One-Dimensional Array

- it is the basic array
- a vertical table number of columns and only one row

  element_type[ ] array_name;

- an array of ints

  int[ ] intArray;

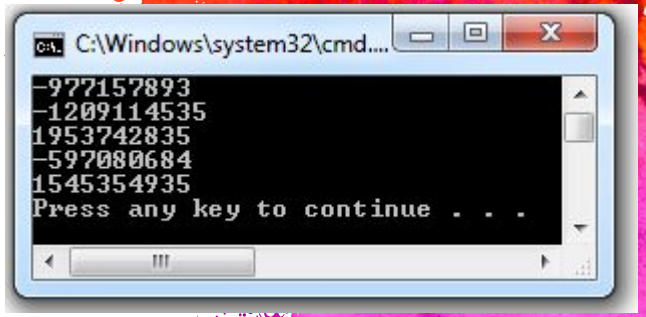
- Arrays are fixed-length structure

- Attempting to access an index which is out of range, gives you ArrayIndexOutOfBoundsException

One-Dimensional Array

- an array of random objects
  Random[ ] RandomArray;

```java
import java.util.Random;

public class arrRandom {

   public static void main(String[] args) {

      Random rd = new Random(); // creating Random object
      int[] arr = new int[5];

      for (int i = 0; i < arr.length; i++) {
         arr[i] = rd.nextInt(); // storing random integers in an array
         System.out.println(arr[i]); // printing each array element
      }
   }
}
```



Two-Dimensional Array

- It is an array of an array, the simplest form of multidimensional array
- Also called as tables or matrices
- Row – first dimension
- Column – second dimension

```
data_type[][] array_name = new data_type[r][c];
```

Example:

```
int[][] arrSample = new int[3][4];
```

Illustration:

	column0	column1	column2	column4
Row0	1	12	3	44
Row1	58	56	17	98
Row2	29	10	31	11

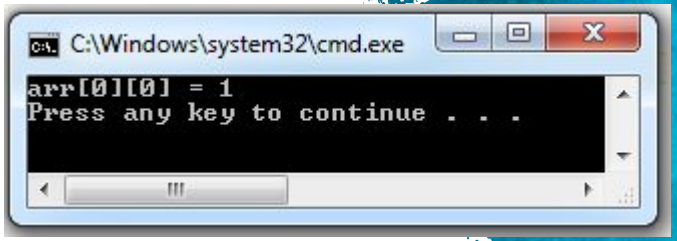
Two-Dimensional Array

- Declaration and instantiation of 2D array is similar to that 1D array

Syntax: array_name [row_index][column_index] = value;

```
int rows = 3, cols = 4;
int[][] tableArray = new int[rows][cols];

for (int rowIndex = 0; rowIndex < rows; rowIndex ++)
{
        for (int colIndex = 0; colIndex < cols; colIndex ++)
        {
                tableArray[rowIndex][colIndex] = rowIndex * colIndex + 1; }
}
```

```
class twoDarray {
    public static void main(String[] args)
    {

        int[][] arr = new int[10][20];
        arr[0][0] = 1;

        System.out.println("arr[0][0] = " + arr[0][0]);
    }
}
```



Two-Dimensional Array

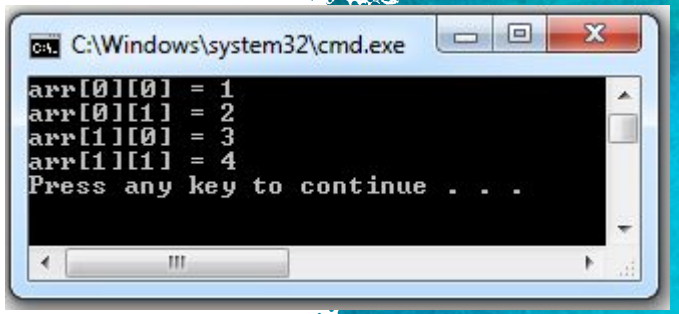
- Curly braces is the alternative constructor for 2D arrays

```
int[][] tableArray = { {2,3,4,5}, {7,8,9,10} };
```

- For a more look like a table

```
int[][] tableArray = { {2,3,4,5},
                       {7,8,9,1} };
```

```
class twoDarrayCurly {
    public static void main(String[] args)
    {

        int[][] arr = { { 1, 2 }, { 3, 4 } };

        for (int i = 0; i < 2; i++)
            for (int j = 0; j < 2; j++)
                System.out.println("arr[" + i + "][" + j + "] = "
                                   + arr[i][j]);
    }
}
```



Two-Dimensional Array

- Matrix is a mathematical object which arises in many physical problems
  -> consists of m rows and n columns

Illustration:

	Column 0	Column 1	Column 2
Row 0	x[0][0]	x[0][1]	x[0][2]
Row 1	x[1][0]	x[1][1]	x[1][2]
Row 2	x[2][0]	x[2][1]	x[2][2]

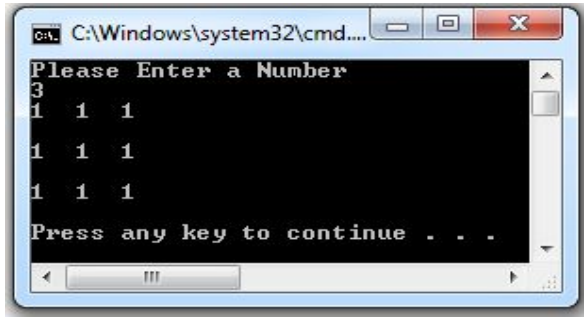
-> m x n (read as m by n) is written to designate a matrix with m rows and n columns

Two-Dimensional Array

- Magic Square is an n x n matrix of the integers 1 to n^2

Algorithm to create a magic square by H. Coxeter

- Begin with 1 in the middle of the top row
- Move up and left assigning numbers in increasing order to empty squares
- If you fall off the square imagine the same square as tilting the plane and continue
- If a square is occupied, move down instead and continue




```java
import java.util.*;

public class magicsquare
{
    public static void main(String args[])
    {
        Scanner x = new Scanner(System.in);
        int DIM;
        System.out.println("Please Enter a Number");
        DIM=x.nextInt();
        int square[][]= new int[DIM][DIM];
        for(int i=0;i<DIM;i++)
        {

            for(int y=0;y<DIM;y++)
            {
                square[i][y]=1;
                System.out.print(square[i][y]+"  ");
            }
            System.out.println();
            System.out.println();
        }
    }
}
```

Three-Dimensional Array

- Complex form of multidimensional array
- Can be seen as an array of two-dimensional array for easier understanding

- Syntax:

```
data_type[][][] array_name = new data_type[i][r][c];
```

Example:

```
int[][][] arrSample = new int[2][3][4];
```

- Initialization:

```
array_name[array_index][row_index][column_index] = value;
```

Example:

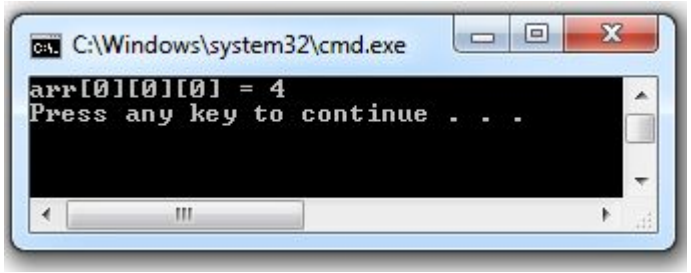
```
arrSample[0][0][0] = 4;
```

Three-Dimensional Array

```
data_type[][][] array_name = {
                              {
                               {valueA1R1C1, valueA1R1C2, ....},
                               {valueA1R2C1, valueA1R2C2, ....}
                              },
                              {
                               {valueA2R1C1, valueA2R1C2, ....},
                               {valueA2R2C1, valueA2R2C2, ....}
                              }
                             };
```

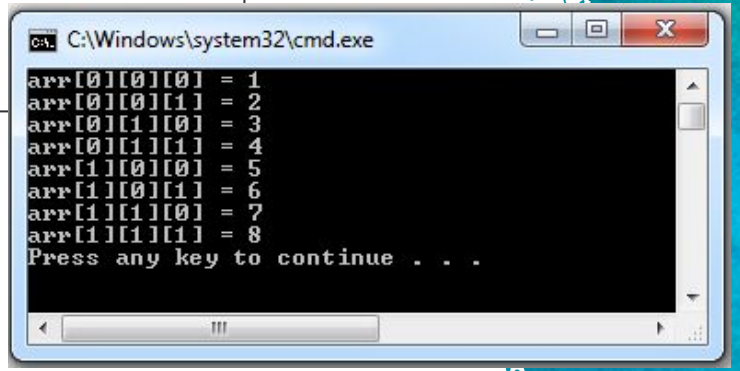
Example: Three-Dimensional Array

```
class threeDarray {
    public static void main(String[] args)
    {

        int[][][] arrSample = new int[10][20][30];
        arrSample[0][0][0] = 4;

        System.out.println("arr[0][0][0] = " + arrSample[0][0][0]);
    }
}
```



Example: Three-Dimensional Array

```
class threeD {
    public static void main(String[] args)
    {

        int[][][] arr = { { { 1, 2 }, { 3, 4 } }, { { 5, 6 }, { 7, 8 } } };

        for (int i = 0; i < 2; i++)
            for (int j = 0; j < 2; j++)
                for (int z = 0; z < 2; z++)
                    System.out.println("arr[" + i
                                       + "]["
                                       + j + "]["
                                       + z + "] = "
                                       + arr[i][j][z]);
    }
}
```



RIZAL TECHNOLOGICAL UNIVERSITY
COLLEGE OF ENGINEERING ARCHITECTURE AND TECHNOLOGY

Thank You 😊
Keep safe
and God bless!