**Chapter 10 Arrays**

* Complete Critical Thinking Pg 266 #2-11
* Complete Exercises # 1-5,8,15
* Ch10 Ex4 – complete all parts including the median and mode

**Two Dimensional Array Question – see separate file.**

An array structure is a structure that can store many of the same kind of data together with a single descriptive name.

**Declaring arrays**

An array must be declared and then space allocated for an array.

**Example:**

An array of names is declared allocating 5 spaces for the array from positions 0 to 4.

String[] names;

names = new String[5];

names[0] = "Dave";

names[1] = "Mary";

names[2] = "Joe";

names[3] = "Peter";

names[4] = "Cathy";

Returns the upper boundary of the array

for (int i =0 ; i<names.length; i++){

System.out.println("Name:" + i + " " + names[i]);

}

**Example2:**

Alternate declaration

String[] names = new String[5];

names[0] = "Dave";

names[1] = "Mary";

names[2] = "Joe";

names[3] = "Peter";

names[4] = "Cathy";

**Example3:**

Alternate declaration

String[] names = { "Dave", "Mary", "Joe" , "Peter", "Cathy"};

for (String element: names){

System.out.println("Name: " + element);

}

**Dynamic Arrays using the ArrayList Class**

Elements in an ArrayList are objects – not primitive types.

**Class ArrayList(java.util.ArrayList)**

**Method**

**add(int index, element)** - Inserts element at index position of the array. Existing elements are shifted to the next position up in the array.

**add(element)** - adds element to the end of the array.

**get(int index)** - returns the element at index position in the array.

**indexof(obj)** - returns the index of the first element matching obj using the equals() method of the object’s class to determine equality between the element and the object.

**remove(int index)** - removes the element at index position in the array.

**set(int index, element)** - replaces the element at index position with element.

**size()** - returns the number of elements in the array.

**Example:** Arraylists of primitive data types(Integer, Double) need to be converted from an object to a type when using ArrayLists.

**import java.util.ArrayList;**

**public class Arrays {**

**public static void main(String[] args) {**

ArrayList<String> names = new ArrayList();

ArrayList<Integer> ages = new ArrayList();

String oldest;

names.add("Dave");

ages.add(19);

names.add("Mary");

ages.add(26);

names.add("Peter");

ages.add(29);

names.add("John");

ages.add(22);

oldest = findoldest(names,ages);

System.out.println(oldest); **}**

**public static String findoldest(ArrayList<String> names,ArrayList<Integer> ages){**

int tempage=0, highage=0;

String highname="";

for (int i=0;i<=ages.size()-1;i++){

tempage = ages.get(i);

if (tempage>=highage){

highage = tempage;

highname = names.get(i);

}

}

return(highname + " is the oldest at an age of " + highage);

**}**

**}**

**Two Dimensional Arrays**

An array with two dimensions can be used to represent data that corresponds to a grid.

**Declaration**

-allocates space from 0 – 3 rows and 0 – 2 columns

String[][] mytable = new String[4][3]

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |
| 10 | 11 | 12 |

mytable[0][0] = “1”

mytable[0][1] = “2”

mytable[0][2] = “3”

mytable[1][0] = “4”

mytable[1][1] = “5”

mytable[1][2] = “6”

mytable[2][0] = “7”

mytable[2][1] = “8”

mytable[2][2] = “9”

mytable[3][0] = “10”

mytable[3][1] = “11”

mytable[3][2] = “12”

**Displaying the table**

for(int intR=0;intR<4;intR++){

for(int intC=0;intC<3;intC++){

System.out.print(mytable [intR][intC]);

}

System.out.println();

}

**OR**

String[][] mytable = {{"1","2","3"}, {"4","5","6"},{"7","8","9"},{"10","11","12"}};

for(int intR=0;intR< mytable.length;intR++){ //rows

for(int intC=0;intC< mytable [0].length;intC++){ //columns

System.out.print(mytable [intR][intC]);

}

System.out.println();

}