Arrays

### Exercise10.1

1. An Array consist of an ordered collection of similar items. An array, as a whole, has a single name, and the items in an array are referred to in terms of their position within the array.
2. Using the index of an element in a array.

### Exercise 10.2

1. a. 23 b. 12 c.156
2. JVM checks the values of the subscripts and throws an ArrayIndexOutOfBoundsException if out of bounds.

### Exercise 10.3

1. int[] abc = new int[500];

for (int i = 0; I < abc.length; i++)

system.out.println(abc[i]);

1. for (int I = abc.length-1; I >= 0; i--)

system.out.println(abc[i]);

1. int index= a.length;

for(int i= a.length-1; I>= 0; i--){

if(a[i]<0)

index=I;

}

1. a. Gets the absolute each number in the array

b. adds all the value in the array

### Exercise 10.4

1. a. double[] ddd = new double[15];

b. String[] yes= new String[20];

1. The list of numbers between the braces {1,2,3,4}
2. A. int[] testscore = { 100,90,75,60,88};

b. double[] rate = {.12,.05,.15}

c. String[] name ={Josh, Martin};

### Exercise 10.5

1. The computer sends back garbage.

### Exercise 10.6

1. Parallel arrays are several arrays with the same number of elements that work in tandem to organize data.
2. String[] name = new string[50];  
   int[]     age = new int[50];

String social = new String[50]; // string type because might have ‘-‘ in it

1. For (int x=0; x < numbers.length; x++)

System.out.printf(“Name: %-%20s Numbers: %20d%n” name[x], number[x]);

### Exercise 10.7

1. A. For (int element : abc)

System.out.println(abc[element]);

b. for (String element : abc)

if (target.equals( element)

found = true;

1. Can not be used to move through an array in reverse. Can not assign values to a positions in an array.

### Exercise 10.8

1. The assignment operator makes the two variables refer to the same array object.
2. As with other objects, arrays are not copied when the assignment operator is used. Therefore, one must instantiate a new array of the same type and length as the original, transfer the items to the new arrays, and return the new array.
3. double average(double[]a){  
   double sum = 0;  
   for (int i : a)  
   sum += i;  
   return sum / a.length;  
   }
4. int[] subArray(int[] a, int first, int last){  
   int[] result = new int[last - first + 1];  
   for (int i = first; i <= last; i++)  
   result[i - first] = a[i];  
   return result;  
   }

### Exercise 10.9

### Student getHighStudent(Student[] students){ Student highest = null; for (int i = 0; i < studentCount; i++) if (highest == null ||  students[i].getHighScore() > highest.getHighScore()) highest = students[i]; return highest; }

1. StudentArray is declared, 5 stundents are declared then printed out.

### Review pg. 377

1. A. vaild

b. vaild

c. unvaild

d. unvaild

E. unvaild

f. vaild

G. vaild

H. vaild

I. unvaild

J. vaild

1. A. 6

b. 3

c. 3

D. error, greater than the size of the array

E. 10

F. error, can’t square a array

1. A. Array type is int can’t make 10 new doubles

b. can’t make 1.5 objects in an array

c. array can’t have a negative

1. public static int selectRandom(int[ ] array) {  
   Random rand = new Random();  
   int randNum = rand.nextInt((array.length-1))+0;  
   return array[randNum];  
   }

##### Critical Thinking

**public** **static** **int** searchArray (**int**[] a, **int** item)

{

**if**(a[a.length/2]==item)

**return** a[a.length/2];

**else** **if**(a[a.length/2]<item)

**for**(**int** j=a.length/2; j> 0; j--)

**if**(a[j]==item)

**return** a[j];

**else**

**for**(**int** t=a.length/2; t<a.length; t++)

**if**(a[t]==item)

**return** a[t];

}//end of method

If the element is not in the middle, than the method loops down through the array if the middle term is less item. If the middle is bigger than item it loops up through the array. This way is more effective because the loop only has to go through half of the array instead of the whole thing.

// project 10-1

**import** java.util.Scanner;

**import** java.util.Arrays;

**public** **class** project101 {

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

{

**int**[] evenList = **new** **int**[10];

**int** [] oddList = **new** **int**[10];

**int**[] negativeList = **new** **int**[10];

**int** num, even = 0,odd = 0,neg = 0;

String evenstr = **null** , oddstr = **null**, negivtive = **null**;

Scanner reader = **new** Scanner(System.in);

**for** (**int** i =0; i < 10; i++)

{

System.out.println(" enter in a number(#" + (i+1) + ") :: " );

num = reader.nextInt();

**if** (num % 2 ==0)

{

evenList[even] = num;

even++;

}

**else** **if** (num % 2 ==1)

{

oddList[odd] = num;

odd++;

}

**else** **if** ( num <0)

{

negativeList[neg] = num;

neg++;

}

}// end of for loop

**for** (**int** y= 0; y < evenList.length;y++)

{

**if** (evenList[y] > 0)

evenstr = evenList[y] +" ";

**if** (oddList[y] > 0)

oddstr = oddList[y] +" ";

**if** (negativeList[y] > 0)

negivtive = negativeList[y] +" ";

}// end of loop number 2

System.out.print(" Even numbers are:: " + evenstr + " \n Odd numbers are:: " + oddstr + "\n The negitves are:: " + negivtive);

}// end of main

}// end of project 10-1 class

// project 10-2

**import** java.util.Scanner;

**public** **class** Project102 {

**public** **static** **double** arraydoubles(**double**[] a)

{ **double** sum = 0, aver = 0;

**int** count = 0;

**for** (**int** i =0; i < a.length;i++ )

{

sum += a[i];

count++;

}

aver = sum /count;

**return** aver;

}// end of double

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

{

**double**[] numbers = **new** **double**[10];

**double** num = 0;

**double** aver = 0;

Scanner reader = **new** Scanner(System.in);

**for** (**int** i =0; i < 10; i++)

{

System.out.println(" enter in a number(#" + (i+1) + ") :: " );

numbers[i] = reader.nextDouble();

}// end of for loop

aver = *arraydoubles*(numbers);

System.out.print(" the avarage is:: " + aver + "\n ");

**for** ( **int** y = 0; y < numbers.length; y++)

{

**if** (numbers[y] > aver)

System.out.print(" "+ numbers[y]);

}

}// end of main

}// end of class project 10-2

//project 10-3

**import** java.util.Scanner;

**public** **class** project103 {

**public** **static** **int** mode (**int**[] a){

**int** [] nums = **new** **int**[a.length];

**for** (**int** i = 0; i < a.length; i++)

{

**for**(**int** j = 0; j < a.length; j++)

**if** (a[j] == a[i])

nums[i]++;

}

**int** modeIndex = 0;

**for**(**int** i = 0; i < nums.length; i++)

**if**(nums[i] > nums[modeIndex])

modeIndex = i;

**return** a[modeIndex];

}

**public** **static** **void** main (String arg[])

{

**int**[] arrayMode = **new** **int**[10];

**int** modeset = 0;

Scanner reader = **new** Scanner(System.***in***);

**for** (**int** i =0; i < 10; i++)

{

System.***out***.println(" enter in a number(#" + (i+1) + ") :: " );

arrayMode[i] = reader.nextInt();

}// end of for loop

modeset = *mode*(arrayMode);

System.***out***.println(" The mode of this set of numbers is::" + modeset);

}

}// end of class

// Case Study 10.1: TestScoresView class

**import** java.util.Scanner;

**public** **class** TestScoresView{

**private** TestScoresModel model;

**public** TestScoresView(TestScoresModel m){

model = m;

run();

}

// Menu-driven command loop

**private** **void** run(){

**while** (**true**){

System.out.println("Number of students: " + model.size());

System.out.println("Index of current student: " +

model.currentPosition());

displayMenu();

**int** command = getCommand("Enter a number [1-11]: ", 1, 11);

**if** (command == 11)

**break**;

runCommand(command);

}

}

**private** **void** displayMenu(){

System.out.println("MAIN MENU \n"

+ "1.Display the current student \n"

+ "2.Display the class average \n"

+ "3. Display the Students with the highest grade \n"

+ "4.Display all students \n"

+ "5. Edit the current student \n"

+ "6. Add a new student \n"

+ "7.Move to the first student \n"

+ "8.Move to the last student \n"

+ "9.Move to the next student \n"

+ "10.Move to the previous student \n"

+ "11. Quit the program " );

// Exercise: List the menu options

}

// Prompts the user for a command number and runs until

// the user enters a valid command number

// Parameters: prompt is the string to display

// low is the smallest command number

// high is the largest command number

// Returns: a valid command number

**private** **int** getCommand(String prompt, **int** low, **int** high){

// Exercise: recover from all input errors

Scanner reader = **new** Scanner(System.in);

System.out.print(prompt);

**int** input = reader.nextInt()

**while**( input < low || input > high)

{

System.out.print(prompt);

**int** input = reader.nextInt();

}

**return** input;

}

// Selects a command to run based on a command number,

// runs the command, and asks the user to continue by

// pressing the Enter key

**private** **void** runCommand(**int** command){

// Exercise

Scanner read = **new** Scanner(System.in);

**if** (command == 1)

model.currentStudent();

**else** **if** (command == 2)

model.getClassAverage();

**else** **if** (command == 3)

model.getHighScore();

**else** **if** (command == 4)

model.toString();

**else** **if** (command == 5)

edit();

**else** **if** (command == 6)

add();

**else** **if** (command == 7)

model.first();

**else** **if** (command == 8)

model.last();

**else** **if** (command == 9)

model.next();

**else** **if** (command == 10)

model.previous();

}

**private** **void** add()

{

**int**[] scores[i] =reader.nextInt();

Scanner reader **new** Scanner(system.in);

System.out.println("Enter The student's name");

String name = reader.next();

**for** (**int** i=0; i<scores.length; i++)

{

System.out.println("Enter the Student's scores");

scores[i] = reader.nextInt();

}

Student s = **new** Student(name, scores);

model.add(s);

}

**private** **void** edit(){

Scanner reader = **new** Scanner(System.in);

Student s = model.currentStudent();

System.out.print("Menu \n"

+ "1. Change Test Scores\n"

+ "2. Change name")

**int** opt=reader.nextint();

**if** (opt== 1){

System.out.println("Which test would you like to change?");

**int** test = reader.nextInt();

System.out.println("Enter the score");

**int** score = reader.nextInt();

s.setScores(test-1, score);

}

**else** **if** (opt == 2 )

{

System.out.print("enter the student's name");

String name = reader.next();

s.setName(name);

}

}

}// end of testScores