King Saud University
College of Computer & Information Science
CSC111 – Lab09
Arrays – I –
All Sections

Instructions

Web-CAT submission URL:
http://10.131.240.28:8080/Web-CAT/WebObjects/Web-CAT.woa/wa/assignments/eclipse

Objectives:

- To know how to define and create an array.
- To know how to access array elements.
- To know how to iterate over arrays using loops
- To know how to manipulate arrays

Lab Exercise 1

Part 1

Write a program that reads and prints 3 integers.

Sample Run

```
Enter the numbers: 4 10 8 ↵
The numbers are: 4 10 8
```

Part 2

In the previous program you have defined 3 integer variables to store the numbers. What if we modify the previous problem to writing a program that reads and prints 100 integers? Are you going to define 100 variables? How about 1000 integers? Clearly, using variables for such problem is not practical. In this case, it is better to use arrays. ***Arrays provide an easy and efficient way to store and process a large amount of data in memory***.

Rewrite your previous program using arrays by completing the following pseudo code:

Complete following pseudo code

```
import java.util.Scanner;
public class Simple {
	public static void main(String[] args) {
		Scanner s = new Scanner(System.in);

		/* define and create integers array a of size 3*/

		//reading first element
		a[/*enter index of 1st element*/] = s.nextInt();

		/*read 2nd element */
		/*read 3rd element */

		//printing array
		System.out.print("The numbers are: ");
		System.out.print(/* first element of array */);

		/* print 2nd element in the array */

		/* print 3rd element in the array */
	}
}
```

Part 3

Arrays give you flexible memory allocation based on user input. Write a program **ReadInt** that:

1. Reads an integer N.
2. Then your program:
    a. Creates an array **a** of size N.
    b. Reads N integers and stores them in **a**
    c. Prints the content of **a**.

Use class name **ReadInt.**

Sample Run

```
Enter number of ints you want to read: 5 ↵
Enter 5 the numbers: 1 5 10 88 15 ↵
The number you entered are: 1 5 10 88 15
```

Complete following pseudo code

```
import java.util.Scanner;
public class ReadInt {
    public static void main(String[] args) {
      Scanner input = new Scanner(System.in);

      // Read array size and make sure it is greater than zero
      System.out.print("Enter number of ints you want to read (> 0): ");
      /* read array size */

      while (/*array size less than one */){
            System.out.print("Number must be > 0. "
                        + "Enter number of ints you want to read: ");
            /* read array size again */
      }

      /* create the array */

      System.out.print("Enter " + numOfInts +" numbers: ");

      /* read array elements */

      System.out.print("The number you entered are: ");
```

```
      /* print array elements */

    }
}
```

As you can see, arrays do not only allow us to easily allocate large amount of memory, but they also give us the ability to dynamically allocate memory of any size depending on user input.

Part 4

Change previous program by adding code at the end that does the following:

1. Changes the content of the 1st element to 6.
2. Changes content of 5th element to number 5. Notice that you have to make sure that your array contains at least 5 elements.
3. Changes content of element at index 5 to number 6. Notice that you have to make sure that your array contains at least 6 elements (why 6? Why not 5?).
4. Changes content of last element to 100. What is the index of the last element? Is it N or N-1 and why?
5. Changes content of the element before the last element to 50.
6. Define an int variable i, read the value of i and then change the value of the element with index (i * 2) to 4.

Now print the array again. Use same program from part 3 **(ReadInt).**

Sample Run I

```
Enter number of ints you want to read (> 0): 5 ↵
Enter 5 numbers: 99 87 63 2 35 ↵
The number you entered are: 99 87 63 2 35
Enter a number i ( > 0): 2 ↵
6 87 63 50 4
```

Sample Run II

```
Enter number of ints you want to read (> 0): -1 ↵
Number must be > 0. Enter number of ints you want to read: 0 ↵
Number must be > 0. Enter number of ints you want to read: 2 ↵
Enter 2 numbers: 3 9 ↵
The number you entered are: 3 9
Enter a number i ( > 0): 1 ↵
```

```
50 100
```

Complete following pseudo code

```
import java.util.Scanner;
public class ReadInt {
      public static void main(String[] args) {
      Scanner input = new Scanner(System.in);

      // Read array size and make sure it is greater than zero
      System.out.print("Enter number of ints you want to read (> 0): ");
      /* read array size */

      while (/*array size less than one */){
            System.out.print("Number must be > 0. "
                        + "Enter number of ints you want to read: ");
            /* read array size again */
      }

      /* create the array */

      System.out.print("Enter " + numOfInts +" numbers: ");

      /* read array elements */

      System.out.print("The number you entered are: ");

      /* print array elements */

      System.out.println();

       if (/*array size greater than zero*/)
            a[0] = 6;

       /* if array size greater than 4
            assign 5th element the value 5 */

       /* if array size greater than 5
             assign 6th element the value 6 */

       /* assign last element the value 100 */

       /* assign element before the last one value */

       System.out.print("Enter a number i ( > 0): ");
```

```
      /* read variable i */

       /* assign 4 to element i * 2 if that is allowed */

      /* print array elements */
    }
}
```

At this point, submit your program to WebCAT.

Home exercise: the previous code still has a small problem, can you find it?

Part 5

Create a new program **ReadIntReverse** (that is based on previous one). Your new program runs as following:

1. The program starts by reading an integer N such that N > 0.
2. Then it creates an array **aReverse** of size N.
3. Then it reads N integers and stores them in **aReverse**.
4. Finally, it prints the integers *in reverse*.

Use class name **ReadIntReverse**.

Sample Run

```
Enter number of ints you want to read (> 0): 5 ↵
Enter 5 the numbers: 1 3 5 7 8 ↵
The number you entered are: 8 7 5 3 1
```

At this point, submit your program to WebCAT.

Part 6

Common mistakes. Now let us do some changes to previous program in ***part 3*** to see some of the common mistakes that beginners commit when using arrays.

1. Change line 14 and remove the = sign and everything after it.

```
int a[];
```

What happens? You must always create and initialize the array.

2. Change line 17 by replacing 0 with 1.

```
for (int i = 1; i < a.length; i++){
```

Did you print the whole array? Remember array indices always start from 0 not 1. (same goes to loop on line 10).

3. Change line 17 in your program by replacing the < with <=.

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

Run your program and see what happens. You ran out of the array and java stopped you from doing this. Remember, last index in an array is always equal to `length – 1`.

Lab Exercise 2

Part 1

Complete the pseudo code below for a program that reads 5 numbers entered by the user and store them in an array. A number is stored only if:

- The number is unique (has not been entered before).
- The number is between 10 and 100.

Your program should go as following:

1. Initialize an integer array **numbers** to hold five elements. This is the maximum number of values the program must store if all values input are unique (program may store less than that).
2. Remember to validate the input and display an error message if the user inputs invalid data (not between 10 and 100).
3. If the number entered is not unique, display a message to the user `"x has already been entered"` where x is the input number; otherwise, store the number in the array and display the list of unique numbers entered so far.

After you finish the pseudo code below, write a class **`UniqueTest`** with a **`main`** method in a separate file. The class creates an object of type `Unique` and calls method `getNumbers()`.

Sample Run

```
Enter number: 10 ↵
10

Enter number: 10 ↵
10 has already been entered.
10

Enter number: 120 ↵
number must be between 10 and 100
10

Enter number: 12 ↵
10
12

Enter number: 23 ↵
10
12
23
```

Complete following pseudo code

```
import java.util.Scanner;
public class Unique {
  // gets 5 unique numbers from the user
  public void getNumbers(){
    Scanner input = new Scanner( System.in );
    /* Create an array of five elements*/
    int count = 0; // number of uniques read
    int entered = 0; // number of entered numbers
    while( entered < numbers.length ){
            System.out.print( "Enter number: " );

            /* Write code here to retrieve the input from
            the user */

            //increment number of entered numbers
            entered++;

            //validate the input
            if(/*validate the input is between 10 and 100 */)
            {
                    //flags whether this number already exists
                    boolean containsNumber = false;
```

```
                    /*Compare the user input to the unique numbers in
                    the array using a for statement. If the number is
                    unique, store new number */

                    //add the user input to the array only if the
                    //number is not already in the array
                    if ( !containsNumber ){

                            /* Write code to add the number to the
                            array and increment
                            the number of unique items */

                    } // end if
                    else
                    System.out.println( number + " has already"+
                                                        " been entered\n");
            }//endif
            else
                    System.out.println("number must be between "+
                                                "10 and 100" );

            // print the list of unique values

            /* Write code to output the contents of the array */
            System.out.println();

    } // end while
  } // end method getNumbers
} // end class Unique
```

At this point, submit your program to WebCAT.

Part 2

Change your previous program as following:

1. Array size is input by the user.
2. User keeps entering unique numbers until array is filled up.

Name your classes `Unique2` and `UniqueTest2`.

Sample Run

```
Enter how many numbers you want to input: 4 ↵
Enter number: 10 ↵
10

Enter number: 10 ↵
10 has already been entered.
10

Enter number: 12 ↵
```

```
10
12

Enter number: 190 ↵
number must be between 10 and 100
10
12

Enter number: 55 ↵
10
12
55

Enter number: 12 ↵
12 has already been entered.
10
12
55

Enter number: 45 ↵
10
12
55
45
```

At this point, submit your program to WebCAT.

Done…