

ENGR 131
BRANICKY
MT#2 REVIEW

- 7.8 Write Java statements to accomplish each of the following tasks:
- a) Display the value of the seventh element of character array f.
 - b) Initialize each of the five elements of one-dimensional integer array g to 8.
 - c) Total the 100 elements of floating-point array c.
 - d) Copy 11-element array a into the first portion of array b, which contains 34 elements.
 - e) Determine and display the smallest and largest values contained in 99-element floating-point array w.
- 7.9 Consider a two-by-three integer array t.
- a) Write a statement that declares and creates t.
 - b) How many rows does t have?
 - c) How many columns does t have?
 - d) How many elements does t have?
 - e) Write the names of all the elements in the second row of t.
 - f) Write the names of all the elements in the third column of t.
 - g) Write a single statement that sets the element of t in row 1 and column 2 to zero.
 - h) Write a series of statements that initializes each element of t to zero. Do not use a repetition statement.
 - i) Write a nested for statement that initializes each element of t to zero.
 - j) Write a nested for statement that inputs the values for the elements of t from the user.
 - k) Write a series of statements that determines and displays the smallest value in t.
 - l) Write a statement that displays the elements of the first row of t.
 - m) Write a statement that totals the elements of the third column of t.
 - n) Write a series of statements that displays the contents of t in tabular format. List the column indices as headings across the top, and list the row indices at the left of each row.

For Exercises 1–7, assume that the following declarations have been made:

```
final int LITTLE = 5,
        MEDIUM = 9,
        BIG = 20;
int i, j, temp;
int [] number = {99, 33, 44, 88, 22, 11, 55, 66, 77};
double [] value = new double [LITTLE],
        large = new double [BIG];
```

Tell what values (if any) will be stored in all the arrays involved or explain why an error occurs.

- 1. for (i = 0; i < LITTLE; i++)
 number[i] = i / 2;
- 2. for (i = 0; i < number.length; i += 2)
 number[i] = i * i;
- for (i = LITTLE; i < number.length; i++)
 number[i] = number[i - 5];
- 3. for (i = 0; i < 3; i++)
 value[i] = 0;
- for (i = 3; i < LITTLE; i++)
 value[i] = 1;

Exercises 7–10 assume that the following declaration has been made:

```
int [][] mat = new int [3][3];
```

Tell what value (if any) is stored in each array element, or explain why an error occurs.

- 7. for (int i = 0; i < 3; i++)
 for (int j = 0; j < 3; j++)
 mat[i][j] = i + j;
- 8. for (int i = 0; i < 3; i++)
 for (int j = 2; j >= 0; j--)
 if (i == j)
 mat[i][j] = 0;
 else
 mat[i][j] = 1;

12.2.2 String Methods

The String class has many methods for working with String objects. These methods are useful for processing text.

- `charAt(int position)` returns the character at the given position in the string. The first character is at position 0, just as the first element of an array is at index 0.

```
> String str1 = "Bye";
> System.out.println(str1.charAt(0));
B
```

- `compareTo(Object o)` returns a negative number if this object is less than the passed object, 0 if this object is equal to the passed object and a positive number if this object is greater than the passed object.

```
> String str1 = "Bye";
> String str2 = "Hi";
> String str3 = new String("Bye");
> System.out.println(str1.compareTo(str2));
-6
> System.out.println(str2.compareTo(str1));
6
> System.out.println(str1.compareTo(str3));
0
```

- `substring(int n, int m)` returns a new string which is a *substring* of the string starting at the *n*th character and preceding up to *but not including* the *m*th character. A substring includes part of the original string.

```
> String str2 = "Do you go to school?";
> String str3 = str2.substring(3,6);
> System.out.println(str3);
you
```

- `substring(int n)` returns a new string which is a substring of the string starting at the *n*th character and including the rest of the characters in the string.

```
> str3 = str2.substring(6);
> System.out.println(str3);
go to school?
```

- `startsWith(String prefix)` returns true if the string starts with the given prefix, else it will return false.

```
> String letter = "Mr. Guzdial requests the ";
> letter = letter + "pleasure of your company ...";
> System.out.println(letter.startsWith("Mr.));
true
> System.out.println(letter.startsWith("Mrs.));
false
```

- `endsWith(String suffix)` returns true if the string ends with the given suffix, else it will return false.

```
> String filename="barbara.jpg";
> if (filename.endsWith(".jpg"))
    System.out.println("it is a picture");
it is a picture
```

- `indexOf(String str)` returns the first index of the passed str, if it is found. If str isn't in the current string, it will return -1.

```
> System.out.println(letter);
Mr. Guzdial requests the pleasure of your company ...
> System.out.println(letter.indexOf("Guzdial"));
4
> System.out.println(letter.indexOf("Mark"));
-1
```

- `indexOf(String str, int fromIndex)` returns the first index of the passed `str` at or after the passed `fromIndex`, if it is found. If `str` isn't in the current string at or after the `fromIndex`, it will return `-1`.

```
> String t = "That which is, is. That which is not, is not.";
> System.out.println(t.indexOf("is",14));
15
```

- `lastIndexOf(String str)` returns the last index of the passed `str`, if it is found. If `str` isn't in the current string, it will return `-1`.

```
> String s = "It is a nice day, isn't it?";
> System.out.println(s.lastIndexOf("it"));
24
```

- `lastIndexOf(String str, int index)` returns the last index of the passed `str` found looking backward starting at `index`. If `str` isn't in the current string before the given index, it will return `-1`.

```
> String s = "It is a nice day, isn't it?";
> System.out.println(s.lastIndexOf("is",17));
3
```

- `toUpperCase()` returns a new string with all the characters in uppercase.

```
> System.out.println("Hello".toUpperCase());
HELLO
```

- `toLowerCase()` returns a new string with all the characters in lowercase.

```
> System.out.println("Hello".toLowerCase());
hello
```

- `trim()` this will return a new string with all white space (spaces and tabs) removed from the beginning and end of the string.

```
> String strWithSpaces = "   Janet Hund   ";
> System.out.println(strWithSpaces.trim());
Janet Hund
```

These methods can be *cascaded*—one modifying the result of another.

```
> String test ="This is a test of Something."
> System.out.println(test.substring(5).toUpperCase());
IS A TEST OF SOMETHING.
```

1. Assume that `s1` and `s2` are Strings, and that

```
String s = "abcdefghijABCDEFGHIJabcdefghij";
```

What will be the contents of `s1` and `s2` after the following?

```
b. s1 = s.substring(1,3);
   s2 = s.substring(7,9);
   s1 = s1.concat(s2);
```

3. Assume the definition

```
String s1 = " The first string ";
```

What will be the results of the following expressions?

```
a. s1.indexOf("st");
b. s1.indexOf("st",14);
c. s1.lastIndexOf("st");
d. s1.indexOf('i');
e. s1.replace('s','S');
f. s1.toUpperCase();
g. s1.trim();
```

2. Assume the definitions

```
String s1 = "Test1";
String s2 = "test1";
String s3 = "Test1";
String s4 = "Test2";
String s5 = s1;
```

What will be the results of the following expressions?

```
a. s1.equals(s2);
b. s1.equals(s3);
c. s1.equalsIgnoreCase(s2);
d. s1 == s3;
e. s1 == s5;
f. s1.compareTo(s2);
g. s1.compareTo(s4);
h. s1.regionMatches(1, s2, 1, 3);
i. s1.regionMatches(true, 1, s2, 1, 3);
j. s1.startsWith("Te");
k. s4.endsWith("1");
```

PROBLEMS

11.1 What is the answer to each of the following?

- What is a field?
- What is a constructor?
- How do you overload a constructor?
- Can objects of a different class (in a different file) directly access private fields?
- Can objects of a different class (in a different file) directly access public fields?
- What is an accessor method?
- What is a modifier method?

11.2 Which of these is the correct class definition for a Teacher class?

- `public CLASS Teacher`
- `public class Teacher`
- `public class TEACHER`
- `public class Teacher []`
- `public class Teacher extends Object, Person`

11.3 Which of these correctly defines a field in a class?

- `private INT count;`
- `public INT count;`
- `private int count = "HI";`
- `private int count;`

11.4 Which of these correctly defines a constructor in the class Student?

- `PUBLIC STUDENT ()`
- `public STUDENT ()`
- `PUBLIC student()`
- `public void Student()`
- `public Student()`

3.5 Which of the following are class methods and which are object methods? How can you tell which are which?

- `Math.abs(-3);`
- `soundObj.play();`
- `FileChooser.pickAFile();`
- `pictureObj.show();`
- `ColorChooser.pickAColor();`
- `turtle1.turnLeft();`

11.16 Write the definition for a class `Car` which has the manufacturer, model, year, and number of doors. An example car has a manufacturer of Toyota, a model of Camry, a year of 2000, and a number of doors of 4. Be sure to code all accessor and modifier methods.

3.15 Create a class called `Date` that includes three pieces of information as instance variables—a month (type `int`), a day (type `int`) and a year (type `int`). Your class should have a constructor that initializes the three instance variables and assumes that the values provided are correct. Provide a *set* and a *get* method for each instance variable. Provide a method `displayDate` that displays the month, day and year separated by forward slashes (/). Write a test application named `DateTest` that demonstrates class `Date`'s capabilities.

6. Assume that `values` is a 101-element array containing a list of measurements from a scientific experiment, which has been declared by the statement

```
double values[] = new double[101];
```

Write the Java statements that would count the number of positive values, negative values, and zero values in the array, and write out a message summarizing how many values of each type were found.

7. Write Java statements that would print out every fifth value in the array `values` described in Exercise 6. The output should take the form

```
values[0] = x.xx
values[5] = x.xx
...
values[100] = x.xx
```

- 6.7*** (*Counting single digits*) Write a program that generates one hundred random integers between 0 and 9 and displays the count for each number. Hint: Use `(int)(Math.random() * 10)` to generate a random integer between 0 and 9. Use an array of ten integers, say `counts`, to store the counts for the number of 0's, 1's, ..., 9's.
- 6.9** (*Finding the smallest element*) Write a method that finds the smallest element in an array of integers. Use {1, 2, 4, 5, 10, 100, 2, -22} to test the method.
- 6.10** (*Finding the index of the smallest element*) Write a method that returns the index of the smallest element in an array of integers. If there are more than one such elements, return the smallest index. Use {1, 2, 4, 5, 10, 100, 2, -22} to test the method.
- 6.11*** (*Computing deviation*) Exercise 5.21 computes the standard deviation of numbers. This exercise uses a different but equivalent formula to compute the standard deviation of `n` numbers.

$$\text{mean} = \frac{\sum_{i=1}^n x_i}{n} = \frac{x_1 + x_2 + \dots + x_n}{n} \quad \text{deviation} = \sqrt{\frac{\sum_{i=1}^n (x_i - \text{mean})^2}{n - 1}}$$

To compute deviation with this formula, you have to store the individual numbers using an array, so that they can be used after the mean is obtained. Use {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} to test the method.

- 6.12*** (*Reversing an array*) The `reverse` method in §6.5 reverses an array by copying it to a new array. Rewrite the method without creating new arrays.
- 6.13*** (*Increasing array size*) Once an array is created, its size is fixed. Occasionally, you need to add more values to an array, but it is full. In such cases, you can create a new, larger array to replace the existing array. Write a method with the following header:

```
public static int[] doubleCapacity(int[] list)
```

The method returns a new array that doubles the size of the parameter `list`.

Section 6.10 Two-Dimensional Arrays

- 6.20*** (*Summing all the numbers in a matrix*) Write a method that sums all the integers in a matrix of integers. Use {{1, 2, 4, 5}, {6, 7, 8, 9}, {10, 11, 12, 13}, {14, 15, 16, 17}} to test the method.
- 6.21*** (*Summing the major diagonal in a matrix*) Write a method that sums all the integers in the major diagonal in a matrix of integers. Use {{1, 2, 4, 5}, {6, 7, 8, 9}, {10, 11, 12, 13}, {14, 15, 16, 17}} to test the method.

12.15 Imagine that you have a list of all the genders (as single characters) of the students in your class, in order of their last name. The list will look something like "MFFMMMFFMFMMFFFM" where "M" is Male and "F" is Female. Write a function (below) `percentageGenders(string)` to accept a string that represents the genders. You are to count all of the "M"s and "F"s in the string, and print out the ratio (as a decimal) of the each gender. For example, if the input string were "MFFF," then the function should print something like "There are 0.25 Males, 0.75 Females." (Hint: Better multiply something by 1.0 to make sure that you get floats not integers.)

8.4* (*Occurrences of a specified character*) Write a method that finds the number of occurrences of a specified character in the string using the following header:

```
public static int count(String str, char a)
```

For example, `count("Welcome", 'e')` returns 2.

8.6* (*Counting the letters in a string*) Write a method that counts the number of letters in the string using the following header:

```
public static int countLetters(String s)
```

Write a `main` method to invoke `countLetters("Java in 2008")` and display its return value.

8.8* (*Binary to decimal*) Write a method that parses a binary number as a string into a decimal integer. The method header is as follows:

```
public static int parseBinary(String binaryString)
```

For example, `binaryString 10001` is 17 ($1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2 + 1 = 17$) So, `parseBinary("10001")` returns 17. Use binary string `11111111` to test the method. Note that `Integer.parseInt("10001", 2)` parses a binary string to a decimal value. Do not use this method in this exercise.

8.5** (*Occurrences of each digit in a string*) Write a method that counts the occurrences of each digit in a string using the following header:

```
public static int[] count(String s)
```

The method counts how many times a digit appears in the string. The return value is an array of ten elements, each of which holds the count for a digit. For example, after executing `int[] counts = count("12203AB3")`, `counts[0]` is 1, `counts[1]` is 1, `counts[2]` is 2, `counts[3]` is 2.

29.5 Write an application that uses random-number generation to create sentences. Use four arrays of strings called `article`, `noun`, `verb` and `preposition`. Create a sentence by selecting a word at random from each array in the following order: `article`, `noun`, `verb`, `preposition`, `article` and `noun`. As each word is picked, concatenate it to the previous words in the sentence. The words should be separated by spaces. When the final sentence is output, it should start with a capital letter and end with a period. The application should generate 20 sentences and output them to a text area.

The `article` array should contain the articles "the", "a", "one", "some" and "any"; the `noun` array should contain the nouns "boy", "girl", "dog", "town" and "car"; the `verb` array should contain the verbs "drove", "jumped", "ran", "walked" and "skipped"; the `preposition` array should contain the prepositions "to", "from", "over", "under" and "on".

After the preceding application is written, modify it to produce a short story consisting of several of these sentences. (How about the possibility of a random term-paper writer?)

SOURCES: Java: How to Program, 6/e by H.M. Deitel and P.J. Deitel
Java: An Intro. to Computing by J. Adams, L.R. Nyhoff, and J. Nyhoff
Intro. to Computing and Programming in Java by M. Guzdial and B. Ericson
Java for Engineers and Scientists, 2/e by S.J. Chapman
Intro. to Java Programming: Fundamentals First, 6/e by Y.D. Liang