**Name:**

**Advanced Programming in Java**

**Lab Exercise 10/3/2019**

**Advanced Array Concepts**

Read Lesson 19 in Blue Pelican Java

1. Write code that will create an array of 300 *BankAccount* objects. You are only to

instantiate two of them. The object with index 47 should have a beginning balance of

$92, and index 102 should have $1007. The name of your array will be *ba*.

2. Write an *if* statement that will decide if *k[3]* is equal to *jm[5]* where it is assumed that *k* and *jm* are numeric arrays.

3. Write an *if* statement that will decide if *s[2]* is equal to *ss[19]* where it is assumed that *s* and *ss* are *String* arrays.

4. Write an *if* statement that will decide if *cir[2]* is equal to *cirr[10]* (with regard to content) where it is assumed that *cir* and *cirr* are object arrays of type *Circle*.

5. What’s wrong with the following code?

char months[];

months[0] = ‘j’;

6. String suv[] = new String[20];

j = 0;

while(j < 17 )

{

suv[j] = “Hello”;

j++;

}

What is the logical size of the *suv* array?

What is the physical size of the *suv* array?

7. Write code using *toCharArray* to convert *String d = “The quick brown fox jumped over the lazy dog.”* into the character array *qbf*.

8. double rub[] = {23.0, -102.1, 88.23, 111, 12.02, 189.119, 299.88};

double dub[] = {1, 2, 3, 4, 5, 6, 7, 8, 9};

Write a single line of code (using arraycopy) that will result in *dub* looking like this:

{1, 2, 3, 4, 111, 12.02, 189.119, 8, 9}

9. double[] zz, top;

top = {12.1, 13.1, 14.1, 15.1, 18};

zz = top;

zz[2] = 99;

top[3] = 100.2;

Show what both arrays would look like at the completion of the above code.

10. char[] a, b;

a = “Groovy dude”.toCharArray( );

b = “I like this”.toCharArray( );

System.arraycopy(a, 1, b, 0, 4);

What do the two arrays look like at the completion of this code?

11. What must be true of any array before we can use *Arrays.binarySearch( )?*

12. Write code that will establish an array called *myArray* having the following elements,

{189.01, 2000, -32, 56, 182, 2}. Then sort the array.

13. Assume the array *myArray* in #12 has been correctly sorted. What would be printed with the following?

System.out.println( Arrays.binarySearch(myArray, 56) );

System.out.println( Arrays.binarySearch(myArray, 102) );

14. What does the following print?

int xc[] = {123, 97, -102, 17};

int pk[] = {123, 79, -102, 17};

int gs[] = {123, 97, -102, 17};

System.out.println( Arrays.equals(xc, pk) + “\n” + Arrays.equals(xc, gs));

15. What does the following print?

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

Arrays.fill(pickle, -1);

System.out.println( pickle[4] );

16. If a command line reads, *java BigClass Munster Herman dude*, what will the following line inside the *main* method print?

System.out.println(“Name=” + args[2] +args[1] );

17. What’s printed by the following?

int px[] = {3, 4, 5, 6, 7, 8, 9};

System.out.println( px[ px[1] + 1 ]);

18. Write code using the “for-each” style of a *for* loop that will accumulate and print the

product of the state variables *int jj* within each object of object array *objArray*. Assume

the objects are created from the class *DummyClass.*

**Project… Sorting a *String* Array**

Create a *String* array call *ss*. It will contain the following *String*s in the order shown.

{“Bill”, “Mary”, “Lee”, “Agnes”, “Alfred”, “Thomas”, “Alvin”, “Bernard”, “Ezra”,

“Herman”}

Using the technique described on page 19-3 of the Blue Pelican text, sort this array and then print the contents of the sorted array (using a loop) from index 0 to the last. Call both your project and class,

*SortStringArray*. Confine all of your code to the *main* method.

The printout should look like the following:

Agnes

Alfred

Alvin

Bernard

Bill

Ezra

Herman

Lee

Mary

Thomas

**Project… Two Orders for the Price of One**

Modify the project above so as to print two side-by-side columns. Call both your project and class *AscendDescend*. The first column should be in ascending order and the second in descending order. The output should appear as below (Be sure to include the headers):

Ascend Descend

Agnes Thomas

Alfred Mary

Alvin Lee

Bernard Herman

Bill Ezra

Ezra Bill

Herman Bernard

Lee Alvin

Mary Alfred

Thomas Agnes