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*.

BankAccount[] b = new BankAccount[300];

b[47] = new BankAccount(92.00);

b[102] = new BankAccoun(1007.00);

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.

if (k[3] == jm[5]) {

…code…

}

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.

if (s[2].equals(ss[19])) {

…code…

}

4. Write an *if* statement that will decide if *cir[2]* is equal to *cir[10]* (with regard to content)

where it is assumed that *cir* and *cirr* are object arrays of type *Circle*.

if (cir[2].equals(cir[10])) {

…code…

}

5. What’s wrong with the following code?

char months[];

months[0] = ‘j’;

The array months hasn’t been instantiated with a size.

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

j = 0;

while(j < 17 )

{

suv[j] = “Hello”;

j++;

}

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

20

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

17

7. Write code using *toCharArray* to convert *String d = “The quick brown fox jumped over*

*the lazy dog.”* into the character array *qbf*.

char[] gbf = d.toCharArray;

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}

System.arraycopy(rub, 3, dub, 4, 3);

9. double[] zz, 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.

zz: {12.1, 13.1, 99, 15.1, 18}

top: {12.1, 13.1, 14.1, 100.2, 18}

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?

a: {‘G’, ‘r’, ‘o’, ‘o’, ‘v’, ‘y’, ‘ ‘, ‘D’, ‘u’, ‘d’, ‘e’}

b: {‘G’, ‘r‘, ‘o’, ‘o’, ‘k’, ‘e’, ‘ ‘, ‘t’, ‘h’, ‘i’, ‘s’}

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

The array must be sorted in ascending order.

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.

double[] myArray = {189.01, 2000. -32, 56, 182, 2};

Array.sort(myArray);

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, 182) );

2

3

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));

false

true

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] );

-1

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] );

Name = Herman Munster

17. What’s printed by the following?

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

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

8

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.*

int product = 1;

for(int i: objArray) {

product \*= i;

}

System.out.println(i);