Worksheet: Declaring and Initializing Arrays

Write the Java code specified, paying attention to descriptive naming, proper case, and proper *syntax*, including proper quotation marks (`"`) and terminating semicolons (`;`). Code should not print anything unless explicitly told to do so in the instructions.

1. Review the basic ***declaration*** and ***initialization*** of variables, and extend this to include arrays.

   a) Write a Java statement that ***declares*** a variable intended to store the height of a student in meters.

```
double studentHeight;
```

   b) Write a Java statement that ***declares*** and ***initializes*** (to any valid value) a variable intended to store a count of the number of students present in a classroom.

```
int numStudents = 0;
```

   c) A variable named `answer` of type `String` has already been declared. Write a Java statement that ***initializes*** that variable to any string of ten characters or fewer.

```
answer = "Any chars.";
```

   d) Write a Java statement that ***declares*** an array intended to store the heights, in meters, of a number students.

```
double[] studentHeights;
```

   e) Write a Java statement that ***declares*** and allocates space for an array of 50 values, each value intended to store the status of a string of lights as either on or off.

```
boolean[] lightStatus = new boolean[50];
```

   f) Write a Java statement that ***declares*** and ***initializes*** an array with the integer digits 0 through 5, inclusive.

```
int[] digits = { 0, 1, 2, 3, 4, 5 };
```

   g) An array of `String` called `names` has already been declared. Write a Java statement that ***initializes*** this array to contain a number of strings equal to the number of Chinese characters in your name, and each string containing the pinyin of a single character from your name. For example if your name were “孙悟空”, the strings would be “Sun”, “Wu”, and “kong”.

```
names = { "Sun", "Wu", "kong" };
```

   h) Write a java statement that declares and initializes an array named `countDown` that has the capacity to store ten integers. Then write a `for` loop that initializes the values of the `countDown` array to these values:
   `9, 8, 7, 6, 5, 4, 3, 2, 1, 0`.

```
int[] countDown = new int[10];
for(int i = 0; i <10; i++) {
   countDown[i] = 9-i;
}
```

Worksheet: Declaring and Initializing Arrays

2. Review the basic ***calling*** of methods, and extend this to include arrays.

a) Given the method header "`public static void log(String s, boolean b)`", write a statement that will be run from within the same class that ***calls*** this method with valid parameters of your choice.

```
log("Hello", true);
```

b) Given the method header "`public String readLine(int lineNum)`", write a statement that will be run from within the same class that ***declares*** a variable of the appropriate type to store the return value, and ***calls*** this method with a valid parameter of your choice.

```
String line = readLine(10);
```

c) Given the method header "`public static double abs(double d)`", that exists in a class named `Math`, write a statement that will be run from a different class that ***declares*** a variable of the appropriate type to store the return value, and ***calls*** this method with a valid parameter of your choice.

```
double absoluteValue = Math.abs(-5.0);
```

d) Given an object named `cynthia` of type `Student`, and the method header "`public boolean setGpa(double gpa)`", that exists in the `Student` class, write a statement that will be executed from a class other than the `Student` class that ***declares*** a variable of the appropriate type to store the return value, and ***calls*** this method with a valid parameter of your choice.

```
boolean setSuccess = cynthia.setGpa(4.0);
```

e) Given the method header "`public int[] getTimeValues(int start, int end)`", write a statement that ***declares*** a variable of the appropriate type to store the return value, and ***calls*** this method with valid parameters of your choice.

```
int[] timeValues = getTimeValues(0, 25);
```

f) Given the method header "`public int findMaxValue(int [] a)`", write two statements: first a statement that ***declares*** and ***initializes*** a variable of the appropriate type to pass as a parameter to this method, and second a statement that ***declares*** a variable to store the return value and ***calls*** the method with the parameter you defined in the first statement. The array should include at least three elements.

```
int[] values = { 4, 9, 7 };
int maxValue = findMaxValue(values);
```

g) Given the method header "`public double[] merge(double[] a, double[] b)`", write three statements as follows. Two statements, each statement ***declares*** and ***initializes*** a separate variable of the appropriate types to pass as a parameter to this method. The third and final statement that ***declares*** a variable to store the return value, and ***calls*** the method with the parameters you defined in the first two statements. Each array declared must have at least three elements.

```
double[] a = { -0.7, 0.0, +0.7 };
double[] b = { -3.14, +3.14, +6.28 };
double[] mergedArray = merge(a, b);
```

Worksheet: Declaring and Initializing Arrays

3. Review the basic ***declaring*** of methods, and extend this to include arrays. Make all methods `public` unless specified.

   a) Write a ***method header*** for a **`static`** ***method*** named `length` that takes a parameter of type `String` and has a return value of type `int`.

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

   b) Write a ***method header*** for an ***instance method*** named `subtext` that takes two parameters, both of type `int`, and returns a value of type `String`.

```
public String subtext(int a, int b)
```

   c) Write a ***method header*** for a `static` method named `sort` that takes an array as a parameter and has no return value.

```
public static void sort(int[] a)
```

   d) Write a ***method header*** for a `static` method named `copyOf` that takes two parameters: an array named `original` of type `double`, and an integer value named `newLength`, and returns an array of type `double`.

```
public static double[] copyOf(double[] original,
                              int newLength)
```

   e) Write a ***method header*** for an instance method named `toArray` that takes no parameters, and returns an array of type `Student`.

```
public Student[] toArray()
```

4. Write a Java ***method*** named `makeSentence` that:
   - has no return value
   - takes an array of `String` as a parameter
   - prints all elements of the array on a single line, each element followed by a space character.
   - After all elements are printed, adds a newline character so the next call to `print` will output text on the next line.

```
public static void makeSentence(String[] a) {
   for(int i = 0; i<a.length; i++) {
      System.out.print(a[i]+" ");
   }
   System.out.println();
}
```