|  |  |  |  |
| --- | --- | --- | --- |
| Instructor | ***Katherine Papademas*** | Due Date | **9/17/16** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Part | **1** | **2** | **3** | **4** | Total |
| *Maximum Points* | **25** points | **25** points | **25** points | **25** points | **100**G101010 pointsG |
| ***Your Score*** |  |  |  |  |  |

**Textbook Reading Assignment**

Thoroughly read Chapter(s) 4 And 5 in your **Java Programming** textbook.

**Part 1 Glossary Terms**

Define, in detail, each of these glossary terms from the realm of computer programming logic and design and computer topics, in general. If applicable, use examples to support your definitions. Consult your notes or course textbook(s) as references or the Internet by visiting Web sites such as:

|  |  |  |
| --- | --- | --- |
| **http://www.askjeeves.com** | **http://www.webopedia.com** | **http://www.wikipedia.org** |

**(a) Global Variables**

|  |
| --- |
| There really isn’t such a thing as a global variable in java, however, you use the word static to give a global state to a variable so that it can be access by all classes. |

**(b) Intrinsic Method**

|  |
| --- |
| A faster way of getting an output working inside the method |

**(c) Method**

|  |
| --- |
| It is a block of code that has a specific name and can be called upon at any point in the program. |

**(d) static Method**

|  |
| --- |
| These methods belong to the class and have no instance variables. They will usually take input from parameters, perform whatever action is on there, and give an output. |

**(e) Stub**

|  |
| --- |
| It is a small program routine that replaces a longer program. This is used when it will be loaded later or loaded remotely. |

**Part 2 Textbook Exercises - Methods**

For each of the following, enter T if the answer is True, otherwise enter F for False.

**T** **(1)** Common programming practices include establishing the algorithm of a method after a stub is defined.

**T** **(2)** Another term utilized to describe the call of a method is invocation.

**T** **(3)** In the call of a method, always use the same number of arguments as there are found in the parameters of the declaration.

**F** **(4)** Never use, in a Java applet, a method which works in a Java program, because they are not compatible.

**F** **(5)** A static method cannot be assessed without creating an object of the class type where the method is declared.

**Part 3 Topics in Computer Programming - class Math Predefined Methods**

Predefined mathematical methods that are part of the class Math in the package java.lang include those below.

|  |  |  |
| --- | --- | --- |
| **method name** | **description** |  |
|  |  |  |
| abs( m ) | returns the absolute value of m |  |
| ceil( m ) | rounds m to the smallest integer not less than m |  |
| floor( m ) | rounds m to the largest integer not greater than m |  |
| max( m , n ) | returns the larger of m and n |  |
| min( m , n ) | returns the smaller of m and n |  |
| pow( m , n ) | returns m raised to the power n |  |
| round( m ) | returns a value which is the integer closest to m |  |
| sqrt( m ) | returns a value which is the square root of m |  |

Evaluate each of the following, which include Math predefined methods.

|  |
| --- |
| **(1) 6.0** Math.abs( 6.0 ) **(2) 6.0** Math.abs( − 6.0 )  **(3) 11.0** Math.ceil( 10.25 ) **(4) 6.0** Math.ceil( − 6.8 )  **(5) -6.0** Math.floor( − 5.1 ) **(6) 7.0** Math.floor( 7.9 )  **(7) 25.0** Math.pow( 5 , 2 ) **(8) 32.0** Math.pow( 2 , 5 )  **(9) 2.0** Math.max( 1.5 , 2 ) **(10) 0.5** Math. min( 3 , 0.5 )  **(11) 1.5** Math.min( 2 , 1.5 ) **(12) 3.0** Math. max( 0.5 , 3 )  **(13) 4.0** Math.sqrt( 16.0 ) **(14) 1.0** Math. round( 0.6 )  **(15) 2.0** Math.ceil( Math.pow( 3 , 0.5 ) )  **(16) 2.0** Math.floor( Math.pow( 1.5 , 2 ) )  **(17) 3.0** Math.ceil( Math.floor( 3.5 ) )  **(18)**  **2.5** Math.min( Math.max( 3.0 , 2 ), 2.5)  **(19)**  **3.0** Math.max( Math.min( 2.0 , 3 ), 3.0)  **(20)**  **2.83** Math.sqrt( Math.pow( 2.0 , 3 ) )  **(21)**  **9.0** Math.pow( Math.sqrt( 9 ) , 2 ) )  **(22)**  **3.0** Math.round( Math.max( 2.1 , 3.1 ) )  **(23) 8.0** Math.abs( Math.abs( 8.0 ) )  **(24)**  **1.5** Math.min( Math.abs( 1.5) , 2 ) )  **(25) 1.0** Math.ceil( Math.pow( 1 , 0.5 ) ) |

**Part 4 Topics in Computer Programming - Methods**

**(1)** Consider the program below. Its purpose is to calculate the population of a colony of bacteria using the formula *P* = *N* *e* *k t* in a method named population , where *P* is the population at time *t* , *N* is the initial population at time *t* = 0 , *k* is the growth rate and *e* is the base number of the natural number system, approximated as 2.718282 . Determine the missing identifiers, symbols or numbers. Write your responses in the spaces provided.

|  |
| --- |
| import java.io.\*;  import java.math.\*;  public class Population {    static BufferedReader theKeyboard = new  BufferedReader(new InputStreamReader(System.in));  public static double population(double **initial** , double rate, double time)  {  double result;  result = initial \* Math.exp(**rate** \* time);  \_\_\_\_\_\_\_\_\_\_ result;  }  public static void main(String args[]) throws IOException  {    double N, k, t, **endResult** ;  System.out.print("Find bacteria population at specified time\n");  System.out.print("Enter initial population -> ");    **N** = Double.parseDouble(theKeyboard.readLine());  System.out.print ("Enter growth rate -> ");  k = Double.parseDouble(theKeyboard.readLine());  System.out.print ("Enter number of time periods -> ");    t = Double.parseDouble(theKeyboard.readLine());  endResult = \_\_\_\_\_\_\_\_\_\_(N, k, t);    System.out.println("Population after " + **t** + " time periods = ");  System.out.println(endResult);    System.out.print ("Press < Return > to continue");  char s = (char)theKeyboard.read();  }  } |