**Multiple Choice**

1) Lines of code that the compiler ignores are called _____.
   a) comments
   b) assignment statements
   c) else statements
   d) loops
   e) built-in methods

2) The lines of code described in question 1 are useful for which of the following?
   a) Documentation
   b) Assigning variables
   c) User Interface
   d) Input/Output operations
   e) Logical Errors

3) What will be the value of the following expression?

```
25.0 / 4 + 3 * (2 + 14 / 5)
```

   a) 15
   b) 18
   c) 18.25
   d) 20.4
   e) 37

4) We want to be able to get input from a user. Which of the following must appear in the program?

```
a) Scanner
b) Input
c) User
d) nextInt()
```

5) In the for loop declaration below, `i` is an example of a(n) _____.

```
for(int i = 1; i < 100; i++)
```

   a) iterator
   b) global variable
   c) constant
   d) float
   e) invalid identifier

6) Which of the following can **NOT** be nested inside of a while loop?
   a) if statement
   b) while (another) loop
   c) for loop
   d) All of these can be nested
   e) None of these can be nested

7) The body of which of the given if statements will be executed? Assume the following values:

```
int sum = 0, total = 0;
```

```
a) if(3 < 2 && total > 0 || sum++ == total)
b) if(3 < 2 && total > 0 || ++sum == total)
c) if(total > 0 || 3 < 2 && sum++ == total)
d) if(total > 0 || 3 < 2 && ++sum == total)
```

8) How many total times will the body of the following for loop execute?

```
for(int i = 1; i < 2015; i++)
      System.out.println(i);
```

a) 0
b) 2014
c) 2015
d) 2016
e) Infinite loop

**True/False (2 points each)**:

9) You should always use single letter variables like `i, j, or k` as loop counters. F

10) A while loop can be rewritten as a for loop. T

11) An expression that evaluates to True or False is called a Boolean expression. T

12) Both the number of parameters AND the type of parameters must match the argument list in the method definition. T

13) Every if statement requires an else section. F

**Matching**

Match the numbered item on the left with its **typical** definition on the right. Note some definitions may be used more than once.

| | |
|---|---|
| 14) Compile-time error (A) | A) The code cannot be executed |
| 15) Run-time error (B) | B) The code executes but quits unexpectedly |
| 16) Logical error (C) | C) The code executes to completion but produces incorrect results |

| | |
|---|---|
| 17) `double` (B) | A) integer (whole number) |
| 18) `int` (A) | B) decimal number |
| 19) `char` (C) | C) single character of text |
| 20) `float` (B) | D) multiple characters |

| | |
|---|---|
| 21) `1way2B` (A) | A) Invalid identifier name (i.e. will cause code to not execute) |
| 22) `aVariable` (B) | B) Valid identifier name (code executes) |
| 23) `sumGrades` (B) | |
| 24) `student&class` (A) | |
| 25) `does_something` (B) | |

**Find and Fix the Error**

For the two code segments below, there is an error which will cause problems when executed. Find the error and note it by circling the location in the code. In the box below the code segment, rewrite the code correctly.

1) Assume this code segment is found inside of a main method definition

```
if(midtermGrade >= 60)
      System.out.println("Passed");
      passingStudents++;
else
      System.out.println("Did not pass");
```

Hint: Error message - 'else' without a previous 'if'

Missing curly braces means only first println will be seen as part of if statement

```
if(midtermGrade >= 60) {
      System.out.println("Passed");
      passingStudents++;
}
else
      System.out.println("Did not pass");
```

2) Assume this code segment is found inside of a main method definition

```
int i = 1;
while(i < 60);
{
      System.out.println(i);
      i++;
}
```

Hint: Blank output window and execution does not stop.

Semicolon causes infinite loop

```
int i = 1;
while(i < 60)
{
      System.out.println(i);
      i++;
}
```

**What is the Output?**

1) Determine the output produced by the following code:

```
public static void main (String args[]) {
      int i = 1, sum = 1;
      while(i < 40 && sum < 20) {
            if (i % 2 == 0) {
                  sum += i++;
                  System.out.println(i + " " + sum);
            } else {
                  System.out.println(i + " " + sum);
                  i += sum++;
            }
      }
}
```

```
1 1
3 4
3 4
7 5
13 18
13 18
31 19
```

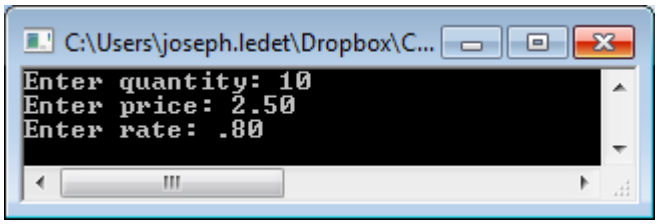
2) Determine the output produced by the following code:

```
public static void main (String args[]) {
      int k = 0;
      for(int i = 0; i < 3; i++)
            for(int j = 0; j < 3; j++)
            {
                  if(i < j)
                        k += j - i;
                  else
                        k += i - j;
                  System.out.println(i + " " + j + " " + k);
            }
}
```

```
0 0 0
0 1 1
0 2 3
1 0 4
1 1 4
1 2 5
2 0 7
2 1 8
2 2 8
```

**Short Program Fragment**

1) Write the code to produce the shown user interface. We are prompting the user for three values and storing the entered values in three variables named quantity, price, and rate. The first variable (quantity) must be a whole number (integer). The other two variables (price and rate) are to be decimal numbers.




```
int quantity;
double price, rate;
Scanner input = new Scanner(System.in);
System.out.print("Enter quantity: ");
quantity = input.nextInt();
System.out.print ("Enter price: ");
price = input.nextDouble();
System.out.print ("Enter rate: ");
rate = input.nextDouble();
```

2) As you may be aware, mobile phone usage is paid by the minute with each partial minute being treated as a full minute (i.e. if you talk for 4 minutes and 2 seconds, you will be charged for 5 minutes). You have been given the task to take an integer variable called "`seconds`" being the amount of time talked in seconds and calculating the amount of time that should be charged. Write a program fragment (***no declarations, println, Scanner, etc.***) that will take the value of *`seconds`* and assign the appropriate value of how many minutes to charge to the integer variable (already declared) *`minutes`*.
Hint: There are multiple ways to do this; the easiest involve integer division. All simple solutions to this problem will fit in the box below.

```
minutes = seconds / 60;
if (seconds % 60 != 0)
      minutes++;

Alternative solution involving loop:
minutes = 0;
for(int i = seconds; i > 0; i -= 60)
      minutes++;
```

**Develop a Solution**

A leap year is defined as divisible by 4, but either not also divisible by 100 or also divisible by 400.
So, 2016 **was** a leap year because it was divisible by 4.
1900 **was not** a leap year because while being divisible by 4, it was also divisible by 100.
2000 **was** a leap year because it was divisible by 400.
You will write a method that will take an integer parameter representing a year. Your method will return true if it is a leap year and false otherwise.

1) Identify two of the smaller sub-problems

How to create a method
- Return type
- How many parameters
- Type of parameters

How to determine if it is divisible by:
- 4
- 100
- 400

1. _____
2. _____

2) Plan how to solve the problem by writing pseudocode **OR** drawing a flowchart

```
Procedure boolean leapYear(n: Integer):
     If year divisible by 400, return True
     Otherwise, if year divisible by 100, return False
     Otherwise, if year divisible by 4 return True
     Otherwise, return False
```

3) Write the Java code for your solution

```
public static bool leapYear(int year){
     if(year % 400 == 0)
          return true;
     else if(year % 100 == 0)
          return false;
     else
          return year % 4 == 0;
}
```

**Bonus**

**Factorial**

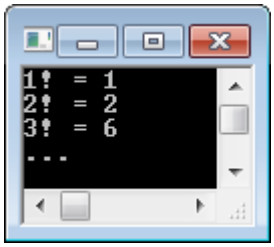
n! is defined as the product of all whole numbers from 1 to n including n.

3! = 3 * 2 * 1 = 6

5! = 5 * 4 * 3 * 2 * 1 = 120

Write a method called “factorial” that takes an integer argument (n) and returns an integer representing the value of n! (i.e. factorial(0) should return 1, factorial(1) should return 1, etc.). Fill in the blanks below, and write your factorial method in the space provided.

After you write your factorial method, write a main program that will use your method to display the first 12 factorial numbers each on a new line in the format of:




```
__________ factorial(__________ n)
```

```
public static int factorial (int n){
      int f = 1;
      for(int i = 2; i <= n; i ++)
            f *= i;
      return f;
}

public static void main(String args[]) {
      for(int i = 1; i <= 12; i++)
            System.out.println(i + “! = “ + factorial(i));
}
```