Junior Knights Programming Assignments - Loop Practice

Part A: Copy Machine (copy.java)
Pick your favorite message and ask the user how many times they would like it printed out. Then print it out that many times, in a numbered list, starting at 1.

Sample Program Runs
Note: Computer output is in plain text while the user's input is in bold for these examples.

Sample Run #1
```
How many times do you want the message printed?
5
Go Knights!
Go Knights!
Go Knights!
Go Knights!
Go Knights!
```

Sample Run #2
```
How many times do you want the message printed?
2
Go Knights!
Go Knights!
```

Note: You may choose whatever non-offensive message you like to print.

Part B: New Year's Count Down (newyears.java)
Ask the user for how many seconds they want to count down before new year's and print out a countdown from that number to 1, followed by printing, "Happy New Year!!!"

Sample Run
```
How many seconds do you want in your countdown?
10
10 9 8 7 6 5 4 3 2 1
Happy New Year!!!
```

Part C: Printing Arithmetic Sequences (sequence.java)

An arithmetic sequence is defined by a first term, a common difference and a number of terms. For example, an arithmetic sequence with the first term 13, a common difference of 4 and 6 terms is 13, 17, 21, 25, 29 and 33. For this program, you'll prompt the user to enter these three values and your program will print out each term in the arithmetic sequence.

Sample Run #1

```
What is the first term of your sequence?
13
What is the common difference of your sequence?
4
How many terms are there in your sequence?
6
Here is the sequence: 13 17 21 25 29 33
```

Sample Run #2

```
What is the first term of your sequence?
-4
What is the common difference of your sequence?
3
How many terms are there in your sequence?
10
Here is the sequence: -4 -1 2 5 8 11 14 17 20 23
```

Part D: Summing Arithmetic Sequences (sumseq.java)

For this program, you'll prompt the user to enter the same values as part C, but this time your program will print the sum of the numbers as well.

Sample Run #1

```
What is the first term of your sequence?
13
What is the common difference of your sequence?
4
How many terms are there in your sequence?
6
Here is the sequence: 13 17 21 25 29 33
The sum of the sequence is 138.
```

Sample Run #2

```
What is the first term of your sequence?
-4
What is the common difference of your sequence?
3
How many terms are there in your sequence?
10
Here is the sequence: -4 -1 2 5 8 11 14 17 20 23
The sum of the sequence is 95.
```

Part E: Radioactive Decay (decay.java)

One of the ways in which we determine the age of old artifacts is through Carbon-14 dating. In particular, the number of Carbon-14 atoms in a specimen divides by two every 5730 years. Thus, if a specimen starts with 1000000 Carbon-14 atoms, 5730 years later, there will be about 500000 left and 11460 years later there will be about 250000 Carbon-14 atoms left.

In this program, you'll ask the user how many particles of the measured atom the specimen started with, the half-life of that atom, and for how many half-lives you want a chart printed. With this, you'll print out a chart labeled with the years and the number of particles, for each increment of the half-life. To calculate the number of atoms at each iteration, use integer division. See the sample runs for clarification.

Sample Run #1

```
How many particles of your atom were in the original sample?
1000000
What is the half life of your atom?
5730
How many lines of the chart do you want printed?
5
Years   Particles
0       1000000
5730    500000
11460   250000
17190   125000
22920   62500
```

Sample Run #2

```
How many particles of your atom were in the original sample?
135
What is the half life of your atom?
20
How many lines of the chart do you want printed?
8
Years   Particles
0       135
20      67
40      33
60      16
80      8
100     4
120     2
140     1
```

Part F: Perfect, Abundant and Deficient Numbers (numbers.java)

Write a program to take in a positive integer n > 1 from the user and print out whether or not the number the number is a perfect number, an abundant number, or a deficient number. A perfect number is one where the sum of its proper divisors (the numbers that divide into it evenly not including itself) equals the number. An abundant number is one where this sum exceeds the number itself and a deficient number is one where this sum is less than the number itself. For example, 28 is perfect since 1 + 2 + 4 + 7 + 14 = 28, 12 is abundant because 1 + 2 + 3 + 4 + 6 = 16 and 16 is deficient because 1 + 2 + 4 + 8 = 15.

Sample Run #1

```
Enter a positive integer n > 1.
12
The sum of the proper divisors of 12 is 16.
12 is an abundant number.
```

Sample Run #2

```
Enter a positive integer n > 1.
28
The sum of the proper divisors of 28 is 28.
28 is a perfect number.
```

Sample Run #3

```
Enter a positive integer n > 1.
15
The sum of the proper divisors of 15 is 9.
9 is a deficient number.
```

Part G: Triangle (triangle.java)

Write a program that asks the user to enter a positive integer n and then prints out a triangle with n rows, where the first row has 1 star, the second row 2 stars, etc.

Sample Run #1

```
Enter a positive integer.
4
Here is your 4 row triangle:
*
**
***
****
```