Exam 1 Practice Assignment

Overview

This is a set of problems to use to prepare for Exam 1. **It is not a "practice exam".** The actual exam will not be this length or have this many questions.

You will NOT hand in any files. This assignment is only practice for Exam 1.

For each problem, you should:

- determine the inputs, outputs, and error conditions
- construct an algorithm (in the form of a flowchart)
- code your algorithm in Java
- construct a set of test cases and use them to test **and fix** your program

Problems

Problem 1

Develop a Java program that computes which bills and coins will be given as change for a sales transaction.

Available bills are:

- $20
- $10
- $5
- $1

Available coins are:

- quarter (25 cents)
- dime (10 cents)
- nickel (5 cents)
- penny (1 cent)

Only print the actual bills and coins returned. Do not print unused denominations (for example, if no $10 bills are returned, do not print anything regarding $10 bills).

Example:

```
What is the total sales charge?
58.12

How much is the customer giving?
100.00

The change is:
2 $20 bills
1 $1 bill
3 quarters
1 dime
3 pennies
```

Problem 2

Your program should compute and print the user's cellphone bill. Charges on the bill should include:

- A base monthly fee.
- Overage charges: when the number of free minutes per month is exceeded, the customer is charged at a rate per minute over.

Your program should not make any assumptions about rates, fees, etc. - get all information from the user.

Problem 3

Your program should determine whether a year is a leap year or not. Ask the user for the year they wish to investigate (an integer). The following rules govern the designation of leap years:

1. Years not divisible by 4 are not leap years.
2. Years divisible by 4 are leap years, **except that...**
3. Years divisible by 100 are not leap years, **except that...**
4. Years divisible by 400 are leap years.

Output true if the specified year is a leap year, and false if it is not.

Problem 4

We wish to develop a program that will count the number of even and odd integers in a set ("even" meaning divisible by 2, "odd" meaning not divisible by 2). We will use zero as an indicator that the set has been completely entered, and this zero should not be counted as part of the set. Ask the user for a sequence of integers, terminated by zero. Output the number even integers and the number of odd integers.

Problem 5

The user puts some money into a savings account. Each month, they might (or might not) make a deposit into that account. The amount of the deposit can vary from month to month. They will not make any withdrawals. Interest is paid on the account once a month.

Your program should determine how long it will take for the savings account to double in worth from its initial value.

Problem 6

Write a program that allows a teacher to compute the average grades for EACH student in his or her class (NOT a single average for the entire class).

Problem 7

A magic number is defined as a positive integer which is equal to the sum of its own factors (except itself). For example, 6 is a magic number because its factors (other than 6 itself) are 1, 2, and 3, and these factors sum up to 6. Write a program that, for any upper limit, computes all magic numbers less than that limit. For the purposes of this problem, assume that Java variables can store numbers of any size.