(Basic) Methods, Math, Strings, Conditionals

Important Dates:

- Assigned: August 21, 2023
- Deadline: September 4, 2023 at 11:59 PM EST

Objectives:

- Students learn to use basic Java concepts involving different datatypes.
- Students gain experience with the Java mathematics library.
- Students work with certain string manipulation functions.
- Students begin to undestand the different types of conditional statements and how they redirect program control flow.
- Students design methods to complete a task and write corresponding unit tests.

What To Do:

For each of the following problems, create a class named `ProblemX`, where `X` is the problem number. E.g., the class for problem 1 should be `Problem1.java`. Write (JUnit) tests for each method that you design in corresponding test files named `ProblemXTest`, where `X` is the problem number. Additionally, write Javadoc comments explaining the purpose of the method, its parameters, and return value.

You must write sufficient tests and adequate documentation. As Ken Shan says, "don't bullshit".

Problem 1

A grocery store sells apples, ears of corn, and bananas (this is a “specialized” grocery store). The price of each item is given in the following table:

Item	**Price**
Apple	\$0.49
Ear of Corn	\$1.95
Banana	\$0.39

Write the `purchase` method, which receives a quantity of apples, ears of corn, and bananas. This method should return the total cost.

Problem 2

The dew point temperature T_d can be calculated (approximately) from the relative humidity RH and the actual temperature T by

$$T_d = \frac{b \cdot f(T, RH)}{a - f(T, RH)}$$
$$f(T, RH) = \frac{a \cdot T}{b + T} + \ln RH$$

where $a = 17.27$ and $b = 237.7°C$.

Write the `dewPoint` method, which receives a relative humidity value (between zero and one) and a temperature in degrees Celsius. It should return the dew point value. Use the Java `Math.log` to compute the natural logarithm.

Problem 3

Write the `numberToMonth` method, which transforms numbers $1, 2, 3, \ldots, 12$ into the corresponding month names January, February, March, . . ., December. There are a few ways you can do this, but *you have to do it without any kind of selection statement*. Hint: use `substring` and padding with spaces!

Problem 4

Easter Sunday is the first Sunday after the first full moon of spring. To compute the date, you can use this algorithm, invented by the mathematician Carl Friedrich Gauss in 1800:

Let y be the year. Divide y by 19 and call the remainder a. Ignore the quotient. Divide y by 100 to get a quotient b and a remainder c. Divide b by 4 to get a quotient d and a remainder e. Divide $8 \cdot b + 13$ by 25 to get a quotient g. Ignore the remainder. Divide $19 \cdot a + b - d - g + 15$ by 30 to get a remainder h. Ignore the quotient. Divide c by 4 to get a quotient j and a remainder k. Divide $a + 11 \cdot h$ by 319 to get a quotient m. Ignore the remainder. Divide $2 \cdot e + 2 \cdot j - k - h + m + 32$ by 7 to get a remainder r. Ignore the quotient. Divide $h - m + r + 90$ by 25 to get a quotient n. Ignore the remainder. Divide $h - m + r + n + 19$ by 32 to get a remainder P. Ignore the quotient.

Then, Easter falls on day P of month n. For example, if y is 2001:

```
a = 6, b = 20, c = 1, d = 5, e = 0,  g = 6, h = 18
j = 0, k = 1,  m = 0, n = 4, p = 15, r = 6
```

Therefore, in 2001, Easter Sunday fell on April 15. Write the `stringifyDate` method, which receives a year and returns a string containing the month and day of Easter Sunday in the corresponding year. The format of the resulting string should be `Month Day, Year`. E.g,.

```
April 15, 2001
```

Hint: for retrieving the month, use the algorithm you wrote in Problem 3 and the `strip` method from the `String` class.)

Problem 5

Write the `isEvenlySpaced` method, which receives three integers and returns whether they are evenly spaced. Evenly spaced means that the difference between the smallest and medium number is the same as the difference between the medium and largest number.

Problem 6

In propositional logic, there are several *connectives* that act on boolean truth values. These include logical conjunction $\wedge$, disjunction $\vee$, conditional $\rightarrow$, biconditional $\leftrightarrow$, and negation $\neg$. We can represent *schemata* as a series of composed method calls. For example, an evaluation of

$$`P \rightarrow \neg(Q \leftrightarrow \neg R)'$$

where 'P' and 'R' are assigned to `false` and 'Q' is assigned to `true`, is equivalent to

```
public static final boolean P = false;
public static final boolean Q = true;
public static final boolean R = false;

cond(P, not(bicond(Q, not(R))))
```

The presented schema resolves to `true`.

Design methods for the five connectives according to the following truth tables. Assume that $\top$ is `true` and $\bot$ is `false`.

P	$\neg P$
$\top$	$\bot$
$\bot$	$\top$

Figure 1: Truth Table of '$\neg P$'

P	Q	$P \wedge Q$
$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$
$\bot$	$\top$	$\bot$
$\bot$	$\bot$	$\bot$

(a) Truth Table of '$P \wedge Q$'.

P	Q	$P \vee Q$
$\top$	$\top$	$\top$
$\top$	$\bot$	$\top$
$\bot$	$\top$	$\top$
$\bot$	$\bot$	$\bot$

(b) Truth Table of '$P \vee Q$'.

P	Q	$P \to Q$
$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$
$\bot$	$\top$	$\top$
$\bot$	$\bot$	$\top$

(a) Truth Table of '$P \to Q$'.

P	Q	$P \leftrightarrow Q$
$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$
$\bot$	$\top$	$\bot$
$\bot$	$\bot$	$\top$

(b) Truth Table of '$P \leftrightarrow Q$'.

Problem 7

Write the `quadraticRoots` method, which receives values for a, b, and c, and computes the roots using the quadratic formula:

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

The method should return a formatted string, whose value depends on the number of roots. If there are no roots, return the string `"No roots"`. If there is one root, return a string containing the one root prefixed with `"x="`, e.g., `"x=5.25"`. If there are two roots, return a string containing the two roots prefixed with `"x1="` and `"x2="` respectively, separated by a space. E.g., `"x1=-.270033 x2=-1.58711"`.

Problem 8

Write the `insideCircle` method that, when given a circle centered at $(c_x,\ c_y)$ and radius r as well as a point at $(p_x,\ p_y)$, returns whether the point is located strictly inside the circle.

Write the `insideRectangle` method that, when given a rectangle centered at $(r_x,\ r_y)$, width w and height h as well as a point $(p_x,\ p_y)$, returns whether the point is located strictly inside the rectangle.

Problem 9

Carlo is shipping out orders of candy to local grocery stores. Boxes have a maximum weight defined by `maxSize`, and we can (potentially) fit both small and large bars of candy in a box. Write the `fitCandy` method that, when given a number of small bars, large bars, and maximum weight, determines the number of small candy bars he can fit in the box. Large bars weigh five kilograms, and small bars weigh one kilogram. Note that Carlo always tries to fit large candies first before small. Return `-1` if it is impossible to fill the box with the given criteria. Below are some test examples. Hint: consider this as an analysis of three cases.

```
fitCandy(4, 1, 9) -> 4
fitCandy(4, 1, 4) -> 4
fitCandy(1, 2, 6) -> 1
fitCandy(6, 1, 13) -> -1
fitCandy(60, 100, 550) -> 50
fitCandy(7, 1, 12) -> 7
fitCandy(7, 1, 13) -> -1
```

Problem 10

An IPv4 address contains four integer values stored in four octets, separated by dots. For instance, `192.168.1.244` is a valid IPv4 address. Another example is `149.165.192.52`. Design the `isValidIpv4` method that, when given a `String`, determines whether it represents a valid IPv4 address. Each octet must be an integer between zero and 255 inclusive. Note that some IPv4 addresses are, in reality, nonsensical, e.g., `0.0.0.0`, but we will not consider these as invalid.

```
isValidIpv4("192.168.1.244") -> true
isValidIpv4("149.165.192.52") -> true
isValidIpv4("192.168.1.256") -> false
isValidIpv4("192.168.1201.23") -> false
isValidIpv4("192.168.1201.ABC") -> false
isValidIpv4("ABC.DEF.GHI") -> false
isValidIpv4("192.168.1A6.201") -> false
```