Introduction

Welcome to the Intro to Java: Functional Programming, Lesson 2 problem set! These problem sets are an opportunity for you to practice the concepts you learned in class before moving on to the next lesson. Learning a computer programming language is similar to learning a human language. Nobody can pick it up overnight, there's a lot of vocabulary and syntax to remember. Language learners often speak of the moment when they realized they stopped translating in their head and actually started thinking in their second language. This will happen with Java too! Eventually, you will be able to consider a task that needs coding and immediately imagine what Java code would complete it. To get there, though, requires practice.

That's where the problem sets come in. They aren't mandatory, and they aren't graded. They're just extra learning materials to help you along.

Completing the Problem Sets

There isn't a right or wrong way to work on these. Some problems require you to examine code or do some arithmetic. You can take notes on paper, print this document and use the space provided, or try to do it all in your head—whatever works for you. For the exercises that require programming, we highly recommend that you pick your favorite text editor, open a blank text file, and try writing out the code.

Question 1

What will the following block of Java code print?

```
double balance = 0;
balance = balance + 20; //Add quarter 1 profits (thousands).
balance = balance - 25; //Subtract quarter 1 expenses (thousands).
balance = balance + 30; //Add quarter 2 profits (thousands).
balance = balance - 25; //Subtract quarter 1 expenses (thousands).

if (balance < 0) {
    System.out.println("We're in the red!");
} else if (balance > 0) {
    System.out.println("We made a profit!");
} else {
    System.out.println("We broke even.");
}
```

A. `We're in the red!`

B. `We made a profit!`

C. `We broke even.`

Question 1 Solution

C. We broke even.

The variable `balance` began at 0. 20 was added, then 25 subtracted, bringing the value to -5. Then 30 was added, bringing the value to 25. Finally, 25 was subtracted, bringing the value to 0. Since the conditions `balance < 0` and `balance > 0` are both false, the else case is the one that is executed.

Question 2

What will be printed by the block of Java code below?

```
int dogs = 1;
int cats = 2;

if (dogs > 0 && cats == 0) {
    if (dogs > 1) {
        System.out.println("Dog lover");
    } else {
        System.out.println("Dog person");
    }
} else if (cats > 0 && dogs == 0) {
    if (cats > 1) {
        System.out.println("Cat lover");
    } else {
        System.out.println("Cat person");
    }
    System.out.println("Meow!");
} else if (cats > 0 && dogs > 0) {
    if (dogs > cats) {
        System.out.println("I guess you like dogs more");
    } else if (dogs == cats) {
        System.out.println("I guess you like both equally");
    } else {
        System.out.println("I guess you like cats more");
    }
} else {
    System.out.println("What, don't you like pets?");
}
```

Question 2 Solution

I guess you like cats more

```
int dogs = 1;
int cats = 2;

//This condition is evaluated first.  Since it is false, Java
//will skip to the first "else if."
if (dogs > 0 && cats == 0) {
    if (dogs > 1) {
        System.out.println("Dog lover");
    } else {
        System.out.println("Dog person");
    }
}
//This condition is evaluated second.  It is also false, so Java
//will skip to the second "else if."
else if (cats > 0 && dogs == 0) {
    if (cats > 1) {
        System.out.println("Cat lover");
    } else {
        System.out.println("Cat person");
    }
    System.out.println("Meow!");
}
//This condition is true, so the contents of this "else if"
//block will be executed.
else if (cats > 0 && dogs > 0) {
    //This condition is false, so Java will skip to the "else
    //if."
    if (dogs > cats) {
        System.out.println("I guess you like dogs more");
    }
    //This condition is also false.  Since there are no more
    //"else if" blocks, the "else" block will be executed
    //automatically.
    } else if (dogs == cats) {
        System.out.println("I guess you like both equally");
    } else {
        //This print statement is executed!
        System.out.println("I guess you like cats more");
    }
}
//This else block is skipped.  No more than one if/else if/else
//block is executed per if statement.  Since the "else if" block
//above was executed, this block is skipped.
```

```
else {
    System.out.println("What, don't you like pets?");
}
```

Question 3

You are programming the behavior of an enemy in a video game. The enemy code has access to two boolean variables, `canSeePlayer` and `playerPoweredUp`. `canSeePlayer` is true when the enemy can see the player and false otherwise, and `playerPoweredUp` is true when the player has found a special item that makes them impossible to defeat temporarily. Fill in the correct conditions so that "Attack!" is printed when the enemy can see the player and the player is *not* powered-up, "Run away!" is printed when the player is visible and *is* powered-up, and finally, "Wander." is printed if the player is not visible.

Starting code:

```
if (    -1-    ) {
    if (    -2-    ) {
        System.out.println("Attack!");
    } else {
        System.out.println("Run away!");
    }
} else {
    System.out.println("Wander.");
}
```

Question 3 Solution

```
if (canSeePlayer) {
    if (!playerPoweredUp) {
        System.out.println("Attack!");
    } else {
        System.out.println("Run away!");
    }
} else {
    System.out.println("Wander.");
}
```

Remember, you can use a boolean variable as a condition directly. You do not need to write `if (canSeePlayer == true)`, you can just write `if (canSeePlayer)`.

Question 4

Assume you have access to two boolean variables, `isSnowing`, and `isRaining`, and one double variable, `temperature`. `isSnowing` is true when it's snowing and false otherwise, `isRaining` is true when it's raining and false otherwise, and temperature gives the outdoor temperature in degrees Fahrenheit. Write code that prints "Let's stay home." if it's raining, snowing, or below 50 degrees Fahrenheit (10 degrees Celsius), and prints "Let's go out!" otherwise.

Starting code:

```
//Assume these can have any value:
boolean isSnowing = false;
boolean isRaining = true;
double temperature = 60.0;

//TODO: print "Let's stay home." if its raining, snowing or
//below 50 degrees and print "Let's go out!" otherwise.
```

Question 4 Solution

Example solution code:

```
boolean isSnowing = false;
boolean isRaining = true;
double temperature = 60;

if (isRaining || isSnowing || temperature < 50) {
    System.out.println("Let's stay home.");
} else {
    System.out.println("Let's go out!");
}
//With these values, this code would print "Let's stay home."
```

Question 5

Assume you have access to a double variable called `time`. Write code that assigns a different value to the String variable `timeOfDay` based on the value of `time` in hours. If `time` is between 5 and 12, including 5 but not including 12, set `timeOfDay` to "morning". If `time` is between 12 and 20, including 12 but not including 20, set `timeOfDay` to "daytime". Finally, if the `time` variable does not satisfy either condition, set `timeOfDay` to "night".

Starting code:

```
//Assume this could have any value between 0 and 24:
int time = 18;

String timeOfDay;
//TODO: set timeOfDay to the correct String value.
```

Question 5 Solution

Example solution code:

```
//Assume this could have any value between 0 and 24:
int time = 18;

String timeOfDay;

if (time >= 5 && time < 12) {
    timeOfDay = "morning";
} else if (time >= 12 && time < 20) {
    timeOfDay = "daytime";
} else {
    timeOfDay = "night";
}
```

Question 6

Assume you have access to an integer variable called `dayOfTheWeek` and a boolean variable called `holiday`, which is true when it is a holiday and false on normal days. Write Java code that prints "Wake up at 7:00" on weekdays that are not holidays, and prints "Sleep in!" on weekends and holidays. For the variable `dayOfTheWeek`, use this key:
1 = Monday
2 = Tuesday
3 = Wednesday
4 = Thursday
5 = Friday
6 = Saturday
7 = Sunday

Starting code:

```
//Assume these could have any value:
int weekday = 5;
boolean holiday = false;
```

Question 6 Solution

Example solution code:

```
//Assume these could have any value:
int weekday = 5;
boolean holiday = false;

if (weekday >= 1 && weekday <= 5 && !holiday) {
    System.out.println("Wake up at 7:00");
} else {
    System.out.println("Sleep in!");
}
```

Question 7

Find the error in this Java code. (Hint: think about scope!)

Imagine that the variables `rewinding` and `playbackPosition` could have different values depending on the situation.

```
double playbackPosition = 120;
boolean rewinding = true;

if (rewinding) {
    double rewindAmount = 0.1;
}
playbackPosition = playbackPosition - rewindAmount;
```

Question 7 Solution

The line

```
playbackPosition = playbackPosition - rewindAmount;
```

references a variable out of scope. When a variable is defined inside an if block, such as

```
if (rewinding) {
    double rewindAmount = 0.1;
}
```

that variable is *only available inside the if block*. This is called the *scope* of the variable `rewindAmount`. Since the contents of an if block may or may not be evaluated, we cannot be sure after the if block that the variable `rewindAmount` was ever defined. For this code to work, the last line must be moved inside the if block:

```
double playbackPosition = 120;
boolean rewinding = true;

if (rewinding) {
    double rewindAmount = 0.1;
    playbackPosition = playbackPosition - rewindAmount;
}
```

Question 8

Assume the int variable `dayOfWeek` has some value, and that it follows this rule:
1 = Monday
2 = Tuesday
3 = Wednesday
4 = Thursday
5 = Friday
6 = Saturday
7 = Sunday

Declare a String variable called `schedule` and write a switch statement that gives schedule a different value based on the day of the week. For Monday, give it the value "Gym in the morning." For Tuesday give it the value "Class after work." For Wednesday, give it the value "Meetings all day." For Thursday give it the value "Work from home." For Friday, give it the value "Game night after work." For Saturday and Sunday, give it the value "Free!" This should also be the default value.

OR (optional): decide what to assign schedule for each day based on your own weekly schedule!

Starting code:

```
//Assume this can have any value from 1 to 7:
int dayOfWeek = 1;

//TODO: declare a String variable called schedule.

//TODO: write a switch statement that give schedule a different
//value for each day of the week based on the dayOfWeek variable.
//Don't forget to "break" after each case, and don't forget to
//provide a default case!
```

Question 8 Solution

Example solution code:

```
//Assume this can have any value from 1 to 7:
int dayOfWeek = 1;

String schedule;

switch (dayOfWeek) {
    case 1:
        schedule = "Gym in the morning.";
        break;
    case 2:
        schedule = "Class after work.";
        break;
    case 3:
        schedule = "Meetings all day.";
        break;
    case 4:
        schedule = "Work from home.";
        break;
    case 5:
        schedule = "Game night after work.";
        break;
    //Tip: when you have several identical cases, you can
    //list them together like this, or you could just list
    //"default" and let it catch any case not covered
    //above.
    case 6: case 7: default:
        schedule = "Free!";
        break;
}
```