Exercise 1: Julia’s Final Exam (Simple computations)

Your friend Julia is studying to be a chef at the Cordon Bleu in Paris. For their final exam, each student must prepare 2 appetizers, 2 main dishes, and 1 dessert. Each is graded on a scale from 1 to 10, but each main dish counts twice as much as each appetizer, and the dessert counts twice as much as each main dish (so a perfect grade is 100: 10 each for the 2 appetizers, 20 each for the 2 main dishes, and 40 for the dessert).

Part (a): Start by writing a Java program to calculate final exam grades. Use the code shown below (which is also on the laptops) and fill in the four blanks.

```java
import java.util.Scanner;
public class ExamGrades {
    public static void main(String[] args) {
        // create a Scanner to read user input
        Scanner sc = new Scanner(System.in);

        // ask the user to type in the scores
        System.out.println("Enter scores for each of the 2 appetizers," + " one per line.");
        int appetizer1 = sc.nextInt();
        int appetizer2 = sc.nextInt();
        System.out.println("Enter scores for each of the 2 main dishes," + " one per line.");
        int mainDish1 = sc.nextInt();
        int mainDish2 = sc.nextInt();
        System.out.println("Enter score for the dessert: ");
        int dessert = sc.nextInt();

        // now compute the final grade and print it
        int appScore = __________________________; // Compute the total score for appetizers
        int mainScore = ___________________________; // Compute the total score for main dishes
        int dessertScore = ___________________________; // Compute the total score for the dessert
        int grade = ___________________________; // Compute the final grade

        System.out.println("Final grade: " + grade);
    }
}
```

Part (b): The teacher changed the final exam! Now each student must prepare 3 appetizers. They’re still graded from 1 to 10, but the lowest score isn’t used to compute the final grade. Change your program to match this new scheme. One way to do this is to use a new variable to hold the lowest score. Figure out what value to assign to that variable as the three scores are typed in; add all three scores, then subtract the lowest one.
Exercise 2: Baking Java Style (Methods that Return Values)

So far, the method calls we’ve used all performed actions. A method can also compute and return a value. For this exercise, we’ll assume we have three objects named brownies, cookies, and fudge, each of which represents the recipe for a chocolate dessert. Each of these objects has the methods listed below. These methods compute and return integer values, so the descriptions all start with the special word int.

```
int numEggs(int numBatches)
    Returns the number of eggs needed to make numBatches batches of this dessert.

int numCupsFlour(int numBatches)
    Returns the number of cups of flour needed to make numBatches batches of this dessert.

int numOzChocolate(int numBatches)
    Returns the number of ounces of chocolate needed to make numBatches batches of this dessert.
```

Part (a).

The following code computes the number of eggs needed to make 1 batch each of brownies, cookies, and fudge.

```
int eggsForBrownies, eggsForCookies, eggsForFudge, totalEggs;
eggsForBrownies = brownies.numEggs(1);
eggsForCookies = cookies.numEggs(1);
eggsForFudge = fudge.numEggs(1);
totalEggs = eggsForBrownies + eggsForCookies + eggsForFudge;
```

First, we’ll act out the execution of the above code by having 3 people play the roles of the brownies, cookies, and fudge objects. Your Team Leader will give each of them their recipes. One more person will be in charge of tracing the code on the blackboard. Start by writing the four variable names eggsForBrownies, eggsForCookies, eggsForFudge, and totalEggs with a box after each one to show its value. Then act out each line of code. For example, act out the line eggsForBrownies = brownies.numEggs(1); by asking the person playing the role of the brownies object “How many eggs are needed for 1 batch?” Write the answer in the box labeled eggsForBrownies. Then go on to the next line of code.
Part (b).

What Java code (similar to the code from Part (a)) could we use to find out how many cups of flour are needed to make two batches of fudge, three batches of brownies, and four batches of cookies? Write the code and act it out.

(Are you getting hungry? Don’t worry, you’ll get to eat a chocolate dessert pretty soon...)

Part (c).

Now let’s assume that we have an object called onHand that represents the ingredients we have in the house. The onHand object has the following method:

```java
int numUnits( String ingredientName )
```

Returns the number of units (cups, ounces, or whatever is appropriate) of the given ingredient that we have in the house.

For example, the expression `onHand.numUnits( "eggs" )` tells us how many eggs are in the house.

To make our desserts, we also need an object called chef, with a bake method. That method has one argument: the recipe to bake. For example, to make a batch of brownies we’d use this code:

```java
chef.bake( brownies );
```

Executing this code would cause the chef to bake one batch of brownies.

Suppose we want to make one batch of cookies. We have plenty of flour and chocolate in the house, but we’re not sure about eggs. How could we do the following:

1. Find out how many eggs are needed, and store that value in a variable.
2. Find out how many eggs we have in the house, and store that value in another variable.
3. Compare the number of eggs needed with the number of eggs in the house. If there are enough eggs in the house, bake one batch of cookies.

Hint: Use a Java if-statement or write an algorithm that uses if control flow.

Part (d).

Now suppose we want to make as many batches of cookies as possible. This time, chocolate is the ingredient in short supply. What Java code could we use to find out how many batches of cookies we can make? (Hint: If we have $N$ ounces of chocolate in the house, and each batch of cookies requires $M$ ounces of chocolate, then what expression gives the number of batches of cookies we can make?)
Exercise 3: Tell me it Ain’t True! (Booleans)

Most computer programs do different things depending on what conditions hold (for example, what the user types, or what data is read from a file). Programmers use Boolean formulas to control what their programs do.

A Boolean formula always evaluates to either true or false. Below are 4 simple Boolean formulas (using English sentences, not Java code). Say which are true and which are false.

- today is Tuesday
- (today is Tuesday) or (yesterday was Wednesday)
- (today is Wednesday) and (we are in Madison)
- (today is Monday) or (we are not in Chicago)

In a Java program, instead of using the words “and”, “or”, and “not” we use these symbols:

& & means and
| | means or
! means not

Here are the 4 formulas again, this time using symbols instead of words for and, or, and not:

- today is Tuesday
- (today is Tuesday) | | (yesterday was Wednesday)
- (today is Wednesday) & & (we are in Madison)
- (today is Monday) | | (! (we are in Chicago))

Part (a).

Below are 4 Boolean formulas that you might find in a Java program. (Writing (x == 0) means “x is equal to zero”.)

- (x == 0) | | (x == 1)
- (x < y) & & (x > 0)
- (y == 0) | | (y > 10)
- ! (x == 0) | | ! (y == x)

Suppose that x has the value 0 and y has the value 20. Work with a partner to decide which of the above formulas are true and which are false. When you’ve decided, compare your answers with another pair. If you don’t agree, talk together to figure out who is right.
Part (b).

If you use more than one of the symbols (&&, | |, !) in a formula, then you may need to know their precedences, just like you need to know the precedences of the arithmetic operators plus and times to know how to evaluate 1 + 2 * 3 (where * means times). Do you understand why?

Here are the precedences for the Boolean operators:

- ! highest precedence
- && next highest precedence
- | | lowest precedence

Here are 2 formulas that use more than one symbol and are very similar:

\[
(x==1) && (y==2) || (x<y) \\
(x==1) && ((y==2) || (x<y))
\]

If x is 0 and y is 20, what are the values of the two formulas?

Part (c).

Practice evaluating Boolean formulas by playing a game. Divide into teams of 1, 2, or 3. Each team takes 5 yellow cards and 3 green cards and uses them (plus as many parenthesis cards as you want) to make the trickiest Boolean formula they can. Put your formula on the table so everyone can see it. Each team writes down the value of all of the formulas (including their own). Once everyone has their answers, go around and figure out the correct answer for each formula. The team that gets the most right wins!
Exercise 4: More Fun with Words (Tracing Algorithms)

Last week we did an exercise with an object called word. We’ll use the word object again today, with a similar list of operations:

\[
\text{moveToFront( int pos )}
\]
Move the letter in position \(pos\) to the beginning of the word (i.e., to position zero).

\[
\text{moveToEnd( int pos )}
\]
Move the letter in position \(pos\) to the end of the word.

\[
\text{swap( int pos1, int pos2 )}
\]
Swap the letter in position \(pos1\) with the letter in position \(pos2\).

\[
\text{reverse( int start, int finish )}
\]
Reverse the order of the letters in positions \(start\) to \(finish\).

In Java, these operations are called \textit{methods}, and code like \texttt{word.swap(0, 1)}, is called a \textit{method call}: the \texttt{swap} method of \texttt{word} is being called to rearrange the letters in the sequence of letters that \texttt{word} represents.

Part (a).
First, make sure you remember what the \texttt{word} operations do; work with a partner to fill in the third column below. Remember that the first letter of the word is in position zero.

<table>
<thead>
<tr>
<th>Original word</th>
<th>Java code</th>
<th>word after the code executes</th>
</tr>
</thead>
<tbody>
<tr>
<td>HIPSH</td>
<td>\texttt{word.moveToFront(2)}</td>
<td></td>
</tr>
<tr>
<td>ZOOLOGY</td>
<td>\texttt{word.moveToEnd(0)}</td>
<td></td>
</tr>
<tr>
<td>PICKLES</td>
<td>\texttt{word.swap(0,6)}</td>
<td></td>
</tr>
<tr>
<td>AVOCADO</td>
<td>\texttt{word.reverse(4,6)}</td>
<td></td>
</tr>
</tbody>
</table>
Part (b).

The process of keeping track of what happens when a piece of code is executed is called *tracing*. Tracing can be tedious, but it is a valuable skill that helps programmers understand and fix code. We’ll do some tracing of method call sequences for practice.

Work in groups of two or three, racing against the other groups. Your Team Leader will give each group some cards with letters on them. Start by arranging them to spell *debit-card*. Then you’ll get a sequence of method calls. Your job is to rearrange your letters to show the effect of each call. When you have the final result, show it to your Team Leader. Don’t say it out loud or the other teams might hear you!

Part (c).

Now your Team Leader will give each group a new starting value for *word* and a worksheet. The worksheet will have a new sequence of method calls, and space to keep track of the value of *word* after each call. Carefully record the execution trace by writing down the change that each method call makes to the word.
Exercise 5: Age and Shake (Logical Thinking)

Here are two logic puzzles. Try solving them now. If you don’t finish, you can keep working on them during the week if you want to. We’ll talk about the answers next time.

1. What are the ages?
   Two mathematicians are sitting together in a building. They don’t have anything to do at the moment, so one says to the other, “Try guessing the ages of my three children. I’ll give you a hint - the product of their ages is 72.” The other mathematician says, “That’s not enough information. Tell me more.” The first mathematician says, “Their ages add to be the number of this building.” The other mathematician goes outside, looks at the building number, comes back, and says, “That still isn’t enough information. Tell me more.” The mathematician says, “My youngest child’s name is Anne.”

   **Problem:** What are the ages of the children?

2. How many handshakes?
   A woman and her partner attend a party with four other couples. Some people shake hands with other people. Of course, no one shakes their own hand or the hand of the person they came with. When the woman asks the other (9) people present how many different people’s hands they shook they all gave a different answer.

   **Problem:** How many different people’s hands did the woman’s partner shake?
Tracing Exercise Part (b)

word.moveToFront(7)  
word.moveToEnd(5)  
word.swap(2,5)  
word.reverse(3,6)  
word.swap(7,4)  
word.moveToEnd(5)  
word.swap(8,9)  
word.moveToFront(5)  
word.moveToFront(7)  
word.moveToEnd(5)  
word.swap(2,5)  
word.reverse(3,6)  
word.swap(7,4)  
word.moveToEnd(5)  
word.swap(8,9)  
word.moveToFront(5)
Tracing Exercise Part (c): 2 copies of this page per group

Starting value of `word: dormitory_`

<table>
<thead>
<tr>
<th>Method Call</th>
<th>Value of <code>word</code> after the call</th>
</tr>
</thead>
<tbody>
<tr>
<td>word.moveToFront(4)</td>
<td></td>
</tr>
<tr>
<td>word.moveToEnd(2)</td>
<td></td>
</tr>
<tr>
<td>word.moveToFront(2)</td>
<td></td>
</tr>
<tr>
<td>word.reverse(0, 2)</td>
<td></td>
</tr>
<tr>
<td>word.moveToEnd(5)</td>
<td></td>
</tr>
<tr>
<td>word.swap(5, 7)</td>
<td></td>
</tr>
<tr>
<td>word.reverse(5, 6)</td>
<td></td>
</tr>
<tr>
<td>word.moveToEnd(3)</td>
<td></td>
</tr>
</tbody>
</table>

Starting value of `word: dormitory_`

<table>
<thead>
<tr>
<th>Method Call</th>
<th>Value of <code>word</code> after the call</th>
</tr>
</thead>
<tbody>
<tr>
<td>word.moveToFront(4)</td>
<td></td>
</tr>
<tr>
<td>word.moveToEnd(2)</td>
<td></td>
</tr>
<tr>
<td>word.moveToFront(2)</td>
<td></td>
</tr>
<tr>
<td>word.reverse(0, 2)</td>
<td></td>
</tr>
<tr>
<td>word.moveToEnd(5)</td>
<td></td>
</tr>
<tr>
<td>word.swap(5, 7)</td>
<td></td>
</tr>
<tr>
<td>word.reverse(5, 6)</td>
<td></td>
</tr>
<tr>
<td>word.moveToEnd(3)</td>
<td></td>
</tr>
</tbody>
</table>
Baking Exercise (Recipes for students who act out brownies, cookies, fudge)

CHOCOLATE BROWNIES

Ingredients

- 8 ounces semisweet chocolate
- 1 stick (4 ounces) unsalted butter
- 3/4 cup granulated sugar
- 3 large eggs
- 1/4 teaspoon salt
- 1 1/2 teaspoons vanilla
- 1 cup all-purpose flour

CHOCOLATE FUDGE

Ingredients

- 2 3/4 cups sugar
- 4 ounces unsweetened chocolate
- 3 tablespoons butter
- 1 cup half-and-half
- 1 tablespoon corn syrup
- 1 tablespoon vanilla extract
- 1 cup chopped nuts

CHOCOLATE COOKIES

Ingredients

- 2 cups all-purpose flour
- 1/3 cup unsweetened cocoa powder
- 1/2 teaspoon baking powder
- 1/2 teaspoon salt
- 2 sticks (1 cup) unsalted butter
- 1/2 cup granulated sugar
- 1 large egg
- 1 teaspoon vanilla
- 5 oz semisweet chocolate, finely chopped
- 1/2 cup nuts, finely chopped
Green cards for bool cond game

The Earth is flat

Madison is the capital of Wisconsin

* has higher precedence than +

+ has higher precedence than –

The moon is made of green cheese

17 is a prime number

The dorms get Cartoon Network

Red and blue make purple

A chestnut tree makes acorns

A giraffe has a black tongue
Ducks can’t fly

We are learning C++ in CS302

The sun moves around the Earth

The UW-Madison was founded in 1492

There are 100 WES-CS groups

This room is CS 1240

Miss Muffet is afraid of spiders

The time now is 3:33pm

The sun sets in the west

Java programs start executing in a class named Main
The following code will compile: boolean b = "true";

Bascom Hall is exactly five miles from the state capitol building.

System.out.println() reads a value typed on the console.

Booleans can have three possible values.

Boolean algebra was invented by Bobby Boolean.

Maine is the only state whose name is just one syllable.

It is snowing outside.

Wisconsin became a state in 1991.

The mascot of the University of Wisconsin is Bucky the Bobcat.

Pascal, Python, and Scheme are all names of programming languages.
Today is Saturday.

Cars were invented before bicycles.

Picasso was a famous piano player.

Einstein was a famous computer scientist.

Rock music became popular in the 1860s.

Mexico is in South America.

Stressed backwards spells desserts.
Yellow cards for bool cond game

&&

&&

&&

&&

&&

&&

&&

&&

&&
Yellow cards for bool cond game

|| || ||
|| || ||
|| || ||
|| || ||
|| || ||

! ! ! !
Yellow cards for bool cond game

!!!
Yellow cards for bool cond game