CS 113 Spring 2023 — CS I — Midterm II - PRACTICE

- You have approximately 80 minutes.
- This is a long exam, with many chances to demonstrate understanding. We encourage you to try the problems in the order that makes most sense to you, and to keep moving if you get stuck on one question.
- The exam is closed book, closed notes.
- Mark your answers ON THE EXAM ITSELF. If you are not sure of your answer you may wish to provide a *brief* explanation. All short answer sections can be successfully answered in a few sentences AT MOST.
- Make sure to write your answers clearly and legibly.
- Write your initials on the top right hand corner of each page.
- You can use scratch paper but all solutions must be written in the space provided. Scratch paper might not be graded.

First name	
Last name	
BMC/HC Email (What you use for gradescope)	

First and last name of student to your left	
First and last name of student to your right	

I agree to complete this exam without unauthorized assistance from any person, materials or device.

Signature:

For staff use only:

Total	/72

Q1. [7 pts] Multiple Choice

(a) [1 pt] As a Java program runs, this process finds stranded objects and deletes them:

- ◯ Garbage sweeper
- ◯ Garbage collection
- ◯ Waste management
- ◯ Garbage disposal

(b) [1 pt] What is the purpose of the keyword "new"?

- ◯ To declare a new variable
- ◯ To create a new object
- ◯ To call a static method
- ◯ To assign a value to a variable

(c) [1 pt] An interface can be implemented by multiple classe

- ◯ True
- ◯ False

(d) [1 pt] A while loop explicitly increment (or decreases) a counter

- ◯ True
- ◯ False

(e) [1 pt] A for loop explicitly increment (or decreases) a counter

- ◯ True
- ◯ False

(f) [1 pt] A subclass is a class that inherits from a parent class:

- ◯ True
- ◯ False

(g) [1 pt] The abstract keyword is used to define a method or class that has no implementation

- ◯ True
- ◯ False

Q2. [19 pts] Short Answers

(a) [2 pts] In one sentence, whats the difference between `arguments` and `parameters`.

(b) [4 pts] In one sentence, what is the difference between an **empty constructor** and a **value constructor**? Please give an example of each.

(c) [5 pts] What is the main difference between **getters** and **setters**? Why are they needed? Please give an example of each.

.

(d) [4 pts] Imagine iterating through an array of objects to search for an object that has an instance variable that has been assigned a specific variable? What errors might we run into if we are not careful in our implementation? Please provide two errors. Explain why we might encounter them and what the solution might be in our implementation.

(e) [4 pts] What must an object have that allows us to print out the object to the console? What happens if we try printing the object but the object does not have that?

.

Q3. [16 pts] Reading Code

In the following problems, you are going to be shown some code. Write out what the output will be. If the code throws an error, indicate what error is thrown, but still make sure to write what gets printed out before the error occurs.

(a) [2 pts]

```
int count = 0;
for (int i = 1; i <= 5; i++) {
    count += i;
}
System.out.println(count);
```

(b) [2 pts]

```
int[] numbers = {1, 2, 3, 4, 5};
int sum = 0;
for (int i = 2; i <= numbers.length; i++) {
    sum += numbers[i];
    System.out.println(sum);
}
```

(c) [2 pts]

```
int result;
for (int i = 10; i >= 1; i--) {
    if (i % 2 == 0) {
        result = i;
    }
}
System.out.println(result)
```

Consider the class `Fibonacci` implemented below

```
public class Fibonacci {
    public static void main(String[] args) {
        int input = Integer.parseInt(args[0]);
        int result = fibonacci(input);
        System.out.println("Fibonacci sequence at index " + input + " is " + result);
    }

    public static int fibonacci(int n) {
        (g) draw function stack here when n == 5 and input in main is 9
        if (n == 0) {
            return 0;
        } else if (n == 1) {
            return 1;
        } else {
            return fibonacci(n - 1) + fibonacci(n - 2);
        }
    }
}
```

(d) [2 pts] What happens if we compile this program and run '*java Fibonacci*'? What will get printed out?

(e) [2 pts] What happens if we compile this program and run '*java Fibonacci ten*'? What will get printed out?

(f) [2 pts] How can we use the method `fibonacci()` in another class?

```
public class Fibonacci {
    public static void main(String[] args) {
        int input = Integer.parseInt(args[0]);
        int result = fibonacci(input);
        System.out.println("Fibonacci sequence at index " + input + " is " + result);
    }

    public static int fibonacci(int n) {
        (g) draw function stack here when n == 5 and input in main is 9
        if (n == 0) {
            return 0;
        } else if (n == 1) {
            return 1;
        } else {
            return fibonacci(n - 1) + fibonacci(n - 2);
        }
    }
}
```

(g) [4 pts] Again consider the class `Fraction` implemented previously

Draw the function stack at point (g) in the program. Specifically, this would be when `n == 5` and `input` in main is 9.

Q4. [30 pts] Programming

Complete the following programming questions. Unless specified, you do not need a main method.

(a) [10 pts] Write a method called `canSpell()`. Given two strings (a list of tiles/letters and a target word), determine if the target word can be spelled based on the letters. You might have more than one tile with the same letter. You can assume that each both strings will not be empty.

Examples:

- `canSpell("abogcdeffygodd", "doggy")` $->$ `true`
- `canSpell("loppela", "apple")` $->$ `true`
- `canSpell("loppela", "apples")` $->$ `false`
- `canSpell("asdofreds", "dog")` $->$ `false`

(b) [10 pts] Design and implement two classes: one named *Time*, and one named *Date*. The `Time` class should have an `hour`, `minute`, and `second`. The `Date` class should have a `day`,`month`, and `year`.

For a given `Time` or `Date`, we would like to know how far apart in years, days, and seconds two `Time` objects are from each other or two `Date` objects are from each other.

Make sure that the `Date` object has all the instance variables and methods that a `Time` object has as well. Lastly, make sure that in your implementation, `Time` and `Date` objects are **immutable**.

Make sure to include the necessary instance methods for every class.

More space for the previous question

(c) [10 pts] Write a method `argMinShortest()` that takes in an array of strings and determines the location of the shortest string. If there are multiple strings that all are the shortest in length, then return the location of the last string with the shortest length.

You are encouraged to write helper methods if necessary.

Examples:

- `argMinShortest({"aaa", "a", "asdds", "edfef"})->[1]`
- `argMinShortest({"dsdsd", "sfss", "as", "edfef", "ds"})->[4]`