CS 113
Fall 2023

CS I

Final

- You have approximately 180 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)	

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

Signature:

For staff use only:

Total	/129

Q1. [20 pts] Multiple Choice

(a) [1 pt] Computer Science is about: (circle the best answer):

- ◯ Computers
- ◯ Algorithms
- ◯ Science

(b) [1 pt] The following is a type of method in Java: (circle all that apply):

- ◯ abstract
- ◯ concrete
- ◯ object
- ◯ static
- ◯ stub

(c) [1 pt] In Java, all classes extend the `Object` class:

- ◯ True
- ◯ False

(d) [1 pt] A $O(n)$ algorithm will **always** be faster than a $O(n^2)$ algorithm:

- ◯ True
- ◯ False

(e) [1 pt] The compilers job is to run a java program:

- ◯ True
- ◯ False

(f) [1 pt] Big-O notation indicates how many steps an algorithm performs in the **best case** scenario:

- ◯ True
- ◯ False

(g) [1 pt] Binary search is a search algorithm that has a time complexity of $O(logn)$

- ◯ True
- ◯ False

(h) [1 pt] The following are valid access modifiers (circle all that apply):

- ◯ private
- ◯ nonpublic
- ◯ protected
- ◯ public

(i) [1 pt] The following are valid keywords for creating an instance of class (circle all that apply):

- ◯ object
- ◯ new
- ◯ create
- ◯ class

(j) [1 pt] A method signature includes the name of the method, the parameters it accept, the return type, and the body:

- ◯ True
- ◯ False

(k) [1 pt] A class is a blueprint for creating objects that define the attributes and methods of those objects:

- ◯ True
- ◯ False

(l) [1 pt] A variable is a set of instructions that perform a specific task in a program:

- ◯ True
- ◯ False

(m) [1 pt] Mutable objects cannot change after they are created:

- ◯ True
- ◯ False

(n) [1 pt] A null pointer error is a type of Exception:

- ◯ True
- ◯ False

(o) [1 pt] A class that implements an interface can implement **just some** of the methods in the interface:

- ◯ True
- ◯ False

(p) [1 pt] The following special keywords can be used to access a parent class (circle all that apply):

- ◯ this
- ◯ super
- ◯ new
- ◯ extends

(q) [1 pt] Every element in a new array of booleans is initialized to true.

- ◯ True
- ◯ False

(r) [1 pt] N-3 is the index of the third-to-last element in an array of size N.

- ◯ True
- ◯ False

(s) [1 pt] A class can define at most one instance method.

- ◯ True
- ◯ False

(t) [1 pt] A class can define at most one constructor

- ◯ True
- ◯ False

Q2. [22 pts] Short Answers

(a) Searching

(i) [2 pts] Imagine we are searching for the number 453 in an array of integers. Below, show an array where linear search would be faster than binary search. The array that you draw must support binary search and contain at least 4 numbers.

(ii) [2 pts] Imagine we are searching for the number 453 in an array of integers. Below, show an array where linear search would be slower than binary search. The array that you draw must support binary search and contain at least 4 numbers.

(iii) [2 pts] Imagine we are searching for the number 453 in an array of integers. Below, show an array where linear search and binary search would perform the same number of comparisons and contain at least 4 numbers.

(b) [2 pts] When reading in the contents of a file, is it better to use a **while** or a **for** loop? Why? (make sure to justify your answer).

(c) Sorting For the following questions, image we are trying to sort the following array of integers in ascending order:

$$33, 3, 333, 4, 44, 53, 2, -5$$

(i) [4 pts] What will the array look like after 3 swaps when running **Selection Sort**.

(ii) [4 pts] What will the array look like after 3 swaps when running **Bubble Sort**.

(d) [2 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?

(e) [2 pts] What happens underneath the hood if we use the `==` operator to compare objects. What should every class/object have to solve deal with this?

.

(f) [2 pts] When an object is created, what determines if the **value** or **empty** constructor runs.

Q3. [10 pts] Reading Code

(a) Answer the queestions below based on this code:

```
public static void main(String[] args) {
  System.out.println(prod(1, 4));
}

public static int prod(int m, int n) {
  if (m == n) {
    return n;
  }
  else {
    int recurse = prod(m, n - 1);
    int result = n * recurse;
    return result;
 }
}
```

(i) [4 pts] Draw a stack diagram showing the state of the program just before the last invocation of prod completes.

(ii) [2 pts] What is the output of this program?

(iii) [2 pts] Explain in a few words what prod does (without getting into the details of how it works).

(iv) [2 pts] Rewrite `prod()` below without the temporary variables recurse and result. Hint: You need only one line for the else branch

Q4. [12 pts] Run Time Analysis

For the following questions, 1) write the Big-Oh run time (based on n) and 2) provide a one sentence justification based on the corresponding code snippet.

(a) [2 pts]

```
for (int i = 0; i < n; i++) {
  for (int j = 0; j < n/2; j= j+2) {
    System.out.println(j * i);
  }
}
```

(b) [2 pts]

```
i = n;
while (i < n+1) {
  for (int x = -1; x > (-1 * n); x--;){
    System.out.println(x);
  }
  i++;
}
```

(c) [2 pts]

```
i = 1;
while (i < n+1) {
  for (int x = -1; x > (-1 * n); x--;){
    System.out.println(x);
  }
  i++;
}
```

(d) [2 pts]

```
i = 0;
while (i < n+1) {
  for (int x = -1; x > (-1 * n); x--;){
    System.out.println(x);
  }
  i = i * 2;
}
```

(e) [2 pts]

```
for (int a = 0; a < n; a++) {
  for (int b = 0; b < n; b++) {
    for (int c = 0; c < n; c++) {
      for (int d = 0; d < 10; d++) {
        System.out.println(a+b+c+d);
      }
    }
  }
}
```

(f) [2 pts]

```
for (int a = 0; a < 10; a++) {
  for (int b = 0; b < 100; b++) {
    for (int c = 0; c < 1000; c++) {
      for (int d = 0; d < 10000; d++) {
        System.out.println(a+b+c+d);
      }
    }
  }
}
```

Q5. [65 pts] Programming

Complete the following questions:

(a) [10 pts] Write a method called `numOccurs()`. Given two strings, determine how many times the shorter string appears in the larger string. You can assume that both strings will not be empty. You can also assume that the strings will not be the same length. For full points, your solution should be recursive.

Examples:

- `numOccurs("apple", "pl")` $->$ 1
- `numOccurs("mississippi", "ss")` $->$ 2
- `numOccurs("hello world", "o")` $->$ 2
- `numOccurs("ab", "abababab")` $->$ 4
- `numOccurs("uvwxy","abcdefghijklmnopqrstuvwxyz")` $->$ 1

More space for the previous question

(b) [20 pts] Design and implement a Java class that represents a vending machine that dispenses various items based on user input.

The vending machine should have at most 3 types of snacks available for purchase, each with its own price. The class should have a method that supports a user inputting the amount of money they wish to spend, select a product to purchase (based on its name), and receive the appropriate change. If the amount of money a user puts in is not enough, the user should see a message and the money they put in should be returned.

The vending machine class should also keep track of the inventory for each snack and prevent the user from selecting a snack that is out of stock.

For full points, your implementation should include the following features using Object Oriented Programming principles:

A Snack class that contains information about each product, including its name, price, and inventory. A VendingMachine class that represents the vending machine and contains a method where a user can select a product and insert money. The class should also have a toString() method that indicates the different types of snacks available, including their name, price, and how many are available in the vending machine.

Error handling for cases where the user inputs an invalid amount of money or selects a product that is out of stock.

Make sure to include necessary instance methods (including getters and setters) for every class.

The classes you implement should support the following driver:

```
public static void main(String[] args) {
  VendingMachine machine = new VendingMachine();
  Snack cheetos = new Snack("cheetos", 1.25);
  Snack bamba = new Snack("bamba", 0.75);
  Snack cheezits = new Snack("cheezits", 1.50);
  Snack oreos = new Snack("oreos", 5.25);

  machine.add(cheetos);
  machine.add(cheetos);
  machine.add(cheetos);
  machine.add(bamba);
  boolean addedCheezits = machine.add(cheezits);
  boolean addedOroes = machine.add(oreos);
  System.out.println(addedCheezits); // prints out true
  System.out.println(addedOroes); // prints out false

  System.out.println(machine.buy("bamba", 1.25)); // prints 0.50
  machine.buy("bamba", 1.25); // returns 1.25 and prints a message indicating there are no more bamba

  machine.buy("oreos", 6); // returns 6 and prints a message saying there are no oreos.

  machine.getPrice("cheezits") // returns 1.50
  machine.getCount("cheetos") // returns 3
}
```

More space for the previous question

More space for the previous question

(c) [15 pts] Write a method called `getSingularValues()` that returns the singular values from a singular value matrix of doubles. If the input is not a singular value matrix, then return an empty array.

A matrix is considered a singular value matrix if all the following conditions are met:

- The matrix is square, i.e. it has the same number of columns as number of rows
- Every value is 0 except for the values along the diagnol are greater than 0.
- Along the diagnol, the first value is strictly greater than the second, the second is strictly greater than the third, etc.

Note: The diagnol of a matrix are the values starting at the first element in the first row, the second element in the second row, the third element in the third row, . . .

Below are examples of matrices that are (Table 1) and matrices that are not (Table 2, Table 3, Table 4) singular value matrices. For Table 1, the method should return {`5.4, 2.2, 1.2, 0.2`}. For the other tables, the method should return an empty array.

5.4	0.	0.	0.
0.	2.2	0.	0.
0.	0.	1.2	0.
0.	0.	0.	0.2

Table 1: Valid singular value matrix

5.4	0.	0.	0.
0.	2.2	0.	0.
0.	0.	1.2	0.
0.	0.	0.	5.2

Table 2: Invalid singular value matrix

5.4	0.	0.	0.
0.	2.2	0.	0.
0.	0.	1.2	0.
0.	0.	0.	0.2
0.	0.	0.	0

Table 3: Invalid singular value matrix

5.4	0.	0.	0.
0.	2.2	0.	1.
0.	0.	1.2	0.
9.9	0.	0.	0.2

Table 4: Invalid singular value matrix

More space for the previous question

(d) [10 pts] Write a method called ‘mode()‘ that determines which integer appears the most amount of times in an array of positive integers. You can assume that there is just one mode in the array.

To receive full credits, your solution's runtime must be $O(n)$. Partial credit will be given for less efficient solutions.

- `mode({1,1,4,4,5,5,6,6,6})` $->$ `6`
- `mode({100,10,100})` $->$ `100`
- `mode({10,9,8,8,7,6,5,4,3,2,1})` $->$ `8`

More space for the previous question

(e) [10 pts]

Write a method called ‘found()‘ that determines whether a 2-D array contains a specific string. Note: it is possible that the arrays in each 2-D array have different lengths. You can assume that both arguments, the 2-D array and the string, will not be empty.

- `found({{"a", "b"}, {"c","d","e"}},"hello")` $->$ `false`
- `found({{"a", "b"}, {"c","d","e"}, {"hello"}},"hello")` $->$ `true`
- `found({{"a", "b"}, {"c","d","e"}},"a")` $->$ `true`

More space for the previous question