



University of Maryland College Park

Department of Computer Science

CMSC131 Spring 2024

Exam #1

FIRSTNAME, LASTNAME (PRINT IN UPPERCASE):

STUDENT ID (e.g. 123456789):

Instructions

- Please print your answers and use a pencil.
- Do not remove the staple from the exam. Removing it will interfere with the Gradescope scanning process.
- To make sure Gradescope can recognize your exam, print your name, write your directory id at the bottom of pages with the text DirectoryId, **provide answers in the rectangular areas provided**, and do not remove any exam pages. Even if you use the provided extra pages for scratch work, they must be returned with the rest of the exam.
- This exam is a closed-book, closed-notes exam, with a duration of 50 minutes and 100 total points.
- Your code must be efficient.
- Multiple choice questions have only one answer unless indicated otherwise.
- You don't need to use meaningful variable names; however, we expect good indentation.

Grader Use Only

#1	Problem #1 (Short Answers – 2 pts each)	16
#2	Problem #2 (Code 0)	14
#3	Problem #3 (Code 1)	18
#4	Problem #4 (Code 2)	22
#5	Problem #4 (Code 3)	30
Total	Total	100

Problem #1 (Short Answers – 2 pts)

1. (2 pts) Which statement is false?
 - a. `'A'` is a **String** literal.
 - b. An **int** type is 4 bytes in Java.
 - c. An example of a Java primitive type is **float**.
 - d. `==` can be used to check for equality of 2 integers.
2. (2 pts) Which statement is true?
 - a. Java has 6 primitive types.
 - b. You can have a **return** statement in a **void static** method.
 - c. Addition has higher precedence than multiplication.
 - d. You can have a variable in Java called **2numbers**.
3. (2 pts) One of the 4 variables below might not compile. Which one and why? If they will all compile, write "All will compile"

<code>int \$\$\$;</code>	<code>int _1_;</code>	<code>int a1;</code>	<code>int Int;</code>
--------------------------	-----------------------	----------------------	-----------------------

4. (2 pts) The expression `"chicken".compareTo("duck")` returns:
 - a. A negative integer
 - b. A positive integer
 - c. Zero
 - d. **false**
 - e. **true**
 - f. None of the above.

5. (2 pts) The base 10 number 183 is what in octal (base 8) (no calculators) ?

6. (2 pts) Assume the following code fragment in a **main** method, what is the output?

```
double x = 3/4;  
int num = 10 * ((int)x) ;  
int result = num++;  
result+=result;  
System.out.println(result++);
```

7. (2 pts) Assume the following code fragment in a **main** method, what is the output? _____

```
int myVar = 5;  
if ("hi".equals("Hi"))  
    myVar*=2;  
else if (myVar > 5)  
    myVar++;  
if (myVar !=5)  
    myVar = 30;  
System.out.println(myVar);
```

8. (2 pts) Assume the following code fragment in a **main** method, how many times will **hi** print? Answer needs to be a number.

```
for(int i= 1; i <= 10; i++)    {  
    System.out.println("hi");  
    for(int j=1; j<= 100; j++){  
        System.out.println("hi");  
    }  
}
```

Problem #2 (Code 0)

Write a public static method called **same2** which returns a **boolean**. If at least 2 of the 3 Strings passed in are equal, return **true**. Otherwise return **false**. You can assume non-nulls String will be passed in as arguments. As for library methods, you can only use the String **equals** method. If you have a **System.out.println** in your code, you are not writing the code correctly. Do not change the given **main**.

```
public class Code0 {  
    //Write your method below
```

```
public static void main(String[] args) {  
    System.out.println(same2("java", "if", "else")); //false  
    System.out.println(same2("else", "if", "else")); //true  
    System.out.println(same2("else", "else", "else")); //true  
}
```

```
}
```

Directory id:

Problem #3 (Code 1)

Write a public static method called **print1** which returns a **String** and has one **String** parameter: **str**. The method returns a **String** where each line has a new letter from **str** starting with the last letter first (and adding new letters from right to left). You can assume a non-null **String** will be passed in for **str**. For this method, you can only use **one loop** (no nested loops). As for library methods, you can only use the **String** methods: **length** and the version of the **substring** method that takes one argument. If you have a **System.out.println** in your code, you are not writing the code correctly. No need for a **main**.

Sample calls	Output of the calls
<code>System.out.println(print1("Java"));</code>	<code>a</code> <code>va</code> <code>ava</code> <code>Java</code>
<code>System.out.println(print1("Computer"));</code>	<code>r</code> <code>er</code> <code>ter</code> <code>uter</code> <code>puter</code> <code>mputer</code> <code>omputer</code> <code>Computer</code>

```
public static String print1(String str) {
```

Problem #4 (Code 2)

Write a public static method called **print2** which returns a **String** and has one **String** parameter: **str**. This method does exactly the same thing as **print1** from the previous question. The method returns a String where each line has a new letter from **str** starting with the last letter first (and adding new letters from right to left). You can assume a non-null String will be passed in for **str**. For this method you should use a nested loop. As for library methods, you can only use the String methods: **length** and **charAt**. If you have a **System.out.println** in your code, you are not writing the code correctly. No need for a **main**. Obviously, you cannot just call your **print1** code and you have to re-develop the logic with the new given restrictions.

Sample calls	Output of the calls
<code>System.out.println(print2("Computer"));</code>	<code>r</code> <code>er</code> <code>ter</code> <code>uter</code> <code>puter</code> <code>mputer</code> <code>omputer</code> <code>Computer</code>

```
public static String print2(String str) {
```

Problem #5 (Code 3)

Write a public static method called **makeSquare** which returns a **String** and has one integer parameter (**rows**) and three character parameters (**ch1**, **ch2**, **ch3**). You can assume arguments of the appropriate type will be passed in. If the integer parameter is less than 3, return **null**. Otherwise, return a String that is a square with the number of rows being the parameter **rows**. The perimeter (i.e. border) of the square should be **ch1** and the inner part of the square should alternate between **ch2** and **ch3**, starting with **ch2**. You cannot use any library methods. If you have a **System.out.println** in your code, you are not writing the code correctly. No need for a **main**.

Sample calls	Output of the calls
<code>System.out.println(makeSquare(7, 'A', 'B', 'C'));</code>	<pre>AAAAAAA ABBBBBB ACCCCCA ABBBBBB ACCCCCA ABBBBBB AAAAAAA</pre>

```
public static String makeSquare(int rows, char ch1, char ch2, char ch3) {
```


Extra Page If You Need It

Last Page