

|  |                      |               |                  |
|--|----------------------|---------------|------------------|
|  |                      |               |                  |
| Course Code:    CSC111                               |                      | Final Exam    |                  |
| Course Title:    Java Programming 1                  |                      | 7.May.2020    |                  |
| Semester:        2 <sup>nd</sup> 2019/2020   - 1441H |                      | 1:00 – 4:00pm |                  |
| Answer Sheet   |                      |               |                  |
| Student Name:  |                      |               |                  |
| Student ID:  |                      |               |                  |
| Section No.  |                      |               |                  |
| Student Serial No:                                   |                      | 40            |                  |
|  |                      |               |                  |
| Course Learning Outcomes                             | Question No.         | Points        | Student's Points |
| CLO 1.1 , 1.3  | Q1-12: T/F           | 6             |                  |
| CLO 1.1 , 1.3  | Q13-32: MCQ & Blanks | 10            |                  |
| CLO 1.3  | Q33-40: Tracing      | 8             |                  |
| CLO 1.3  | Q41-44: Coding       | 16            |                  |

**Instructions: (Read Carefully):**

- Write your full name, student ID, and section number in the spaces provided.
- This is an open book exam. Use the mark distribution given by the table above to judge the amount of time you should spend on a question.
- Write **ALL** your answers on this question file/paper **ONLY if you cannot answer on LMS**. Answers written elsewhere will **NOT** be accepted.
- If you write more than one solution, the answer will be considered a **wrong answer**.
- Write your answers clearly, if I can't read it, the answer will be considered a **wrong answer**.
- Read the questions carefully.
- General notes on "filling the blanks"**
  - Give the exact value, or write "error"
  - Pay attention to decimal points, spaces, spelling, and capitalization
- General notes on programming:**
  - Stick to the requirements of the question
  - If variable names are given stick to them
  - Choose the most appropriate statement for your implementation. I.e., choose between if and switch wisely, choose between while, do-while, for loops wisely.
  - Pay attention to whether the code will be in the object's class or in the class with the main method.
  - You may add setters or getters, but ONLY if needed.

*Good luck...*

**True/ False****(6 pts.)**

A. Answer (T) rue or (F) alse for each of the following statements:

44. ( ) A method can be defined inside another method

45. ( ) You will get an error if these statements

```
{ ...
  if(8<9) return 1;
  if(9>8) return 0;
} // end of method
```

are the last two statements in a method that returns an integer.

46. ( ) Methods of a class can access its own private attributes without using a getter or setter method

47. ( ) When you parse an integer value with `Double.parseDouble()` it will cause a runtime error.

48. ( ) using `num%3` is commonly used to test if `num` is even or odd or zero.

49. ( ) Default constructors and explicitly defined constructors cannot return values.

B. Consider the declarations, then evaluate the expressions:

```
char q = 'Q';
char ch = 'Z';
char letter = 'a';
double num = 19.9;
char[] arr1 = { 'A', 'B', 'C' };
char[] arr2 = { 'A', 'B', 'C' };
```

|    | Boolean expression                          | result<br>(true/false) |
|----|---|------------------------|
| 7  | <code>!Character.isLowerCase(q)</code>      |                        |
| 8  | <code>( letter &lt; ch )</code>             |                        |
| 9  | <code>( 21%7 &gt; 1)    ( q == 'q' )</code> |                        |
| 10 | <code>Math.floor(num) == 20.0</code>        |                        |
| 11 | <code>arr1 == arr2</code>                   |                        |
| 12 | <code>arr1[1] == arr2[0]+1</code>           |                        |

**MCQ and Fill in the blanks****(10 pts)**

Select one correct answer **OR** fill in the blank with an **exact value** or the word **error**.

**Note:** Pay attention to decimal points, spaces, spelling, and capitalization when filling the blanks

**A. Strings and arrays:**

```
String[] myArray = {"Good Day!", "Hello!", "Bye!"};  
int i = 1;
```

13. The expression `myArray.toUpperCase()` will do the following:
- a. nothing
  - b. return the first String converted to upper case letters.
  - c. return the whole array converted to uppercase letters
  - d. cause an error
14. The expression `myArray[2].length()` will return: \_\_\_\_\_
15. The expression `myArray.length()` will return: \_\_\_\_\_
16. The statement `myArray[0] = myArray[i++]`; will result the following:
- a. `myArray` now contains "Good Day!" twice
  - b. `myArray` now contains "Hello!" twice
  - c. `myArray` now contains "Bye!" twice
  - d. it will cause an error because you have to use method `equals()` with strings.

**B. Method Overloading:** consider the following statements

**Method#1:** `public int sum (int x) { return x+5; }`

**Method#2:** `public long sum (int m, int n) { return m + 2*n ; }`

**Method#3:** `public double sum (int x, double y) { return x + y ; }`

17. The expression `double result = sum(9.5);` will do the following:
- a. Call Method#1 and perform integer casting for the parameter
  - b. Call Method#3 and set the first parameter to 0
  - c. cause an Error
  - d. we don't have enough information to know what will happen
18. The expression `long result = sum(10);` will make `result =` \_\_\_\_\_
19. The expression `double result = sum(12,1);` will make `result =` \_\_\_\_\_
20. Consider adding this method to the previous 3 methods
- Method#4:** `public double sum (int x, int y) { return x + y + 0.5; }`
- a. we can't overload it, because Method#3 also returns a double
  - b. we can't overload it, because Method#3 also has parameter `x` and `y`
  - c. we can't overload it, because Method#2 also takes 2 integer parameters
  - d. overloading is ok, because the signature is different from the others

**C. General**

21. When a method is invoked in a stand-alone statement, it is most likely of type...

- a. String
- b. void
- c. overloaded
- d. static

22. When an attribute of a class is declared as `static`...

- a. it means it is fixed and can't be changed outside the class
- b. there is only one copy of it and it is shared by all instances of that class
- c. it can only be used by static methods
- d. all of the above

23. The body of this `<< while(false){...} >>` loop is executed...

- a. once
- b. never
- c. infinitely
- d. None of the above

24. Which of the following is true about this code?

```
int[] X = {1, 3, 5, 7};
int[] Y = X;
```

- a. Y has the same elements as X
- b. X and Y point to the same memory location.
- c. Both A and B.
- d. None of the above.

**D. Consider the declarations, then evaluate each expression independently:**

```
String str = "CSC111#JAVA#Final";
```

```
String word = "Start";
```

```
int two = 2;
```

|    | expression   | result |
|----|--|--------|
| 25 | <code>str.substring(6,11).replace("#",word)</code>                             |        |
| 26 | <code>str.charAt(str.length()/two)</code>                                      |        |
| 27 | <code>str.substring(0,1).toUpperCase() + str.substring(1).toLowerCase()</code> |        |
| 28 | <code>- Math.abs( -9.0 + Math.sqrt(36) )</code>                                |        |
| 29 | <code>17.0 + Math.floor(19.3) % two</code>                                     |        |
| 30 | <code>Math.pow(Math.ceil(8),two)</code>  |        |
| 31 | <code>Character.isDigit(char(two))</code>                                      |        |
| 32 | <code>Character.isDigit(str.substring(two+two,5))</code>                       |        |

**(8 pts)**

**33. What is the output?**

[illegible]

34. Consider the code skeleton where ( ... ) can be any valid statements, and answer the questions that follow [34-40]. Assume the values of the variables are only changed in the statements that are shown.

```
1#  public class EXAM {
2#  private char letter = 'o';
3#  static String[] list = new String[5];
4#  public static int y = 0;
5#
6#  public static void main(String[] args){
7#      SOME objA = new SOME();           // valid?
8#      SOME objB = new SOME('B');
9#      ...                               // value of y?
10#     System.out.println(objB.letter);   // valid?
11#     addStudent("Heidi");
12#     int y = 55;
13#     System.out.println(last);           // valid?
14#     ...                               // value of y?
15# }// end main
16#
17# static int numOfStudents = 0;
18#
19# public static void addStudent(String name ){
20#     ...
21#     y += 10;                           // valid?
22#     list[numOfStudents] = name;         // any consideration?
23#     numOfStudents ++;
24#     System.out.println(letter);         // valid?
25#     ...
26# }// end addStudent
27#
28# public void printAll (String[] arr){
29#     ...
30#     System.out.print(letter);           // valid?
31#     System.out.print(arr.length);       // valid?
32#     System.out.print(list.length);      // valid?
33#     System.out.print(last);             // valid?
34#     ...
35# }// end printAll
36#
37# double last = 999.999;
38# }// end of class EXAM
39#
40# public class SOME {
41#     public char letter;
42#     public SOME (char x)
43#     {   letter = x;   }
44#     public void setLetter (char x)
45#     {   letter = x;   }
46#     ...
47# }// end of class SOME
```

Which method can access variable `numOfStudents`? **(check all that apply)**

- ☐ `main`
- ☐ `addStudent`
- ☐ `printAll`
- ☐ `none`

35. Which method can access variable `name`? **(check all that apply)**

- ☐ `main`
- ☐ `addStudent`
- ☐ `printAll`
- ☐ `none`

36. Can we call method `printAll()` inside method `main()`? And why?

37. Can we call method `addStudent()` inside method `printAll()`? And why?

38. Which line is valid? **(check all that apply)**

- ☐ line #7
- ☐ line #10
- ☐ line #13
- ☐ line #21
- ☐ line #24
- ☐ line #30
- ☐ line #31
- ☐ line #32
- ☐ line#33

39. What is the value of `y` on each of the following lines? Give a value or write “error”

- ☐ line #9      \_\_\_\_\_
- ☐ line #14     \_\_\_\_\_

40. Assuming the code on line #22 is valid, is it possible it will still give an error? Explain briefly.

**Write Java statements/Methods:****(16pts)**

Consider the UML:

| Medicine   |
|--|
| - name: String<br>+ price: double<br>- qty: int<br>- numSold: int<br>+ form: char  |
| + setMed(String n, double p, char f): void<br>+ buyMed(int num): double<br>+ displayMed():void<br>+ resetNumSold():void<br>(add setters or getters ONLY if needed) |

| Short Description  |
|--|
| <ul style="list-style-type: none"> <li>the name of the medicine</li> <li>the price of the medicine in the pharmacy</li> <li>the quantity available, initially it is 1500</li> <li>to count how many were sold <b>today</b></li> <li>the form: 'o' for ointment, 'p' for pill, 'l' for liquid, 's' for spray</li> </ul> |
| <ul style="list-style-type: none"> <li>see below</li> <li>see below</li> <li>see below</li> <li>see below</li> </ul>   |

| Aspirin                                     |
|---|
| name: "Aspirin"<br>price: 5.95<br>form: 'p' |

| Ventolin                                      |
|---|
| name: "Ventolin"<br>price: 12.25<br>form: 's' |

**Method Description:**

- setMed**

Sets name, price, and form, according to the parameters. It also initializes qty to 1500 and numSold to 0.

- buyMed**

Allows a customer to buy this medicine and returns the total price (if sold). Parameter num refers to the quantity the customer wants to buy of this medicine.

- displayMed**

Displays the medicine's information formatted as in the example. Note: the numbers below help you count the columns; they are **not** part of the output.

Assume 55 Aspirins and 200 Ventolins have already been sold; the result of

```
Aspirin.display(); Ventolin.display();
```

should look like this:

```

  Aspirin as pill      @      5.95 SR sold 55 remaining 1445
  Ventolin as spray    @      12.25 SR sold 200 remaining 1300
1234567890    12345789    12345678
```

- resetNumSold**

Resets the number of medicines sold today (numSold) to 0



**Assume you have this class with the main method**

```
import...
public class Pharmacy
{
    static Scanner input = new Scanner (System.in);
    public static void main (String[] args)
    {
        Medicine Aspirin = new Medicine();
        Medicine Ventolin = new Medicine();
        Aspirin.setMed(.....);
        Ventolin.setMed(.....);

        Medicine[] meds = new Medicine[50];    // inventory of the pharmacy
        int numMeds = 0;                        // number of different medicines

        for(int i=0; i<10 ; i++)
        { med[numMeds++] = new Medicine();
          // ... read some input
          // ... set the medicine
        }
        /* *** more code goes here *** */
    } // end main

    /* *** additional methods go here *** */

} // end class Pharmacy
```

41. (15 Min) write the implementation of `displayMed()` for class `Medicine`

```
public void displayMed()
{

}

}
```

42. (10 Min) Write some code in the `main` method to discount the price of `Ventolin` to the half if it has been sold less than 50 times today, otherwise display how often it was sold. Assume `Ventolin` is a `Medicine` object and is created and initialized.

43. (15 Min) In class `Pharmacy`, write a **method** `getAllOintments` that will receive an array of medicines (`arr`) and the number of medicines stored in it (`numMeds`) and return an array of medicines that are in the form of ointment. (it is ok if the returned array is larger than needed, but choose an appropriate size)  
NOTE: You are given the partial header of the method, repeat writing the complete header in the text box with your implementation

```
public _____ getAllOintments(Medicine[] arr, int numMeds)
{

}

}
```

44. (5 Min) Write some statements in the `main` method to display the first 5 ointment medicines in the pharmacy. (assume there are for sure 5 or more ointment medicines)