

Software Construction & Development (SE3001)

Sessional-I Exam

Date: September 23, 2024

Course Instructor(s)

Dr. Farooq Ahmed, Mr. Waqas Ali

Total Time: 1 hour

Total Marks: 30

Total Questions: 3

22i-2505

Roll No

SE-5A

Section

[Signature]

Student Signature

Do not write below this line

Attempt all questions on the answer sheet

CLO 1: Apply software engineering concepts to construct (i.e. design, develop, and test) software in team setting

Question 1

[10 marks]

Implement a generic class **FrequencyCounter** to support calculation of frequency of elements occurring in an array. Following operations need to be supported:

1. **countFrequency**: Takes data and a specific value as an argument and returns the frequency (count of occurrences) for the given value. For instance, in an array {2, 1, 3, 4, 2, 3, 2}, frequency of 2 is 3.
2. **mostFrequentValue**: Takes data as argument and returns the value with the highest frequency. For instance, in an array {'b', 'a', 'c', 'd', 'b', 'c', 'b'}, most frequent value is 'b'.

Question 2

[10 marks]

You are provided with a file named **cart.txt**, where each line contains a user ID followed by a list of product IDs separated by commas. The format of each line is as follows:

User_id, product1_id, product2_id, product3_id, product4_id, ...

Each line represents a unique user and the products they wish to purchase. Write a Java program that performs the following task:

- **Display Stats**: Display the user ids in descending order depending upon the number of products they are purchasing.

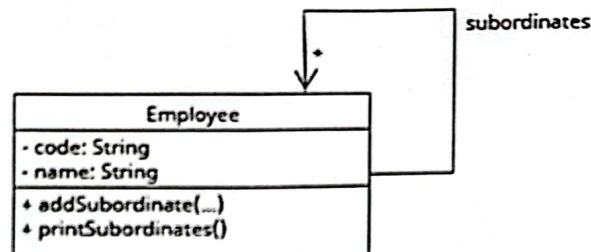
Make proper use of file and exception handling.

CLO 2: Implement software design patterns as a part of software construction activity

Question 3


[10 marks]

Reflexive Association is a common occurrence in UML class diagrams. Consider the following UML diagram showing an Employee – Subordinate relationship using reflexive association.



Implement the above diagram using Java code.

National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	Software Construction & Development	Course Code:	CS-3001
	Degree Program:	BS(SE)	Semester:	Fall 2023
	Exam Duration:	60 Minutes	Total Marks:	45
	Paper Date:	2 – Oct - 2023	Weight:	15.00%
	Section:	ALL	Page(s):	6
	Exam Type:	Midterm-I		

Student Name: _____ **Roll No.** _____ **Section:** _____

Instruction/Notes: Attempt all questions. Do not use pencil or red ink. In case of confusion or ambiguity make a reasonable assumption. **Do not attach any extra sheet.** Use extra sheet for rough work only

Question 1 [CLO-1]

10+10=20 points

(a) Find the exception handling related errors in the following code considering best practices and suggest correction:

<pre> class InvalidAgeException extends Exception { public InvalidAgeException(String message) { super(message); System.out.println("User defined exception"); } } public class CustomExceptionDemo { public static void main(String[] args) { FileReader reader = new FileReader("file.txt"); char[] buffer = new char[1024]; reader.read(buffer); reader.close(); try { validateAge(15); } catch (Exception e) { System.out.println("Caught an Exception: " + e.getMessage()); } catch (InvalidAgeException e) { System.out.println("Caught an InvalidAgeException: " + e.getMessage()); } } public static void validateAge(int age) throws ArithmeticException { validateAge2(age); } public static void validateAge2(int age) throws InvalidAgeException { if (age < 18) { throw new Exception("Age must be 18 or older."); } System.out.println("Valid age: " + age); } } </pre>	<p>Errors/Correction:</p>
---	----------------------------------

(b) Given the code snippets below, complete the **main** to: (1) first get a list of all the discounted products in inventory and (2) print the list showing product id and discount rate. You cannot change any of the classes or the code already

provided but assume that setters and getters are already available. Also note that there is no specific order in which Discounted Products can be added to Inventory.

```
class Product {
    protected int id;
    protected String name;
    protected double price;
    public Product(int id, String name, double price) {
        // set appropriate data members ...
    }

    // setters and getters go here ...
}

class DiscountedProduct extends Product{
    protected float discount;

    public DiscountedProduct(int id, String name, double price, float discount) {
        super(id,name,price);
        this.discount = discount;
    }

    // setters and getters go here ...
}

class Inventory<T extends Product> {
    private Map<Integer, T> products = new HashMap<>();
    public void addProduct(T product) {
        products.put(product.getId(), product);
    }
    public List<T> getAllProducts() {
        return new ArrayList<T>(products.values());
    }
}

public class Main {
    public static void main(String[] args) {
        Inventory<Product> inventory = new Inventory<>();
        inventory.addProduct(new Product(1, "Laptop", 899.99));
        inventory.addProduct(new DiscountedProduct(2, "Headphones", 79.99,0.075));
        inventory.addProduct(new DiscountedProduct(3, "Charger", 99.99,0.05));
        inventory.addProduct(new Product(4, "Smartphone", 499.99));

        // other products and discounted products may be added here ...

        // complete code to get a list of all discounted products from inventory

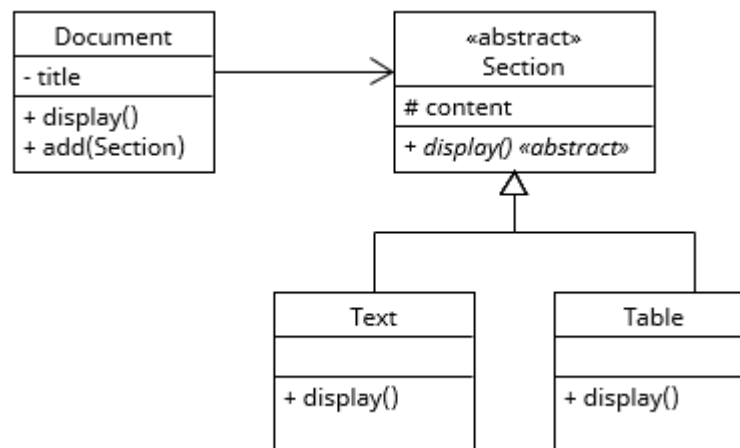
        // write code to show discount rate against each discounted product

    }
}
```

Question 2 [CLO-4]**5+5=10 points**

(a) A developer committed some unnecessary changes to an existing code file (Main.java) in the Git repository by mistake. (S)he wants to undo those changes and move the repository back to its previous state. What steps are required to accomplish this goal? Describe the strategy in general.

(b) What are the advantages of using Git?

Question 3 [CLO-1]**15 points**

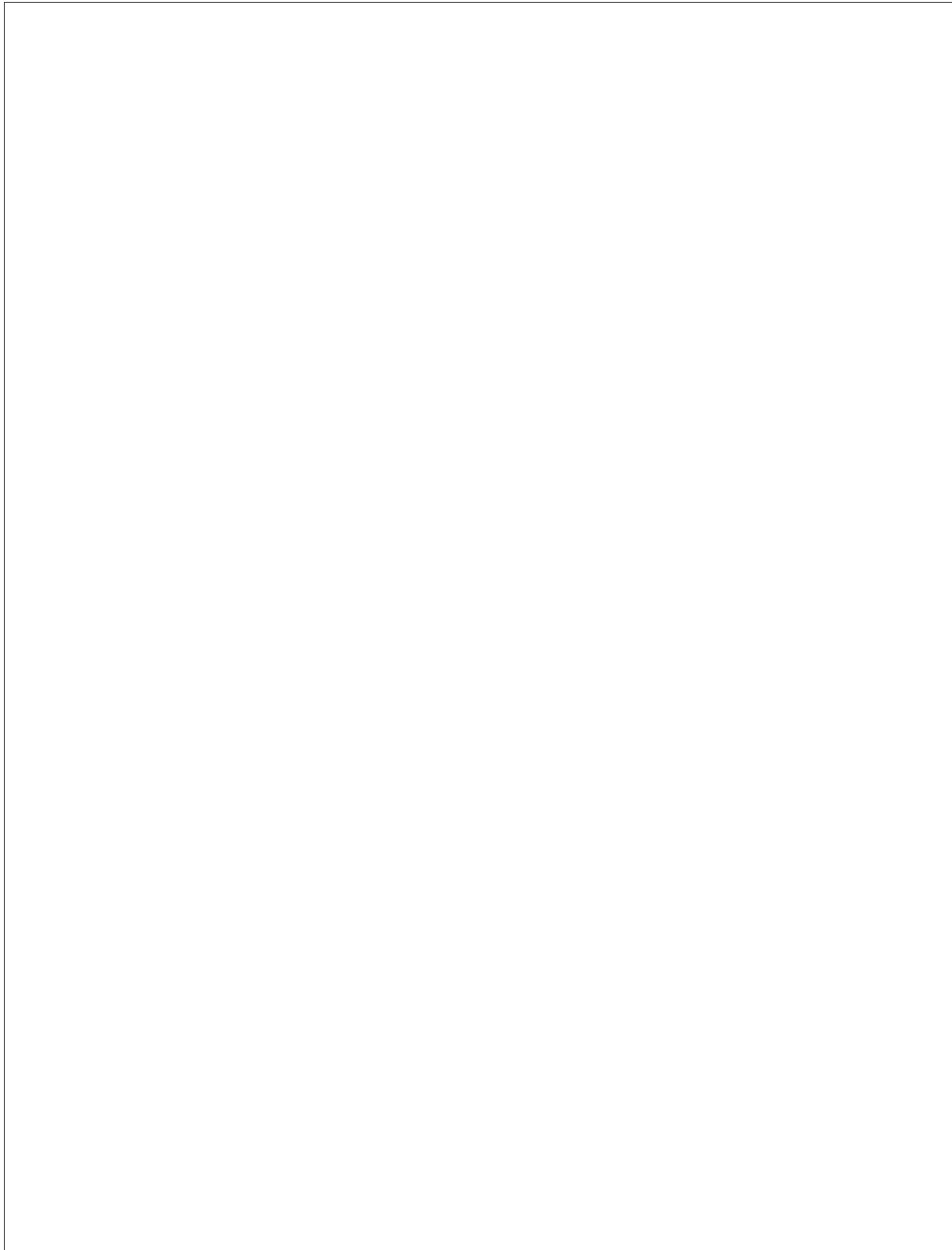
Consider a simple document editing system using the concepts of object-orientation and a simple markup language that annotates the content. **Document** is organized in form of multiple **Section** subtypes. A **Text** is a section that displays the content in form of paragraphs. Content for **Text** may contain multiple paragraphs each separated using a simple markup **###**. Moreover a **Table** is a section that displays the content in form of rows and columns. Markup for tabular content can be [{ , , }, { , , }] where square brackets (i.e. []) contain rows and curly braces (i.e. {}) contains columns, both separated by comma (,).

Some examples are given below:

Content	Output
It is a light-weight markup language for creating documents.###It is inspired from many other markup languages used to produce diverse and professional documents.###However, it is just experimental.	It is a light-weight markup language for creating documents. It is inspired from many other markup languages used to produce diverse and professional documents. However, it is just experimental.
[{word,synonym},{document,record},{deed},{language,expression}]	word synonym document record deed language expression


All sections shall be displayed in the order they appear in the document.

Write Java code to implement the given UML diagram and make this Document Editor possible





National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	Software Construction & Development	Section:	ALL
	Program:	BS (Software Engineering)	Semester:	Fall 2022
	Duration:	1 Hour	Total Marks:	40
	Evaluation Type:	Mid 1 Exam	Weight:	15 %
	Course Code:	SE3001	Page(s)	7
	Name:		Roll Number:	

Important Note:


- The quality of the code will affect the marks.
- Students will receive **ZERO** marks if the answers are plagiarized.
- Use of **mobile phones, internet, and, ANY type of smart devices** during the exam is strictly prohibited.
- Discussion with other students is not allowed.
- Exchange of notes and stationery with other students are not allowed.
- If any of the above rules is violated by the student. The invigilator has the right to file DC case against that student and the invigilator also has the right to take your exam away and ask you to leave the exam hall.

Question 1[CLO1]: Tracing

a) [4 marks] For each part, either check the box that indicates that a compiler error would occur, or write the output on the line associated with the code fragment.

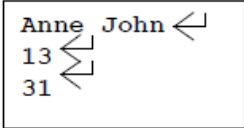
<pre>public class X { private void foo() { System.out.print("A"); } public void bar() { System.out.print("B"); foo(); } public static void main(String [] args) { X x = new X(); System.out.print("C"); x.bar(); } }</pre> <p>Compiler Error: <input type="checkbox"/></p> <p><u>OR</u></p> <p>Write Output:</p>	<pre>public class X { private void foo() { System.out.print("A"); } public void bar() { System.out.print("B"); foo(); } } public class Y { public void banana() { X x = new X(); System.out.print("C"); x.foo(); } } public static void main(String [] args) { Y y = new Y(); System.out.print("C"); y.banana(); }}</pre> <p>Compiler Error: <input type="checkbox"/></p> <p><u>OR</u></p> <p>Write Output:</p>
--	---

<pre> public class X { private int orange = 0; public void bar() { orange++; System.out.print("B" + orange); } public static void main(String [] args) { X x = new X(); System.out.print("C"); x.bar(); } } </pre> <p>Compiler Error: <input type="checkbox"/></p> <p><u>OR</u></p> <p>Write Output:</p>	<pre> public class X { public int orange = 0; public void bar() { orange++; System.out.print("B"); } } public class Y { public void banana() { X x = new X(); System.out.print("C"); x.orange; } public static void main(String [] args) { Y y = new Y(); System.out.print("D"); y.banana(); } } </pre> <p>Compiler Error: <input type="checkbox"/></p> <p><u>OR</u></p> <p>Write Output:</p>
--	---

- b) [20 marks] What do the following code fragments print? In some cases, spaces and empty lines are important in the final answer and must be clearly shown. To make this clear, you can indicate the presence of a space using the underscore character ('_') and indicate an empty line with the text "<Blank line>". If the code generates an error during execution, indicate/write the nature of the error. However, you must still note any output the program makes before the error occurs. Note, the symbol:  refers to the newline character when it appears in the input.

<pre> public class X { public static void main(String [] args) { System.out.println("Hi + \"SE3001\" + Students"); } } </pre> <p><u>Write Output:</u></p>	<pre> public class X { public static void main(String [] args) { System.out.println("Hi "); System.out.print("CS133"); System.out.println("Students"); } } </pre> <p><u>Write Output:</u></p>
---	---

<pre>public class X { public static void main(String [] args) { String s="Hi SE3001 Students!"; String t=null; System.out.println(s.substring(3,17)); s=t; System.out.println(s.charAt(1)); } }</pre> <p><u>Write Output:</u></p>	<pre>public class Rectangle { public static int area(int height, int width) { return height * width; } public static void main(String [] args) { System.out.println("The result of area is: " + Rectangle.area(4, 5)); } }</pre> <p><u>Write Output:</u></p>
<pre>public class X { public static void main(String [] args) { int mark=85; if(mark > 90) { System.out.println("Excellent"); }else if(mark < 90 mark > 80) { System.out.println("Good job!"); }else if (mark == 85) { System.out.println("Good job! your mark is exactly 10 marks above the average"); }else if (mark > 70) { System.out.println("Doing okay."); }else if(mark > 50) { System.out.println("Passed"); } else { System.out.println("Failed"); } } }</pre> <p><u>Write Output:</u></p>	<pre>public class X { public static void main(String [] args) { for (int i=10; i >0 ; i=i/2) { System.out.print(i + ","); } System.out.println("Good job!"); } }</pre> <p><u>Write Output:</u></p>

<pre>import java.util.*; public class X { public static void main(String [] args) { String name; int first; String second; Scanner keyBoardInput=new Scanner(System.in); System.out.println("Enter your name:"); name=keyBoardInput.nextLine(); System.out.println("Enter two numbers on different lines:"); first=keyBoardInput.nextInt(); second=keyBoardInput.nextLine(); System.out.println("Hi:" + name); System.out.println("first + second = " + first + second); } }</pre> <div data-bbox="154 808 485 934" data-label="Text"> <p>Input: </p> </div> <p><u>Write Output:</u></p>	<pre>public class C { public int x; } public class D { public static void f (C c, int y) { System.out.println(c.x); c.x = y; y++; System.out.println(c.x); c = new C(); c.x = y+2; System.out.println(c.x); } public static void main (String[] args) { int z = 4; C c = new C(); c.x = 3; System.out.println(c.x); f(c, z); System.out.println(c.x); System.out.println(z); } }</pre> <p><u>Write Output:</u></p>
<pre>public class SimpleCalc { public int value; public void calculate() { value += 7; } } public class MultiCalc extends SimpleCalc { public void calculate() { value += 3; } public void calculate(int multiplier) { calculate(); super.calculate(); value *= multiplier; } public static void main(String[] args) { MultiCalc calculator = new MultiCalc(); calculator.calculate(2); System.out.println(" Value is: " + calculator.value); }}</pre> <p><u>Write Output:</u></p>	<pre>class Pizza { java.util.ArrayList toppings; public final void addTopping(String topping) { toppings.add(topping); } } public class PepperoniPizza extends Pizza { public void addTopping(String topping) { System.out.println("Cannot and Uoppings"); } public static void main(String[] args) { Pizza pizza = new PepperoniPizza(); Pizza.addTopping("Mushrooms"); } }</pre> <p><u>Write Output:</u></p>

Question 2: [CLO1] [6 marks]

In this problem you are asked to write a simple class to represent elevators. An elevator has a current floor, a number of floors, a current number of passengers, and a maximum capacity of passengers.

An elevator:

- is constructed by specifying:
 - the total number of floors in the building
 - the maximum elevator capacity
 - that the elevator initially doesn't have any passengers,
 - and that the elevator is initially located on the bottom floor
- can move one floor up if not on the top floor
- can move one floor down if it is not on the bottom floor
- can accept a certain number of passengers (up to its maximum capacity)
- can drop off a certain number of passengers (no more than it actually has)
- can tell us which floor it's on.

Question 3: [CLO1] [5 marks]

Write a program that input English words using JOptionPane dialog and store the inputted words to the suitable collection. The program should have two methods: one to add the word into the suitable collection and other method will display all the stored words from the collection in the ascending order in an alert box (**Note:** *you cannot use Collections.sort() method for this program and your suitable collection should store distinct words only*)

Question 4: [CLO1] [5 marks]

Create a generic class that accept only numbers including short, integers, floats, and doubles only and must contains two generic methods namely **oddSum** and **evenSum** that returns the sum of odd numbers and even numbers respectively that stored within an arraylist of accepted numbers.

Good
Luck!

