**Day 23 – 02nd Aug 2025**

**Day 23 - 02nd Aug 2025**

**JUNIT testcases - atleast one or two if working pla raise your hands.**

**TASK 01**

**Tags Annotation:**

**import org.junit.jupiter.api.Test;**

**import org.junit.jupiter.api.Tags;**

**import org.junit.jupiter.api.Assertions.assertEquals;**

**class TestCase02 {**

**@Test**

**@Tags("firstPriority")**

**void testMethod01() {**

**}**

**@Test**

**@Tag("firstPriority")**

**void runTestcase02() {**

**}**

**@Tags("fastTag")**

**void testMethod03() {**

**}**

**@Test**

**@Tag("slowTag")**

**void runTestcase04() {**

**}**

**}**

**package lib;**

**import org.junit.jupiter.api.Test;**

**import org.junit.jupiter.api.Tag;**

**import static org.junit.jupiter.api.Assertions.*assertEquals*;**

**class TestCase02 {**

**@Test**

**@Tag("firstPriority")**

**void testMethod01() {**

**int expected = 2;**

**int actual = 1 + 1;**

***assertEquals*(expected, actual);**

**}**

**@Test**

**@Tag("firstPriority")**

**void runTestcase02() {**

**String str = "JUnit";**

***assertEquals*("JUnit", str);**

**}**

**@Test**

**@Tag("fastTag")**

**void testMethod03() {**

***assertEquals*(4, 2 \* 2);**

**}**

**@Test**

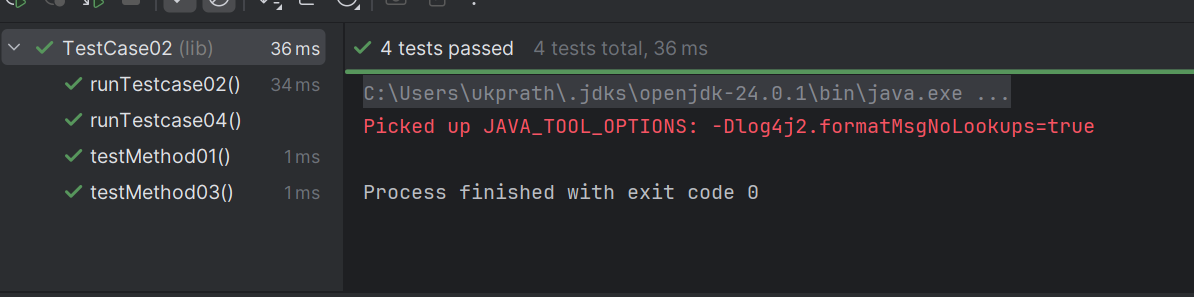
**@Tag("slowTag")**

**void runTestcase04() {**

***assertEquals*(10, 5 + 5);**

**}**

**}**

****

**—----------------------------------------------------------------------------------------------------------------------**

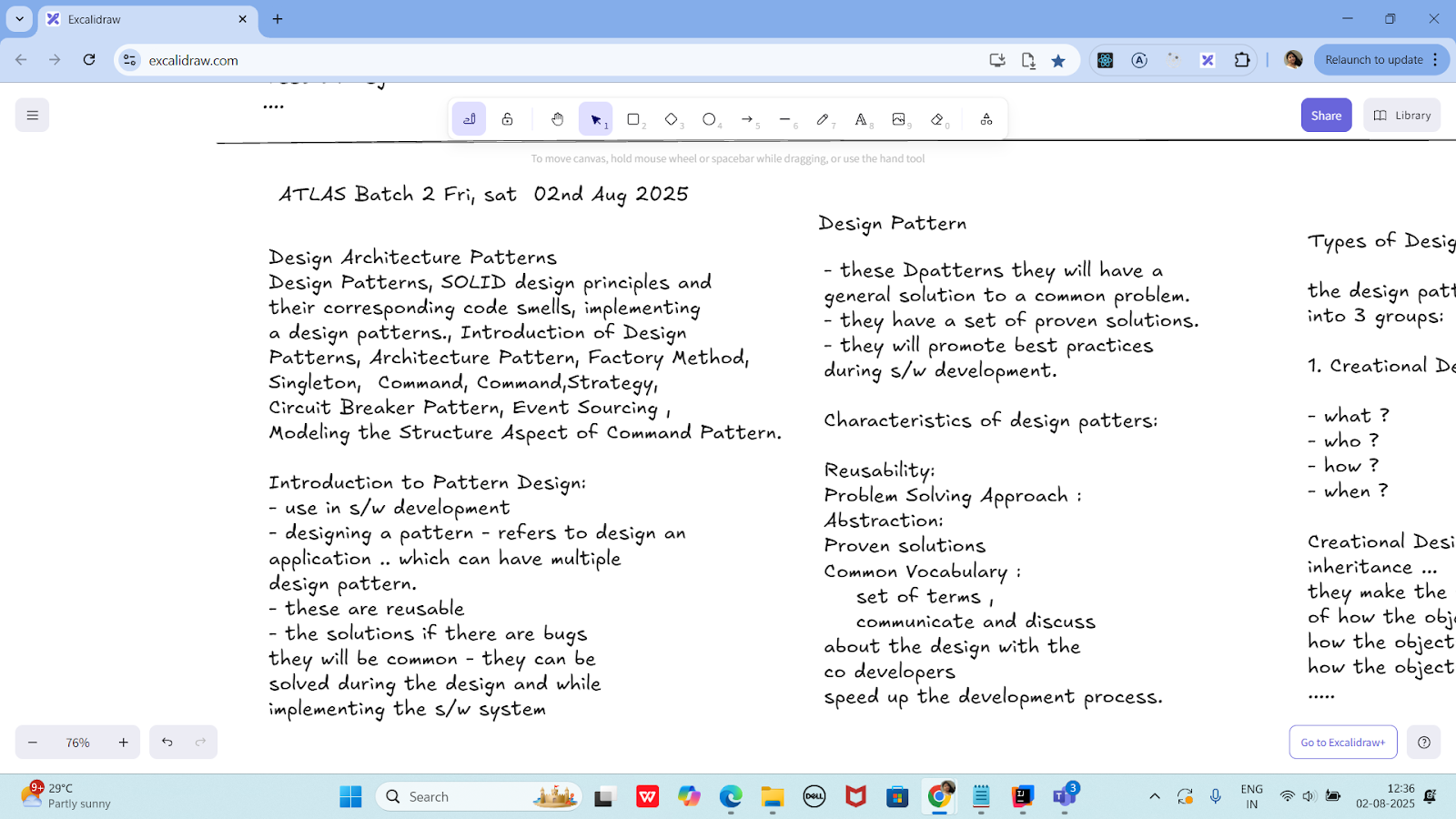
**Design patters:**

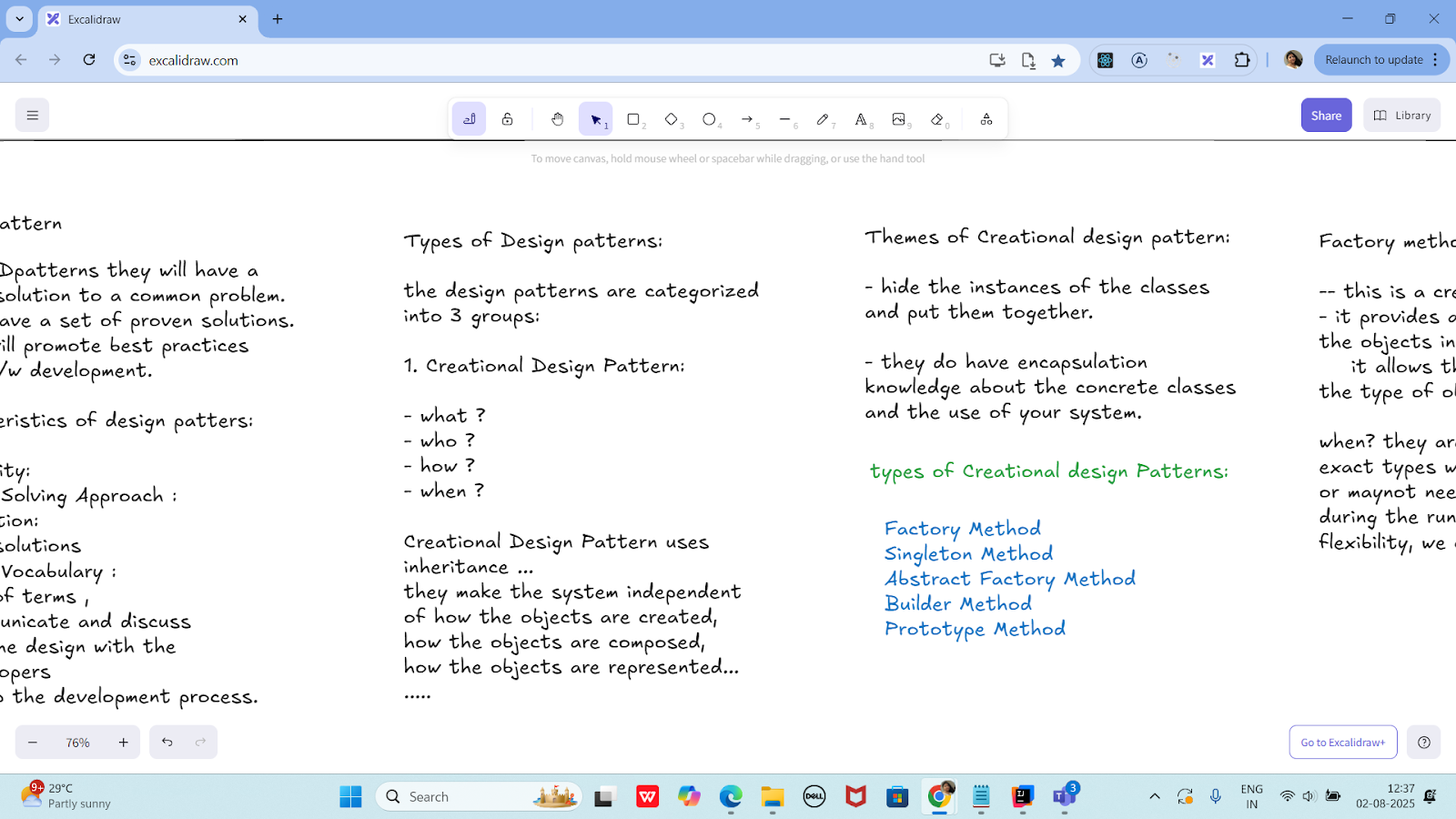
1. **Creational Design Patterns:**

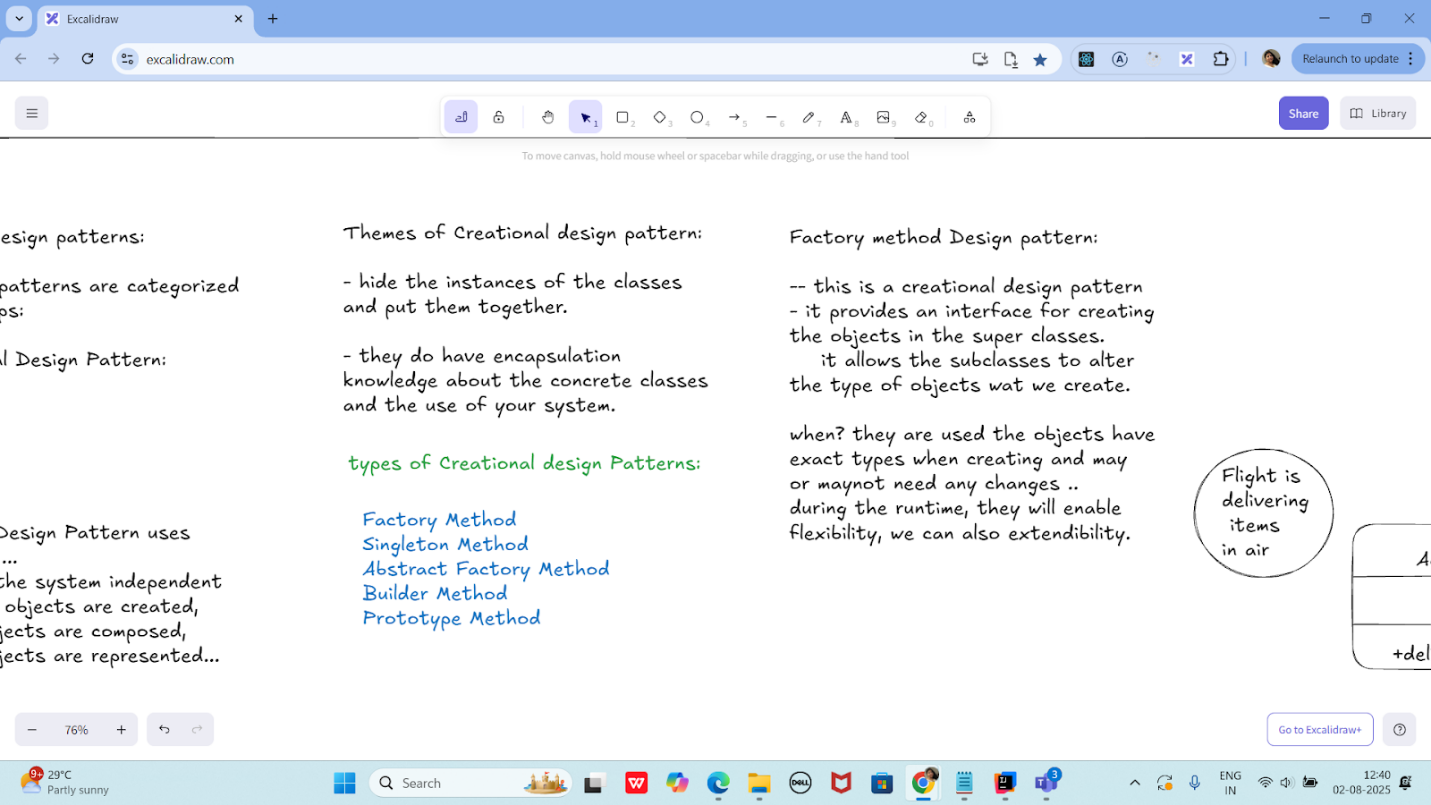
**types of Creational design Patterns:**

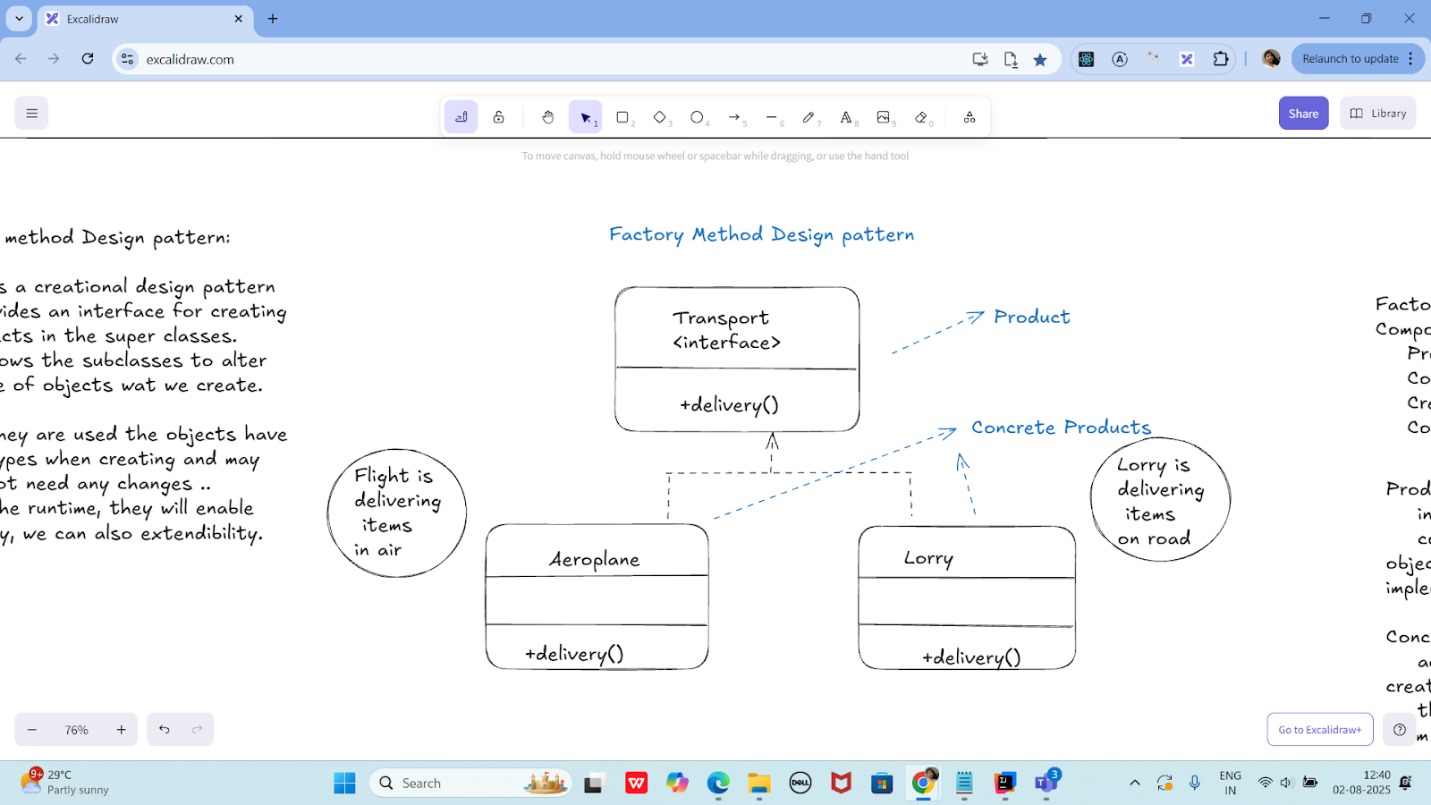
1. **Factory Method**
2. **Singleton Method**
3. **Abstract Factory Method**
4. **Builder Method**
5. **Prototype Method**

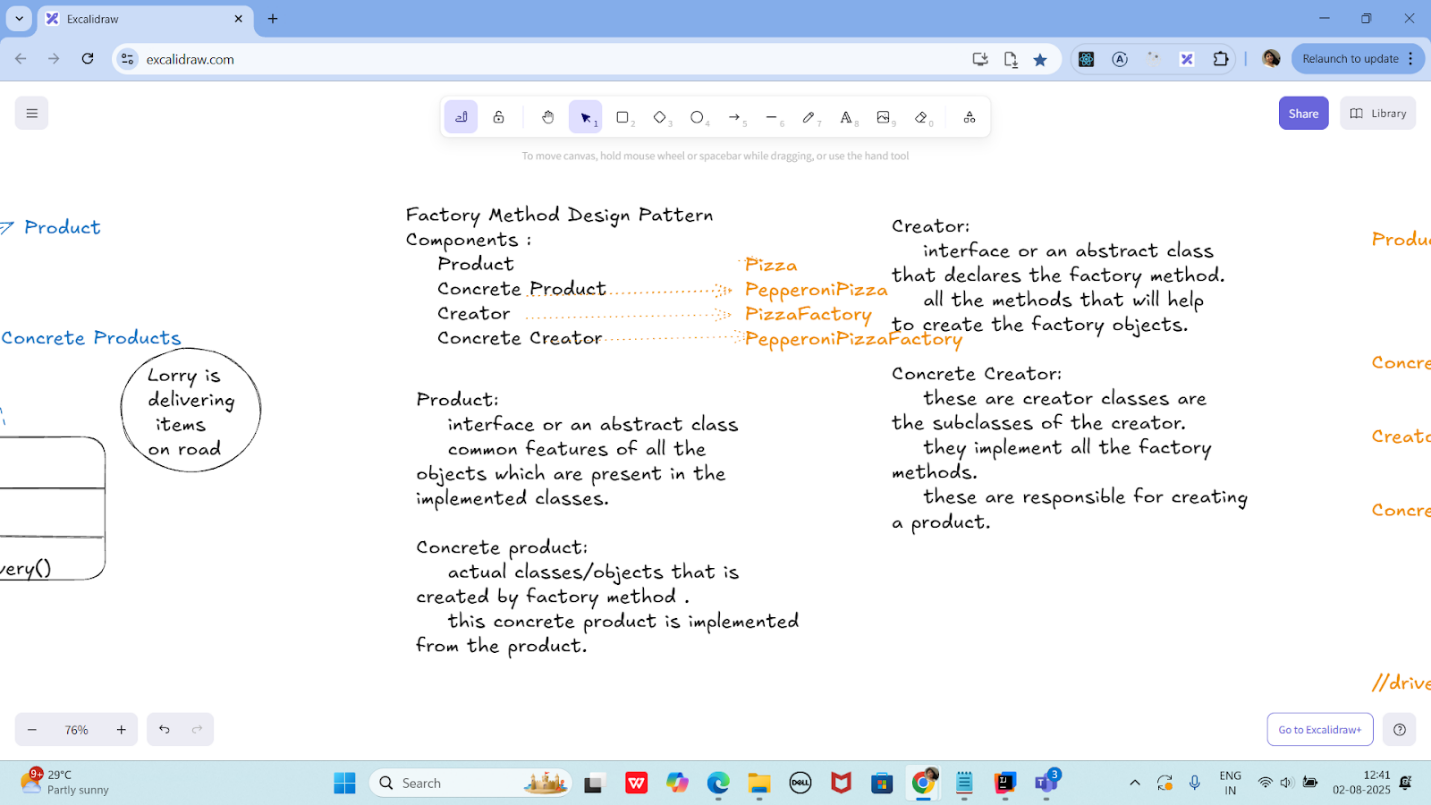
**Factory Method Design Pattern**

****

****

****

****

****

**Product :->       Pizza  interface/abstract class**

**preparation();**

**baking();**

**cutting();**

**boxing();**

**ConcreteProduct --> PepperoniPizza**

**//implement all the features of Pizza.**

**Creator --->          PizzaFactory interface/abstract class**

**Pizza createPizza(); - type is Pizza**

**Concretecreator ---> PepperoniPizzaFactory**

**@Override**

**Pizza createPizza() {**

**....**

**retrun new PepperoniPizza();**

**}**

**//driver class**

**class Sams {**

**psvm() {**

**PizzaFactory pfobj = new PepperoniPizzaFactory();**

**Pizza pobj = pfobj.createPizza();**

**}**

**}**

**public abstract class Pizza {**

**public abstract void preparation();**

**public abstract void baking();**

**public abstract void cutting();**

**public abstract void boxing();**

**}**

**public class PepperoniPizza extends Pizza {**

**@Override**

**public void preparation() {**

**System.*out*.println("Preparing pepperoni toppings and dough.");**

**}**

**@Override**

**public void baking() {**

**System.*out*.println("Baking pepperoni pizza at 350 degrees.");**

**}**

**@Override**

**public void cutting() {**

**System.*out*.println("Cutting pepperoni pizza into slices.");**

**}**

**@Override**

**public void boxing() {**

**System.*out*.println("Boxing pepperoni pizza in official box.");**

**}**

**}**

**public abstract class PizzaFactory {**

**public abstract Pizza createPizza();**

**}**

**public class PepperoniPizzaFactory extends PizzaFactory {**

**@Override**

**public Pizza createPizza() {**

**System.*out*.println("Creating a PepperoniPizza via factory...");**

**return new PepperoniPizza();**

**}**

**}**

**public class Sams {**

**public static void main(String[] args) {**

**PizzaFactory pfobj = new PepperoniPizzaFactory();**

**Pizza pobj = pfobj.createPizza();**

**pobj.preparation();**

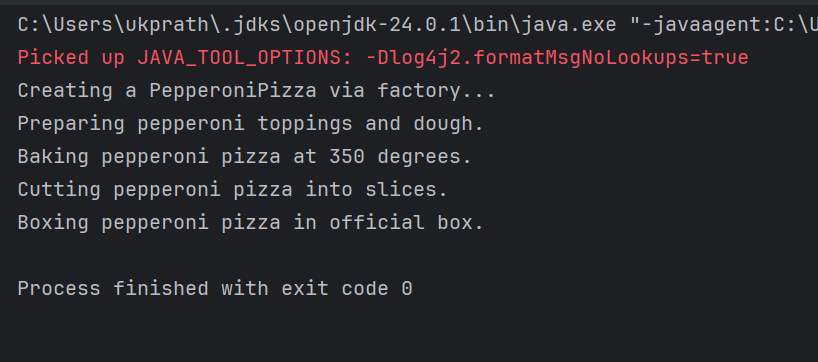
**pobj.baking();**

**pobj.cutting();**

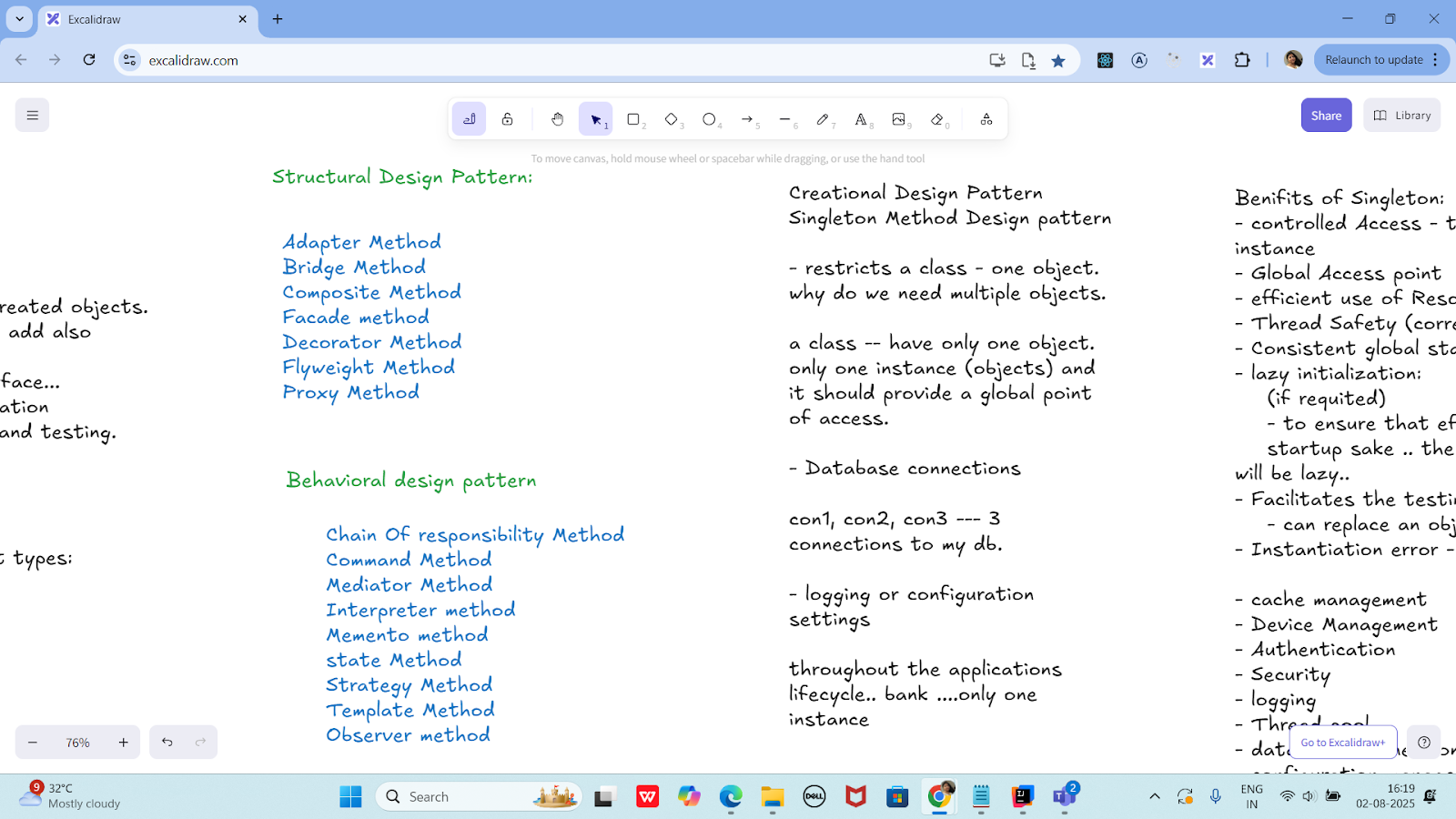
**pobj.boxing();**

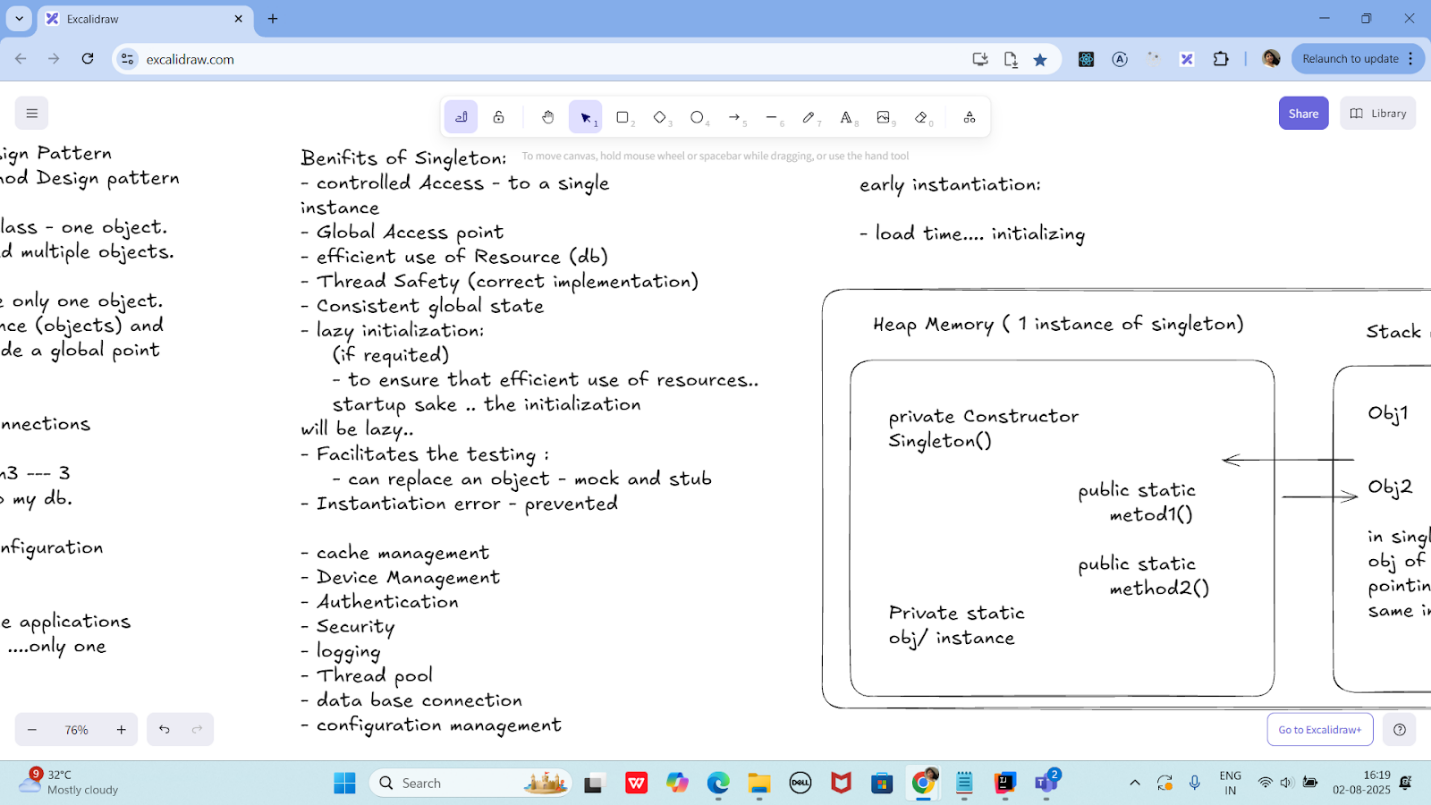
**}**

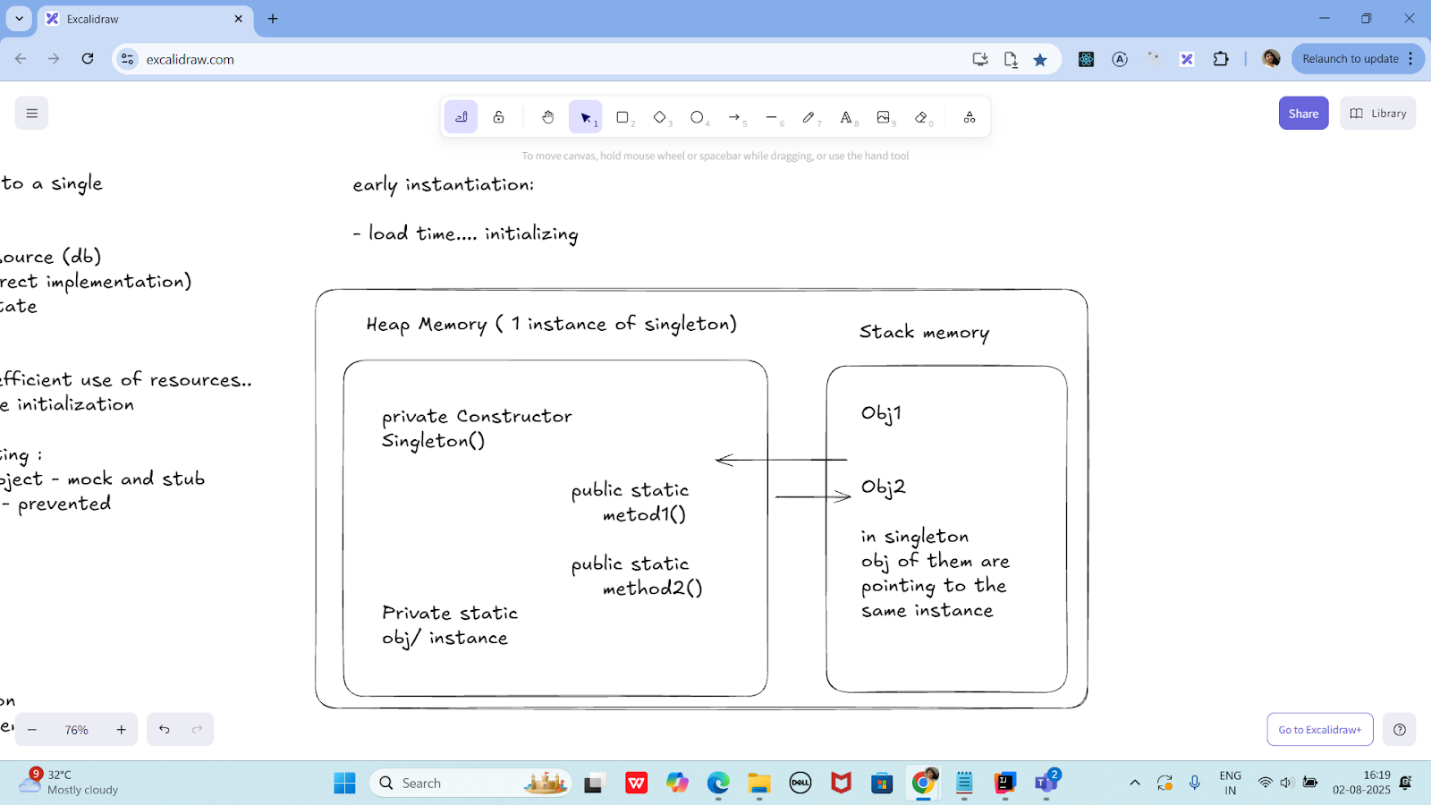
**}**

****

**Singleton Method Design principle**

****

****

****

**class SingletonDemo {**

**private static SingletonDemo *instance*; // lodinids.. //pass/ pin no**

**private SingletonDemo() {**

**System.*out*.println("initiating the singleton");**

**}**

**public static SingletonDemo getInstance() {**

**if (*instance* == null) {**

***instance* = new SingletonDemo();**

**System.*out*.println("in get instance");**

**}**

**return *instance*;**

**}**

**public static void doHere() {**

**System.*out*.println("doing here some thing");**

**}**

**}**

**public class SingletonDP {**

**public static void main(String[] args) {**

**SingletonDemo.*getInstance*().*doHere*();**

**//        SingletonDemo obj = new SingletonDemo();**

**//        obj.doHere();**

**//        SingletonDemo obj2 = new SingletonDemo();**

**//        obj2.doHere();**

**}**

**}**

**—----------------------------------------------------------------------------------------------------------------------------------**

**—----------------------------------------------------------------------------------------------------------------------------------**

**Home Tasks:**

**Home Task 1:**

**Write a test case for the below java file**

**public class Junit4Test {**

**public int compare(int n1, int n2) {**

**if (n1 > n2) return 1;**

**return -1;**

**}**

**}**

**import org.junit.Test;**

**import static org.junit.Assert.*assertEquals*;**

**public class Junit4TestTest {**

**@Test**

**public void testCompare\_WhenFirstIsGreater() {**

**Junit4Test obj = new Junit4Test();**

**int result = obj.compare(10, 5);**

***assertEquals*(1, result);**

**}**

**@Test**

**public void testCompare\_WhenSecondIsGreater() {**

**Junit4Test obj = new Junit4Test();**

**int result = obj.compare(3, 8);**

***assertEquals*(-1, result);**

**}**

**@Test**

**public void testCompare\_WhenEqual() {**

**Junit4Test obj = new Junit4Test();**

**int result = obj.compare(7, 7);**

***assertEquals*(-1, result); // Because your logic returns -1 even if both are equal**

**}**

**}**

**Home task 2:**

**public int compare(int n1, int n2) {**

**if (n1 > n2) return 1;**

**else if (n1 < n2) return -1;**

**return 0;**

**}**

**public class CompareUtil {**

**public int compare(int n1, int n2) {**

**if (n1 > n2) return 1;**

**else if (n1 < n2) return -1;**

**return 0;**

**}**

**}**

**import org.junit.Test;**

**import static org.junit.Assert.*assertEquals*;**

**public class CompareUtilTest {**

**CompareUtil obj = new CompareUtil();**

**@Test**

**public void testCompare\_FirstGreater() {**

***assertEquals*(1, obj.compare(10, 5));**

**}**

**@Test**

**public void testCompare\_SecondGreater() {**

***assertEquals*(-1, obj.compare(4, 9));**

**}**

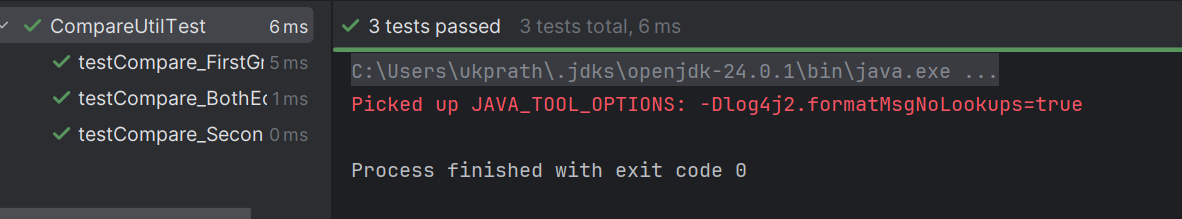
**@Test**

**public void testCompare\_BothEqual() {**

***assertEquals*(0, obj.compare(7, 7));**

**}**

**}**

****

**Hoe Task 3:**

**Include a test case and run the test suite.**

**Test Suite:**

**- its a bundle of unit test cases and run them**

**all together.**

**@Suite**

**@RunWith**

**TestSuite Demo**

**import org.junit.runner.RunWith;**

**import org.junit.runners.Suite;**

**@RunWith(Suite.class)**

**@Suite.SuiteClasses( {**

**JunitTest01.class,**

**JunitTest02.class,**

**JunitTest03.class**

**})**

**public class TestSuiteDemo {**

**}**

**JunitTest01.java**

**import org,junit.Test;**

**import static org.junit.Assert.assetEquals;**

**import org.junit.Igore;**

**public class junitTest01 {**

**String msg = "running test01 ";**

**MessageUtil msgUtilobj = new MessageUtil(msg);**

**//MessageUtil is a utility class**

**//store a message**

**@Test**

**public void msgTest() {**

**sout("we are inside the msgtest()");**

**assertEquals(msg, msgUtilobj.printMessage());**

**}**

**}**

**junitTest02.java**

**test Runner**

**public class MessageUtil {**

**private String message;**

**public MessageUtil(String message) {**

**this.message = message;**

**}**

**public String printMessage() {**

**return message;**

**}**

**}**

**import org.junit.Test;**

**import static org.junit.Assert.*assertEquals*;**

**public class JunitTest1 {**

**String msg = "running test01 ";**

**MessageUtil msgUtilobj = new MessageUtil(msg);**

**@Test**

**public void msgTest() {**

**System.*out*.println("Inside msgTest()");**

***assertEquals*(msg, msgUtilobj.printMessage());**

**}**

**}**

**import org.junit.Test;**

**import static org.junit.Assert.*assertEquals*;**

**public class JunitTest2 {**

**String msg = "running test02 ";**

**MessageUtil msgUtilobj = new MessageUtil(msg);**

**@Test**

**public void msgTest() {**

**System.*out*.println("Inside JunitTest02 msgTest()");**

***assertEquals*(msg, msgUtilobj.printMessage());**

**}**

**}**

**import org.junit.Test;**

**import static org.junit.Assert.*assertEquals*;**

**public class JunitTest3 {**

**String msg = "running test03 ";**

**MessageUtil msgUtilobj = new MessageUtil(msg);**

**@Test**

**public void msgTest() {**

**System.*out*.println("Inside JunitTest03 msgTest()");**

***assertEquals*(msg, msgUtilobj.printMessage());**

**}**

**}**

**import org.junit.runner.RunWith;**

**import org.junit.runners.Suite;**

**@RunWith(Suite.class)**

**@Suite.SuiteClasses({**

**JunitTest1.class,**

**JunitTest2.class,**

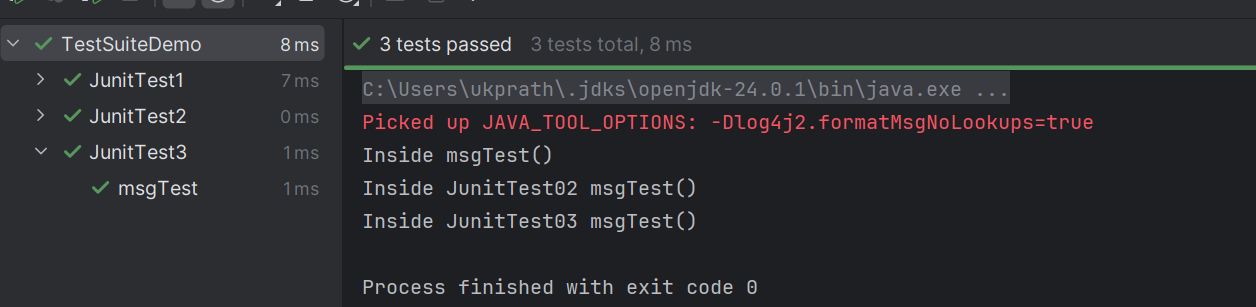
**JunitTest3.class**

**})**

**public class TestSuiteDemo {**

**// No code needed here — just grouping tests**

**}**

****

**Home Task 4:**

**Implement the above code also add one more test case and test runner.. And execute the test suite.**

**================================================================================================================================================**

**Info Box**

**================================================================================================================================================**

**Info Box (Carry forward from Day 22 02nd Aug juz fyi..**

**Plz refer Best Programming Practices - Doc 08 in Docs to study**

**Updated link at 12.17 ..**

[**https://excalidraw.com/#json=lKmZDbhZ5V6ZRlQ3zJOXv,K6r0FziLUTf3htjdUrnN4w**](https://excalidraw.com/#json=lKmZDbhZ5V6ZRlQ3zJOXv,K6r0FziLUTf3htjdUrnN4w)

**Updated link at 15.57**

[**https://excalidraw.com/#json=Mp0F2MJiO57HKm74LVHXK,XmhQybYlqAswiHdRIeokDQ**](https://excalidraw.com/#json=Mp0F2MJiO57HKm74LVHXK,XmhQybYlqAswiHdRIeokDQ)

**Junit dependency**

[**https://junit.org/junit4/dependency-info.html**](https://junit.org/junit4/dependency-info.html)

**<dependencies >**

**<dependency>**

**<groupId>junit</groupId>**

**<artifactId>junit</artifactId>**

**<version>4.13.2</version>**

**</dependency>**

**</dependencies >**

**The above dependency you need to add in the dependencies tag of  pom.xml..**

**Junit Official link**

**Download**

**Hamcrest  and Junit.jar file**

[**https://junit.org/**](https://junit.org/)

**Or add the below dependency**

**<dependency>**

**<groupId>org.junit.jupiter</groupId>**

**<artifactId>junit-jupiter-api</artifactId>**

**<version>6.0.0-M2</version>**

**</dependency>**

**Hamcrest dependency**

**<!-- https://mvnrepository.com/artifact/org.hamcrest/hamcrest-junit -->**

**<dependency>**

**<groupId>org.hamcrest</groupId>**

**<artifactId>hamcrest-junit</artifactId>**

**<version>2.0.0.0</version>**

**<scope>test</scope>**

**</dependency>**

**JUNIT Jupiter api - junit.jar file link**

[**https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api**](https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api)

**Junit Hancrest**

[**https://mvnrepository.com/artifact/org.hamcrest/hamcrest**](https://mvnrepository.com/artifact/org.hamcrest/hamcrest)

**================================================================================================================================================**