06/07/2025:

**Exception Propagation:**

If a method throws an exception and does not handle it with a try catch then the exception goes(or propagates) to the caller method. If that caller also does not handle it. It goes to its caller. And so on, upto the jvm which will crash the program if no one handles it.

Example:

Class example

Here method b throws arithmetic exception, no try catch in method b() so it goes to method a().

Method a() also does not catch it,so it propagates to main.

Main() does not catch it, so it goes to JVM and the JVM handles it by printing the error and stopping the program and this flow is called exception propagation.

public class Example{

    void a(){

    }

    void b(){

        int x = 5/0;

    }

    public static void main(String[] args) {

        Example e = new Example();

        e.a();//Here method B throws Arithmrtic exception

    }

}

Rules:

Only **unchecked exceptions** like Arithmetic exceptions, null pointer exceptions propagate automatically. Checked exceptions like IO Exception must be declared with throws or caught with try-catch block.

Question:

Design a university course registration system with proper object-oriented principles and exception propagation.

An abstract class Course with fields: courseld, title, and credits.

Subclasses: LabCourse and Theory Course, which override calculateFees().

credits is stored using the Integer wrapper class.

If credits are negative or greater than 10, throw a custom exception InvalidCredits Exception.

Demonstrate exception propagation by not catching the exception in the Course constructor or subclass, but only handling it in main().

Use polymorphism to calculate fees for multiple courses.

Fees Rules:

Theory Course: ₹1500 per credits

LabCourse: ₹1500 per credit plus ₹2000 lab fee

Constraints

credits: between 1 and 10

use an ArrayList to store the courses

let the exception propagate up to main

**MAP:**

A map in Java is a data structure to store **key:value pairs.**  A map allows you to store, retrieve, update and delete values based on a **unique key.** Unlike a list or set, which stores single elements. Map is built for, associating data.

**Types of Map:**

Java provides several implementation in the java.util package.

1. HashMap:

It stores key-value pairs with no guaranteed order. It uses a hash table internally(hashing of the key).Null keys and values are allowed.

Hashing = smart formula to jump directly to the storage location of data.

1. **Tree Set:**

It stores a key value pair in sorted order of the keys.

It uses a red-black tree internally.

No null keys(throws NullPointerException)

It is good if we need data sorted by keys.

1. **LinkedHashMap:**

It keeps insetion order.

Under the hood it uses a hashtable+linkedlist.

It is useful if we want predicatability in iteration order.

**Usages:**

HashMap- Fast lookups, no ordering req.

TreeMap- Sorted order of keys.

LinkedHashMap- Preserve insertion order.

**Problem statement:**

Given a list of the student names and their marks.

* Design a program that stores the names and marks in the hashmap.
* Allows the user to search by name.
* Update the marks if we the user requests.
* Print the top scorer.

**Common map methods:**

1. Put(k,v), k->key and v->value 🡪 It is used to insert or update.
2. Get(k,v) 🡪 It is used to fetch.
3. Remove() 🡪 It is used to delete.
4. Containskey() 🡪 It is used to check
5. Keyset()
6. Values() 🡪 It is used to get all.
7. Entryset() 🡪 Get key value pairs as entry objects.

**Problem Statement**

**Aliens buy products in a shop. Each allen has an ID and their cart total.**

**Use HashMap<Integer, Double> to store alienID totalPurchaseAmount.**

**Insert 4 aliens with random purchases.**

**Find the alien with maximum spending.**

**If totalPurchaseAmount < 0, throw InvalidAmountException**

Solution: