15-May-2025

__/__/____

Lecture-4

UML Diagrams.

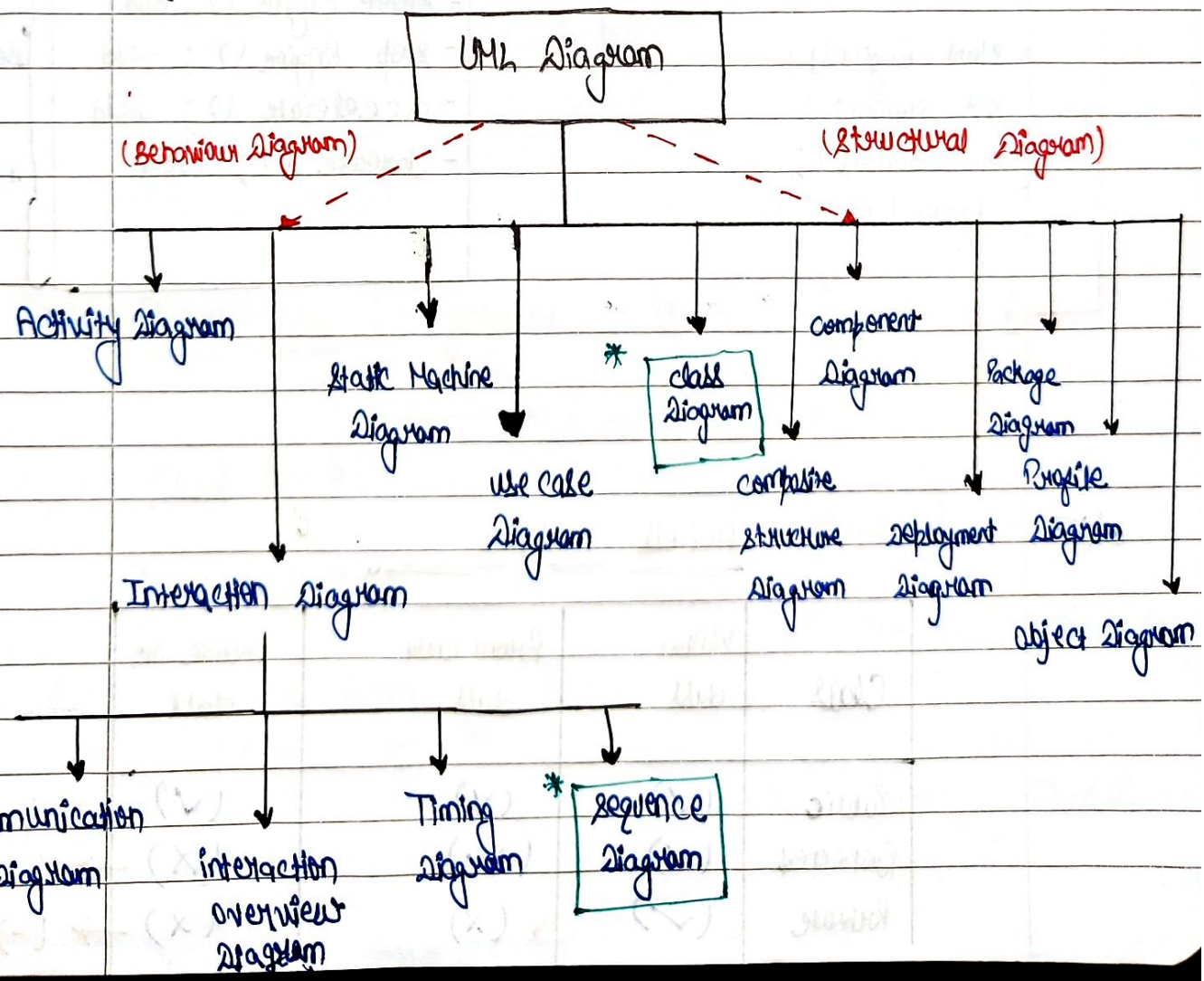
→ Unified Modeling Language (UML) is a general-purpose Modeling language.

→ The main aim of UML is to define a standard way to visualize the way a system has been designed.

→ It is quit similar to blueprint used in other field of engineering.

• Note :- UML is not a programming language, it is rather a visual language.

Type of UML Diagram :-



__/__/____

Class Diagram :-

- Class Structure
- Association / Connection

The most widely use UML diagram is the class diagram. We use class diagram to depict the static structure of a system by showing system classes, their Method and attributes.

```
class car {

String brand;
String model;
int engine;

void
start Engine();
stop engine();
accelerate();
brake();

}
```

<< abstract >> or not

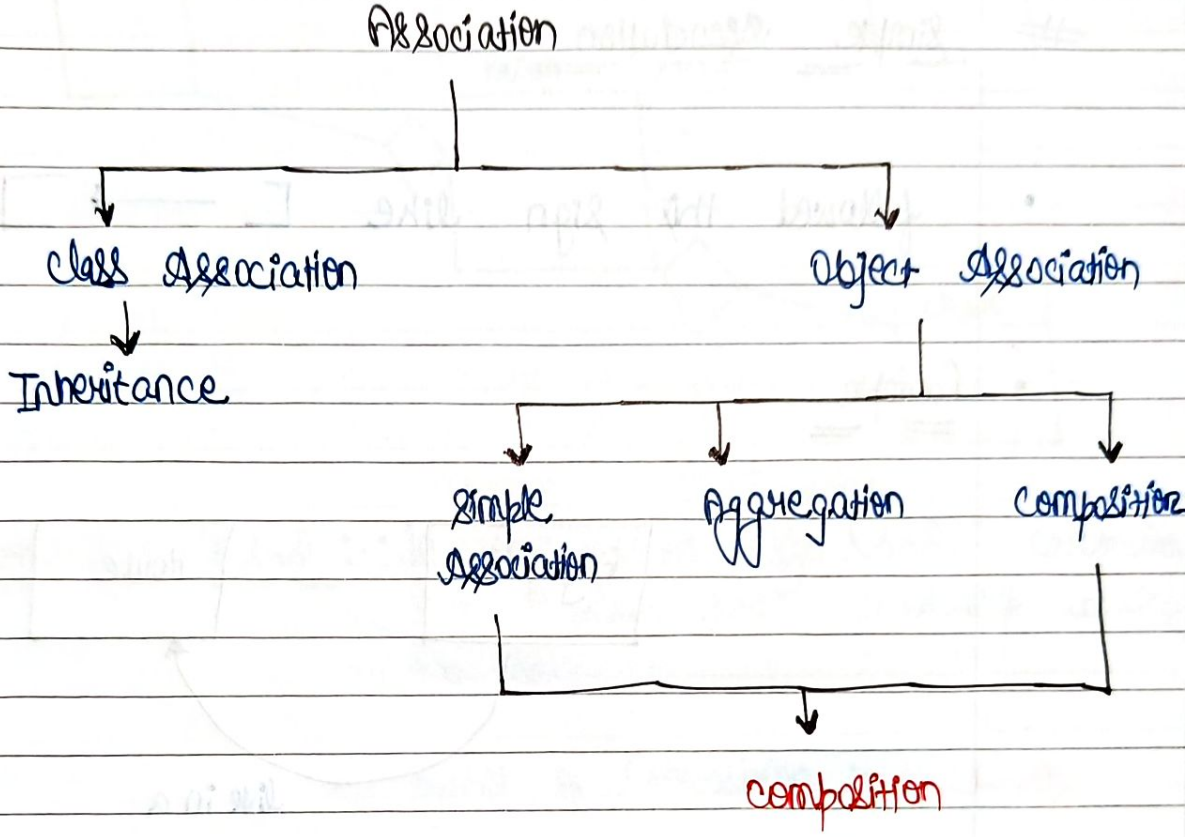
Car	← class name
+brand : String +Model : String #engine cc : int	← character or variable
- start Engine () : void - stop Engine () ; void - accelerate () ; void - brake () ; void	← Behaviour or Method

Access Modifier

Class	Within class	From child class	outside the class	
Public	(✓)	(✓)	(✓)	→ (+)
Protected	(✓)	(✓)	(✗)	→ (#)
Private	(✓)	(✗)	(✗)	→ (-)

__/__/____

Association :- it is defined as the two classes are interelated to each other is called association.

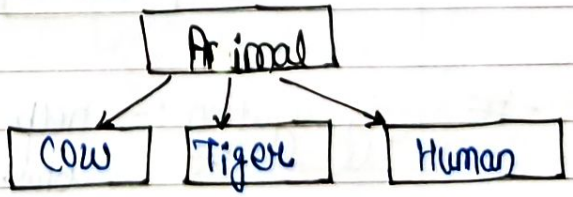


Inheritance :- followed (is-a) relationship

(→ is-a)

```
Class A {

  Method 1();
};
class B : Public A {
   Method 2();
};
Main () {
   B* b = new B();
   b → method();
}
```



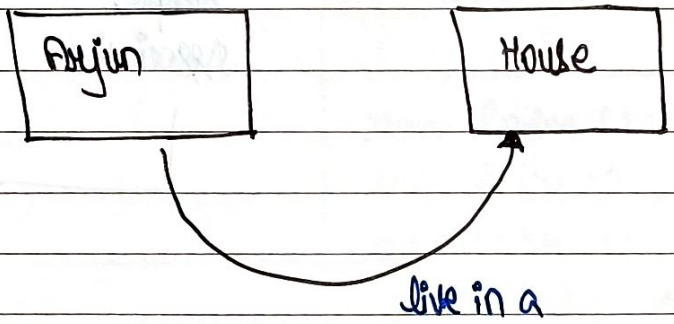
Cow is-a Animal.

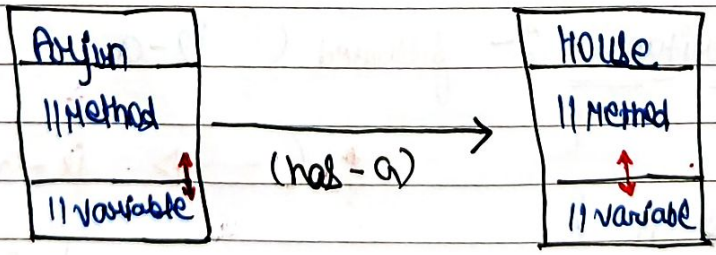
__/__/____

Composition :- followed (has-a) relationship

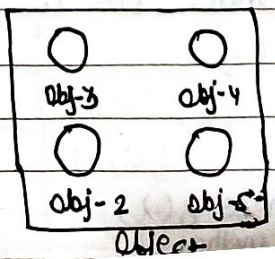
Simple association.

- followed this sign like [⟶]
- Example



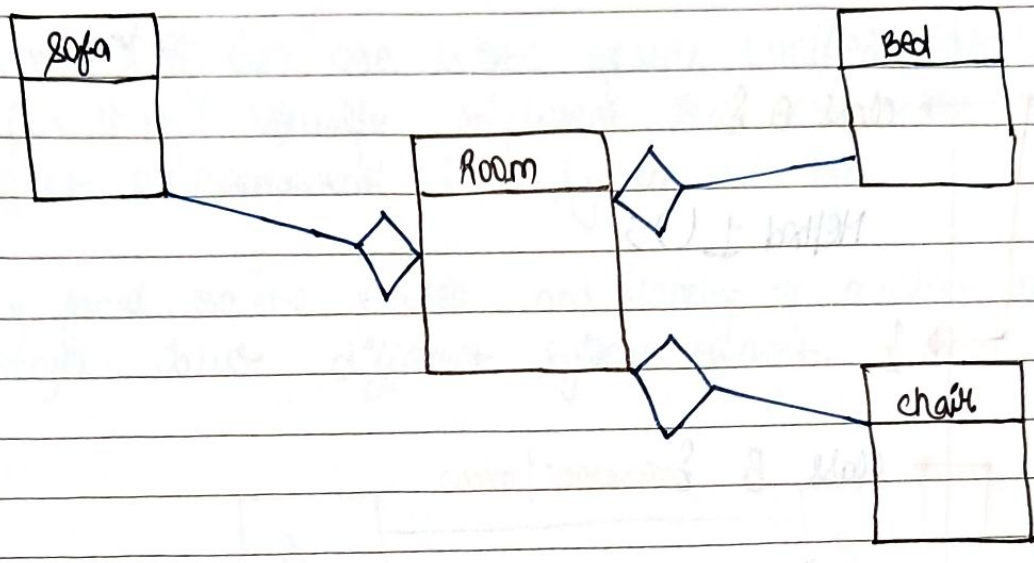


Aggregation :- help in accurately representing relationship b/w objects.



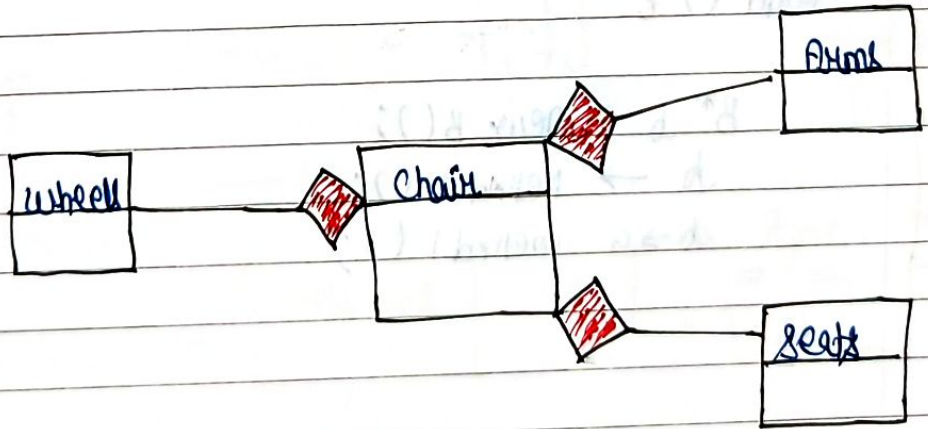
__/__/____

→ Aggregation is represented through this sign like :-



Composition in LLD :- • In low-level design (LLD), composition is a "whole-part" relationship between two object.

• Symbol of Composition :-



__/__/____

Source code:-

```
class A {
    Method 1 ();
}

class B {
    A* a;
    B() { a = new A() }
    Method 2 ();
}

Main () {
    B* b = new B();
    b → Method 2 ();
    b → a  method 1 ();
}
```

__/__/____

Sequence Diagram :-

Sequence Diagram are a type of UML (unified Modeling Langu
design that visually represent the interaction betw
object or component in a system over time.

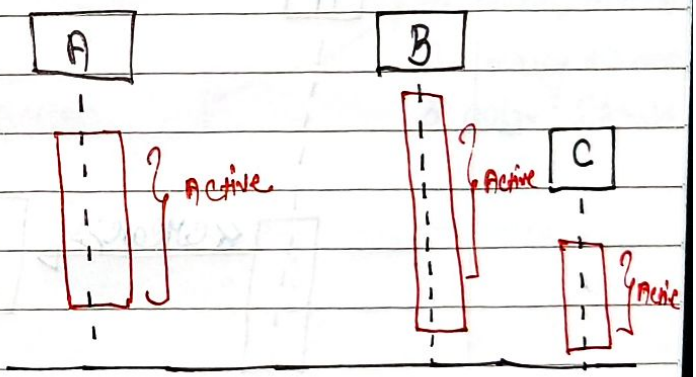
They focus on the order and timing of message or event
exchanges betw different system element.

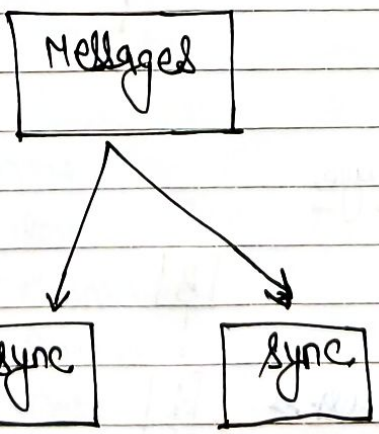
1. 

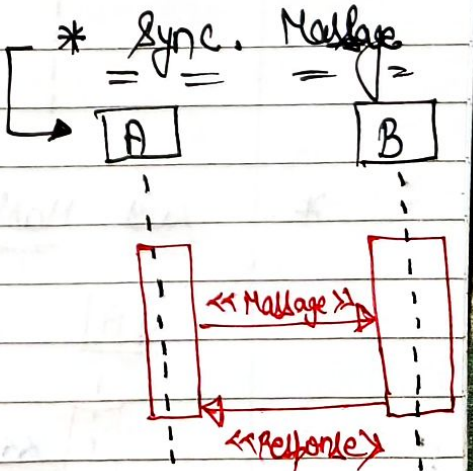
2. Lifeline

3. Activation Bar

4. Message

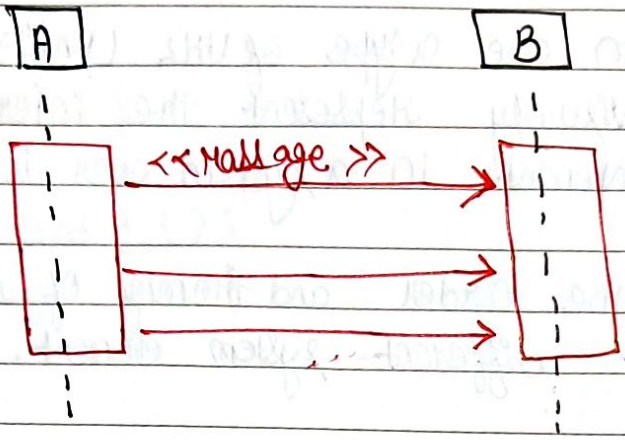




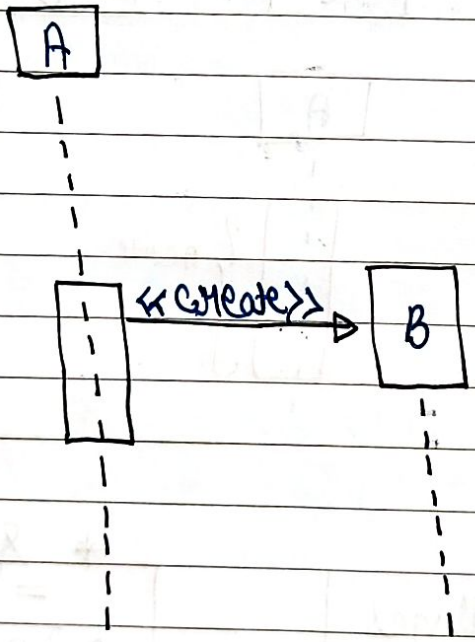


__/__/____

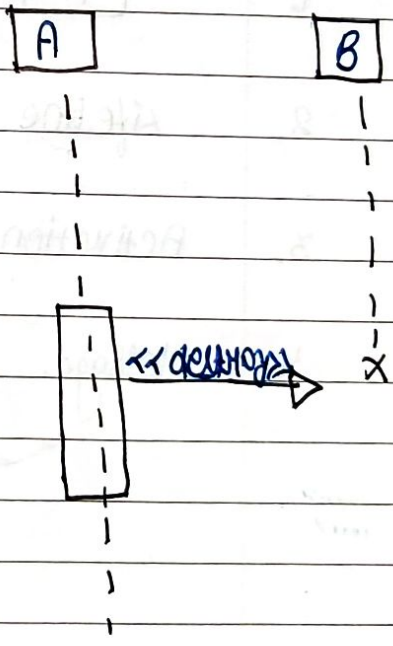
* Async. Message



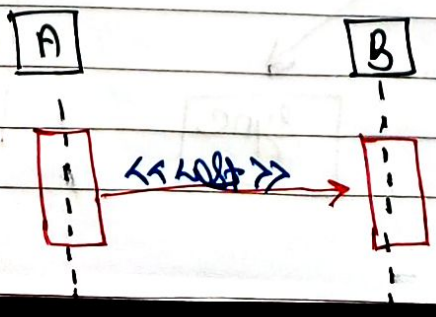
* Create Message



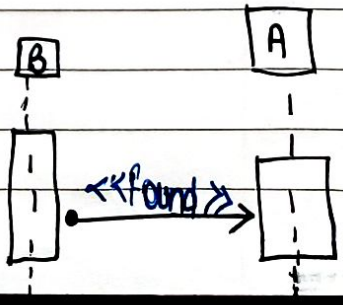
* Destroy Message



* Lost Message

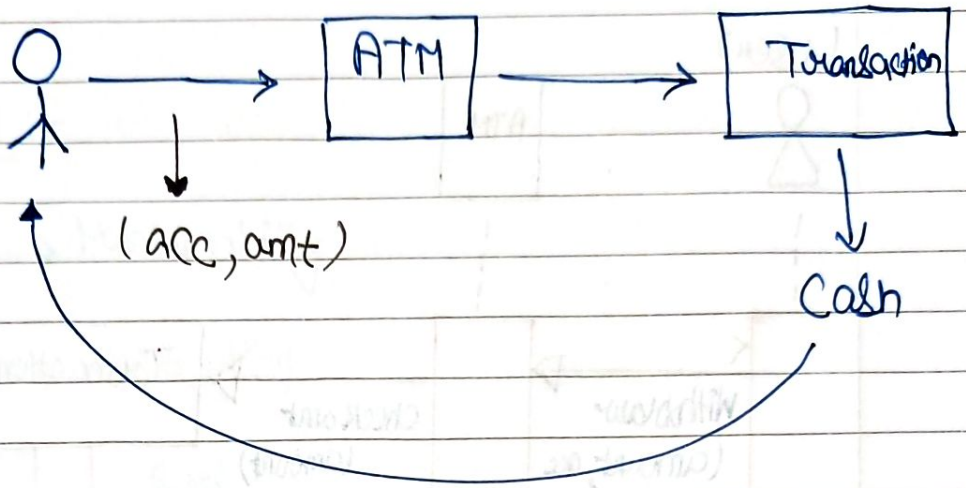


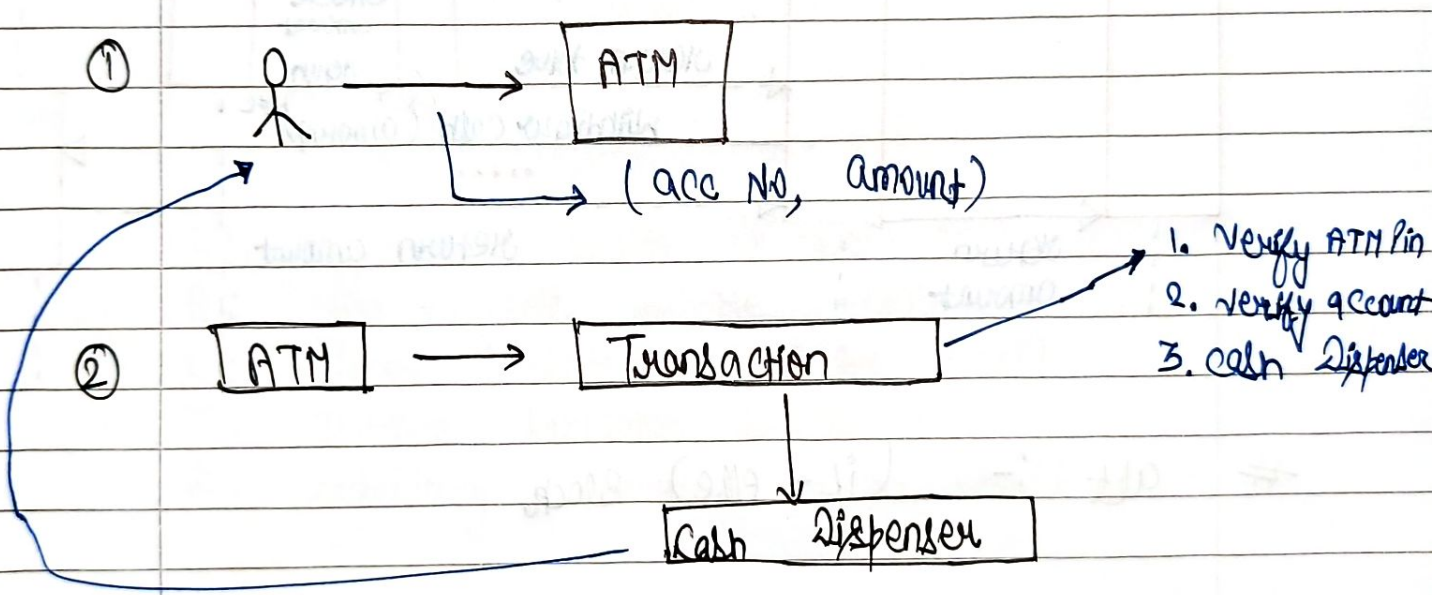
Found Message



__/__/__

How to Draw Sequence Diagrams ?



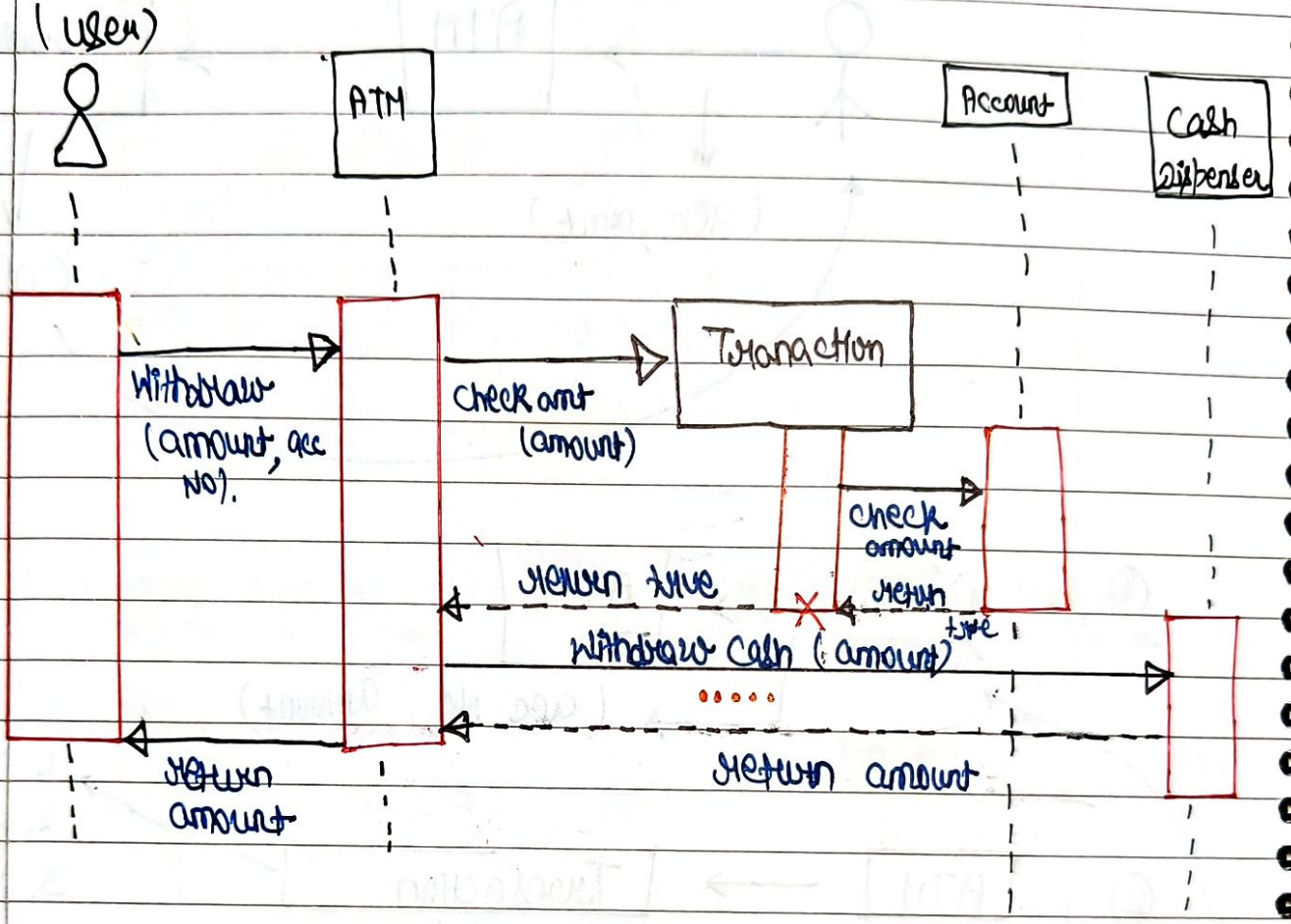


Sequence Diagram.

1. Use Case
2. Object
3.
   - ATM
   - User
   - Transaction
   - Account
   - Cash Dispenser

__/__/____

Sequence Diagram of ATM :-



alt :- (if - else) Block

option :- (if) Block

loop :- for / while