* Polymorphism :→ Latin Words

(Poly) (Morph)
↓ many ↓ forms / shapes

Example :→

Entity (Rahul)

Location		Role
Classroom	→	Student
Restaurant	→	Customer
Home	→	Son
TAP cell	→	Coordinator

Entity doesn't change, only the behaviour changes.

The process by which the same entity (variable, method) can behave differently under different scenarios, is called "Polymorphism".

operators ✓ methods ✓

(I) Static	Compile Time	Overloading	Same Class
(II) Dynamic	Run Time	Overriding	Multiple Classes

Overriding → methods

Overloading → (I) Changing the no. of parameters.
(II) Changing the return type of the parameters.

Function Overriding or Method Overriding

Parent → shop (Groceries)

Son — Tutorial Center | virtual override | Daughter — Boutique

Employee → general

work → email, meetings, drafts, etc

Software Developer
Develop Apps, Test, Deploy
work ( ) → specific

Lecturer
Seminars, Lectures, etc
specific ← work ( )

* What is the most important use of Polymorphism?

Ans :→ The most important use of Polymorphism is that we can use a parent class reference or pointer to access a child class object.

Abstraction :→

Hiding → How it is done! → Implementation Details

Showing → What is being done! → Functionality

* Overwhelming the End User with extra info. (Not good for CSAT)

* User Experience → Easy & Smooth

For Abstraction in C++ we use :→

pure virtual functions. (It behaves like interface in Java)

virtual void function() = 0;
↳ override

All 4 Pillars :→ Encapsulation | Inheritance | Polymorphism | Abstraction

Association

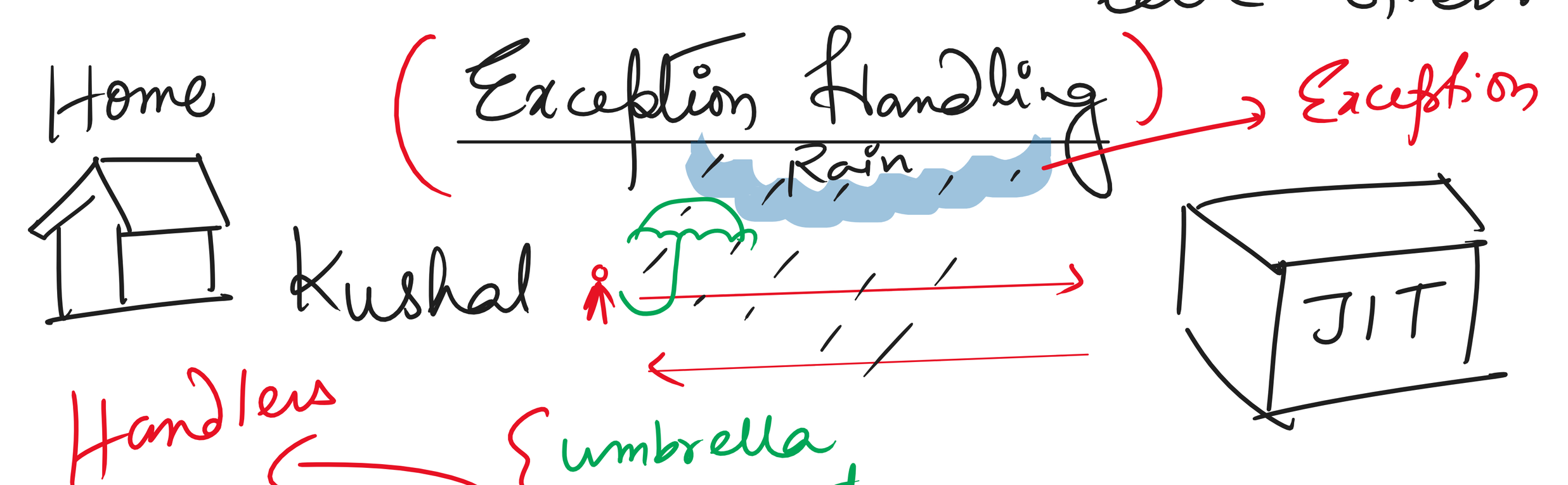
It is the relation b/w classes & interfaces inside an application.

IS – A → Inheritance

HAS-A
Car | MP3 Player
Loose Coupling
Aggregation
When the objects of two classes are independent of each other.

Car | Engine
Tight Coupling
Composition
When the objects of two classes are dependent on each other.

(Exception Handling) → Exception



Handlers ← umbrella, rain coat, cab

Anything that disrupts the normal flow of execution of code is called an exception.
The process of handling exceptions is called Exception Handling.

Types :→ (I) checked / known exceptions
(II) unchecked / unknown exceptions
(III) User-defined exceptions / custom exceptions

destructor

C++	Java	Python
try { } catch ( ) { }	try { } catch ( ) { } finally { }	try: except: finally:

Important keywords :→

Object (Java)
↓
Throwable
→ Errors
↳ Compile time → Syntax
↳ Runtime → logic
→ Exceptions
↳ checked
↳ unchecked
↳ UDE.

try, catch, [except], throw, throws, finally
[raise] python
finally: P, J