Object Oriented Programming

"खुल जाएंगे सभी रास्ते, तू रुकावटों से लड़ तो सही,
सब होगा हासिल, तू अपनी ज़िद पर अड़ तो सही।"

- What is OOPs
- Classes & Objects
- Access Specifiers
- Constructor & this pointer
- Encapsulation
- Polymorphism
- Inheritance & its types
- Abstraction & Inheritance
- Static keyword
- Packages
- Final Keyword
- Practice Problems on OOPS

C → procedural

10th → Experiment

(i) Aim

(ii) Apparatus

(iii) Procedure

add()
{
}

sub()
{
}

main()
{
int x
}

add()
{
}

main()
{
}

Car

Car

Accelerator
Break
Brand

```
void add ()
{
    c = 7+8;
    printf (c);
}

void fun ()
{
    add();
}
```

✓ Compile
✓ Run

Error

- ✓ Compilation
- Error

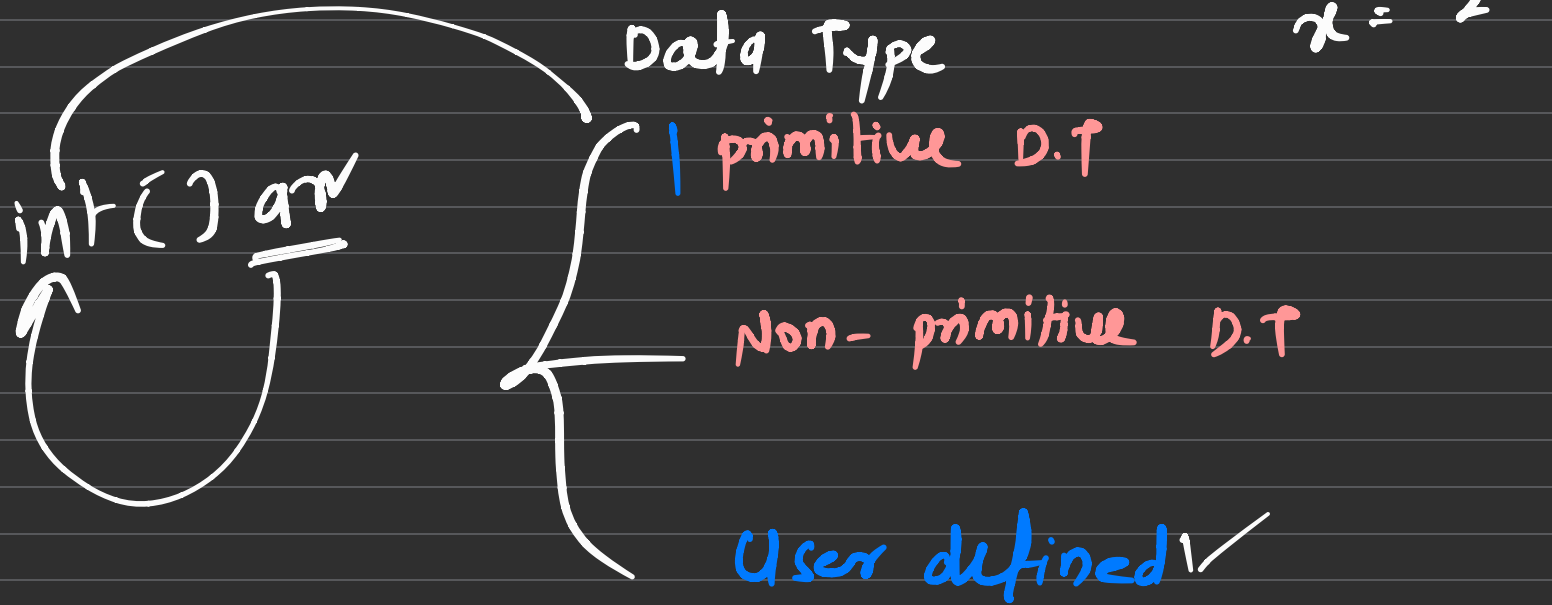
Linker

What is OOPs

OOPs is a programming technique which mainly revolve around real life objects

int x = 10

x = 'z'



Classes & Objects

Class

It is user defined data type, which contains data members (variables) & methods (functions).

syntax:

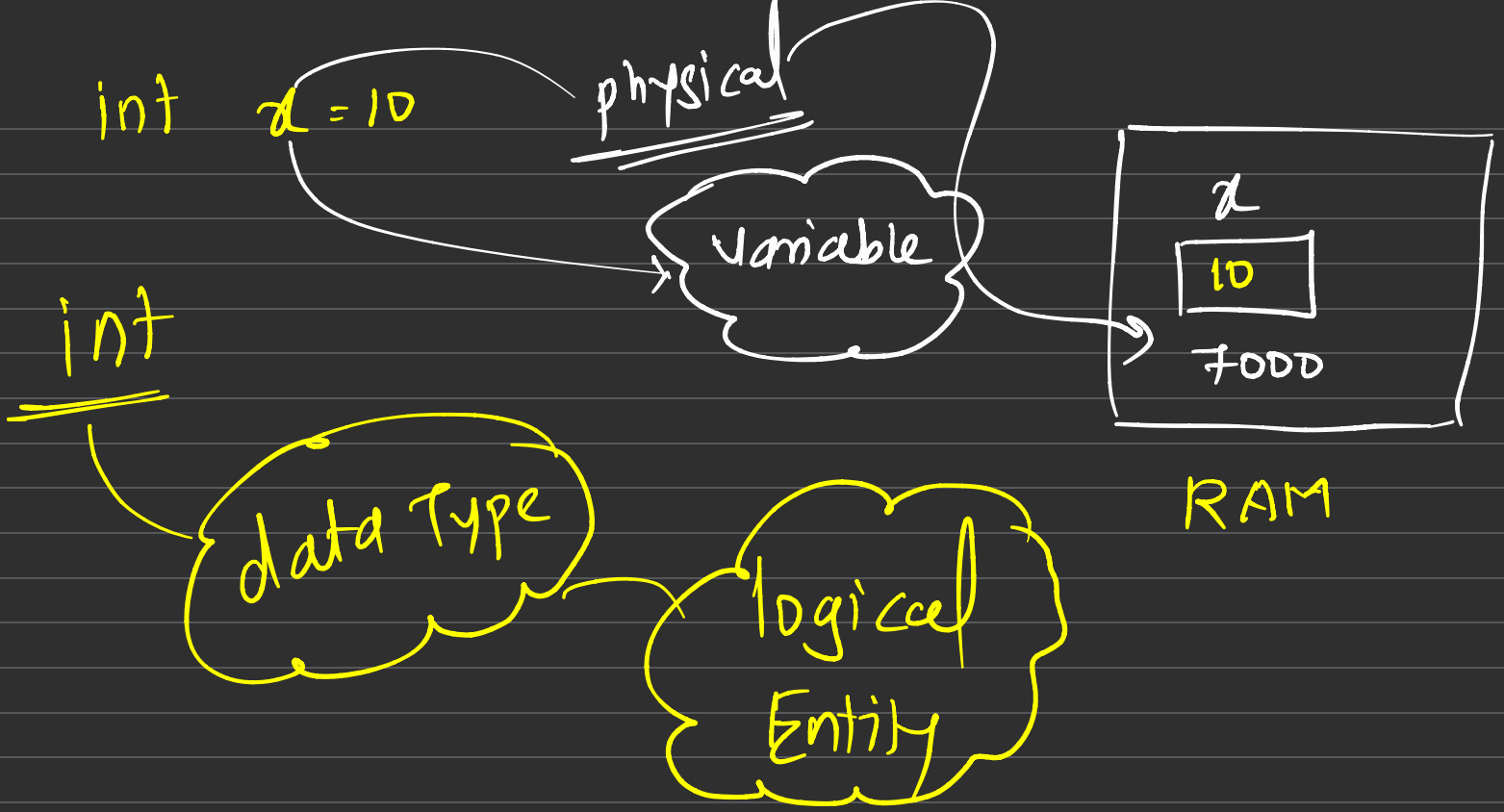
```
class ClassName
{
    // data member

    // method

}
```

```
class Person
{
    int age
    string nm

    void show()
    {
        println(age)
        println(nm)
    }
}
```

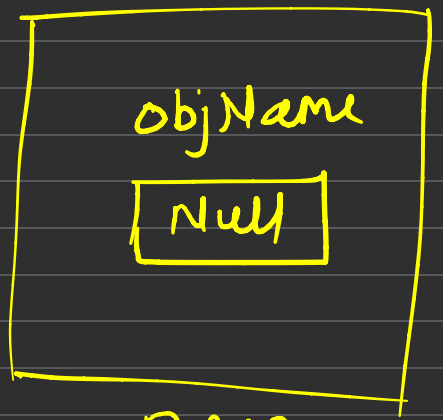


Object

It is a instance of the class – which tells what kind of operation we can perform on the class.

Syntax:

```
className objName;
objName = new className()
```




```
className obj = new className()
```

↳ dynamically

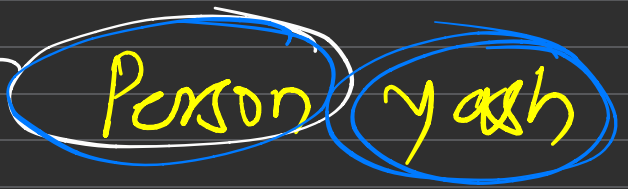
```java
class Person{  no usages
    // data members
    int age;  1 usage
    String name;  1 usage

    // member function
    public void info(){  no usages
        System.out.println("Age of the person is: " + age);
        System.out.println("Name of the person is: " + name);
    }
}
```

data Type

Person yash = new Person();

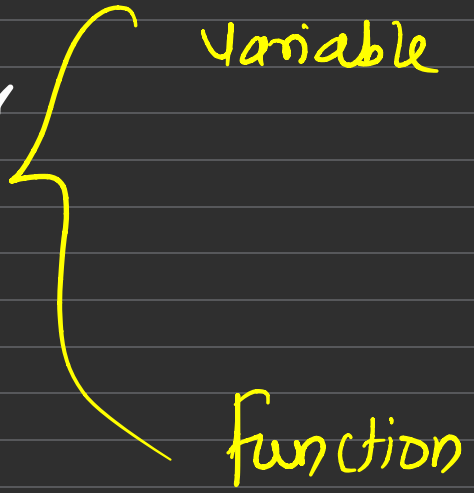
data Type



How to access members of the class

Using dot (.) operator

obj . member



Access Specifiers

Mainly decides the scope of accessibility of members of the class

- ✓ default
- public
- private
- Protected
  - inheritance (X)

* default : Can be access inside of class & outside of class but in same package

* private : Only inside the class

* public : can be accessed anywhere inside / outside package or class

Constructor & this pointer

Constructor

Mainly used to initialise the object of the class.

- name of constructor = name of class
- alway public
- called automatically
- No return type → not even void
- Diff. betn function vs Constructor

Types of Constructor

- Default Constructor
- Parameterized Constructor

this pointer

Mainly used to instialise the current object of the class.

Encapsulation

Binding of data member and member function together in the same class is called as Encapsulation.

```
class Person
{
    int age
    String name

    public void info()
    {
        println (age)
        println (name)
    }
```

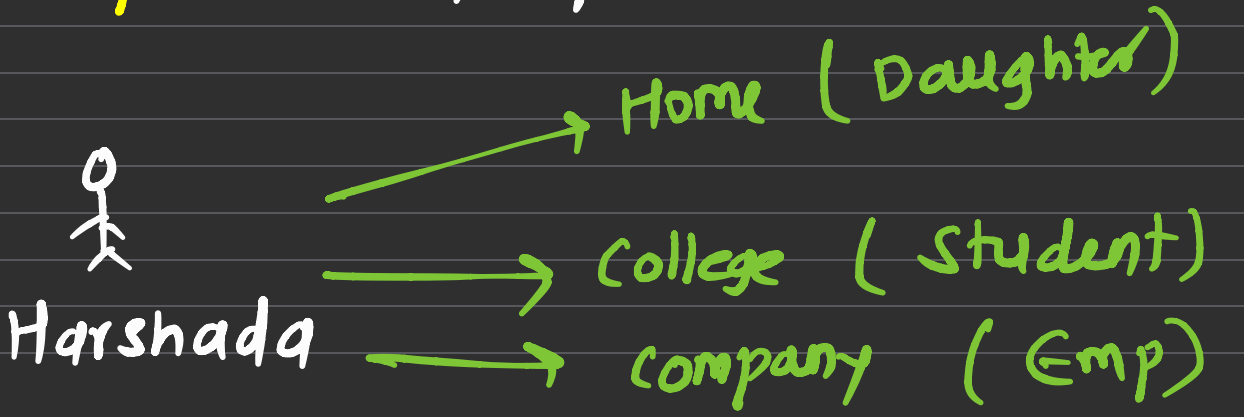
Encapsulation: Real life examples

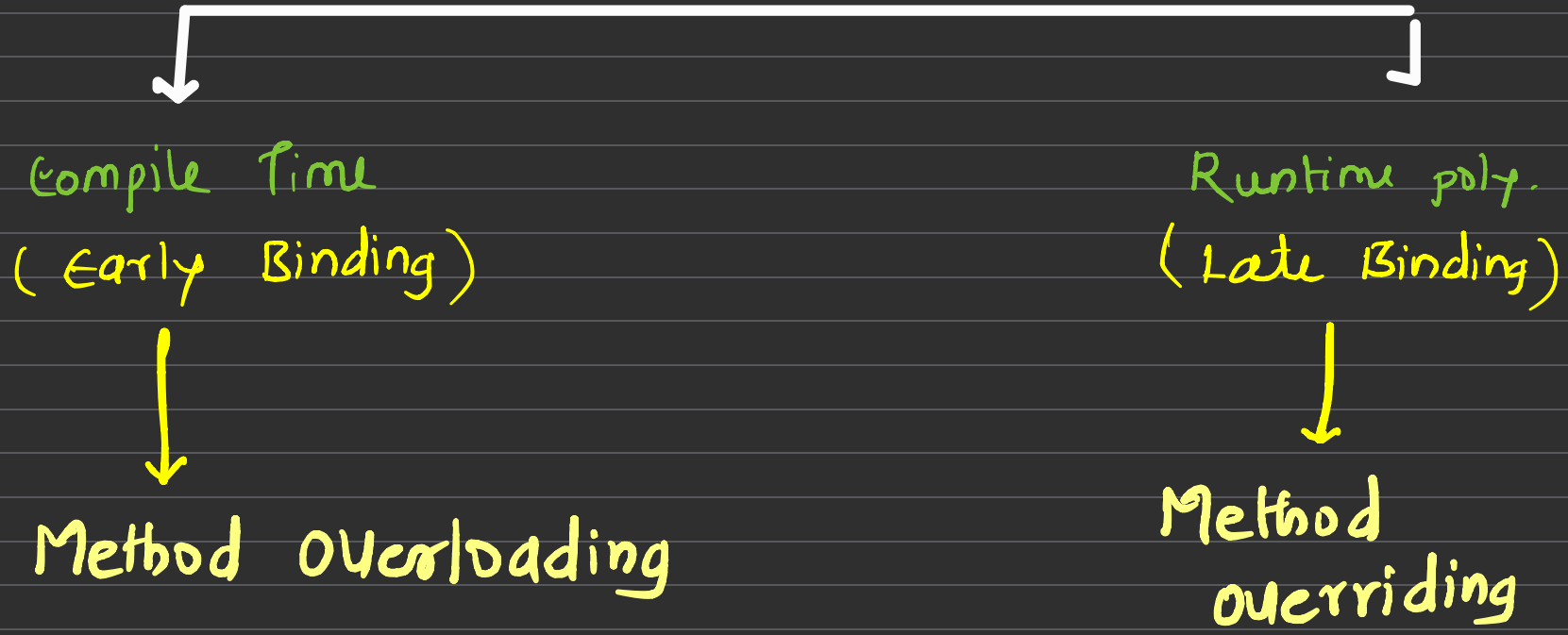
An organization consists of several departments like the production department, purchase department, sales department, and Accounts department. It combines all these departments together and formed the organization.

Polymorphism

When single entity acting differently in different scenarios is called as Polymorphism.

Greek { poly → many
morphism → forms





* Method Overloading

Two or more methods having same name but different parameters

- All method must be in same class
- Return type does not matter

* Method Overriding

- Redefining the meaning of method from parent class to child class

- function should have
  - Same name
  - Same parameter
  - Same return type
- Should be in diff class