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 → procedurral

10th → Experiment

1. Aim

11. Apparatus

111. 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

✓ Compilat

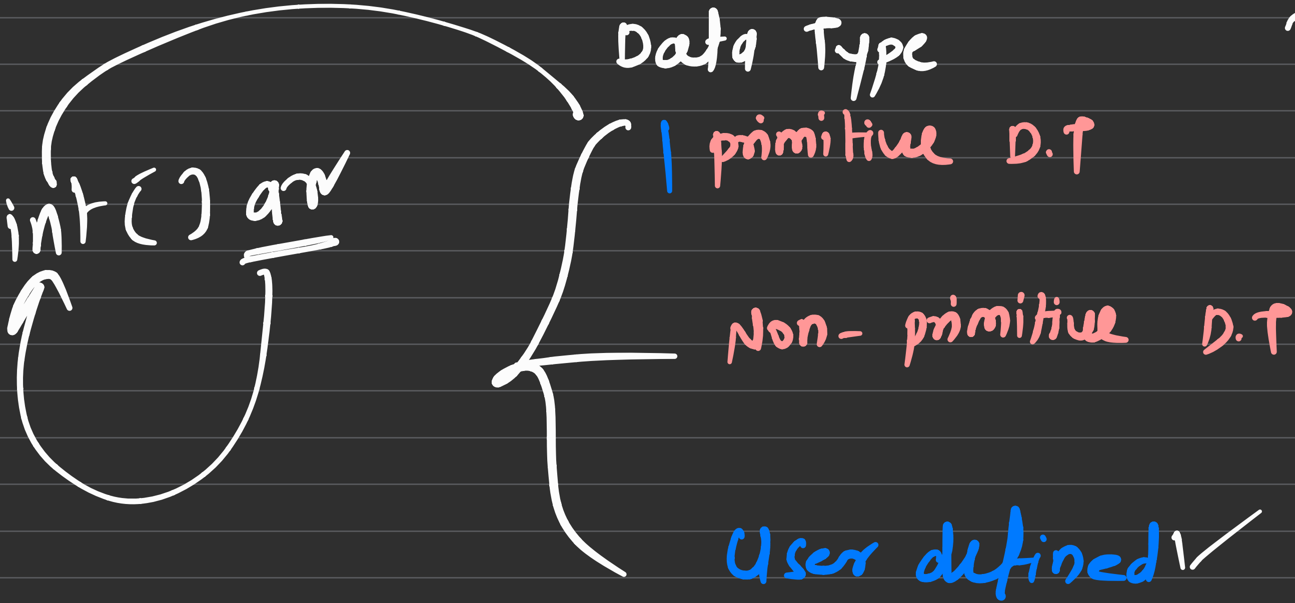
Error

linker

What is OOPs

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

int x=10
x = 'z'



int[] arr

Data Type

- primitive D.T
- Non- primitive D.T
- User defined ✓

Classes & Objects

Class

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

syntax:

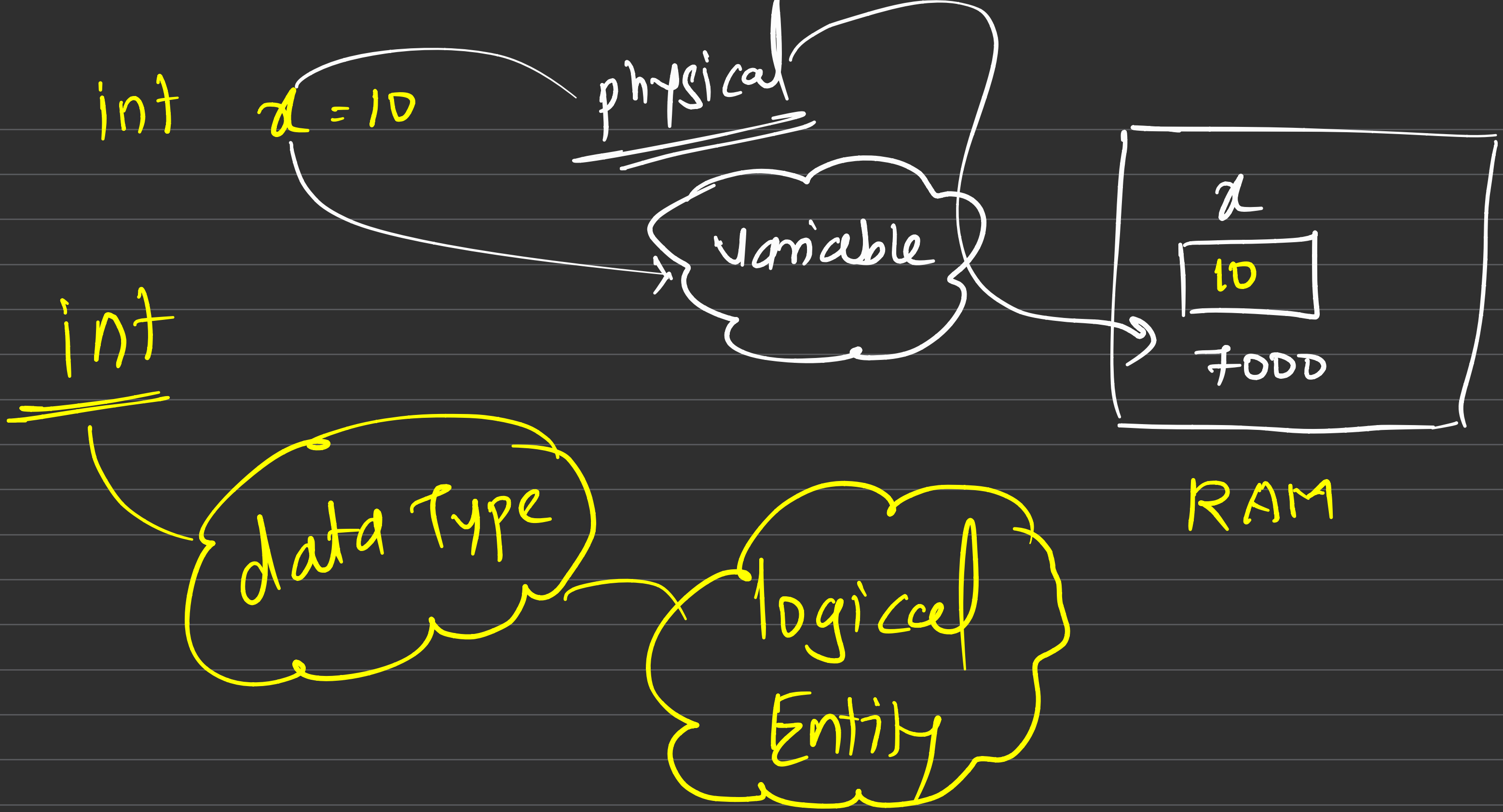
```
class ClassName
{
        // data member


        // method

}
```

```
class Person
{
    int age
    string nm

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


int x = 10
physical
variable
x
10
7000
RAM
int
data Type
logical
Entity

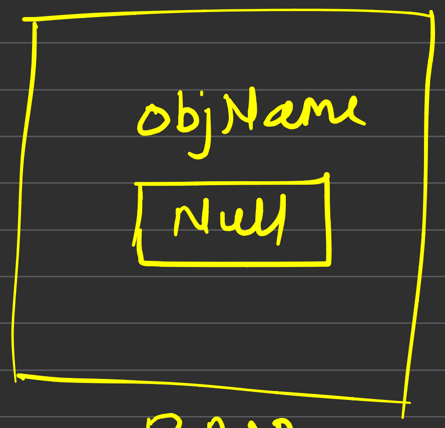
Object

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 ( )

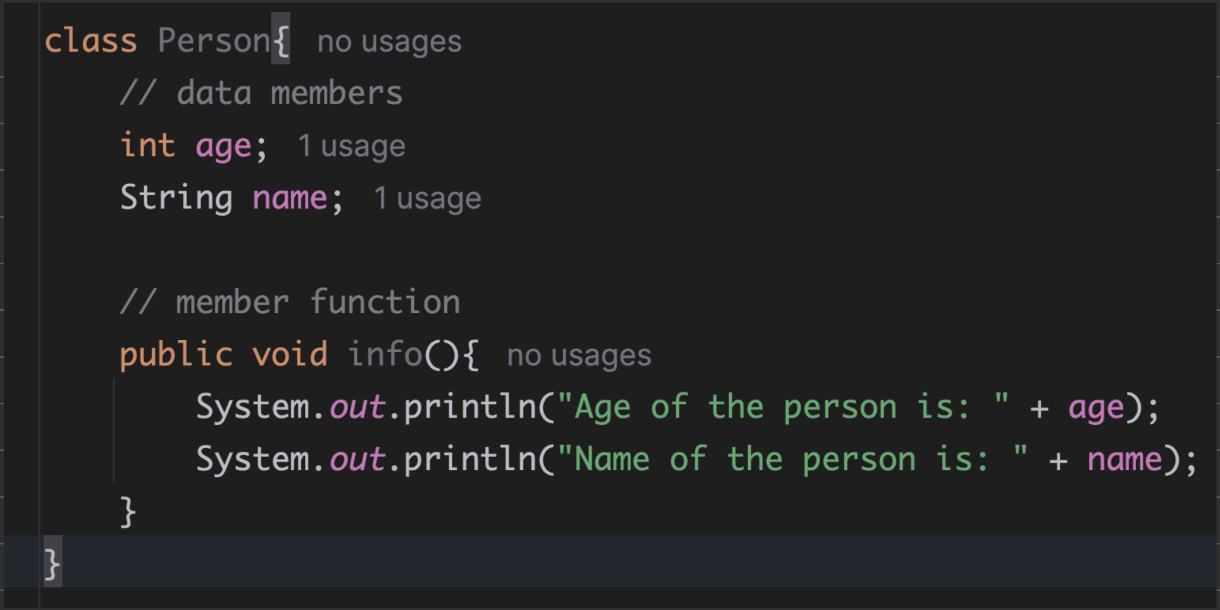


RAM

(Heap)

className obj = new className( )

dynamically



```
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();

int x

data Type

Person yash

How to access members of the class

Using dot (.) operator

$$\text{obj . member} \begin{cases} \text{Variable} \\ \text{Function} \end{cases}$$

Access Specifiers

Mainly decides the scope of accessibility of members of the class

✓ default   private

public   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