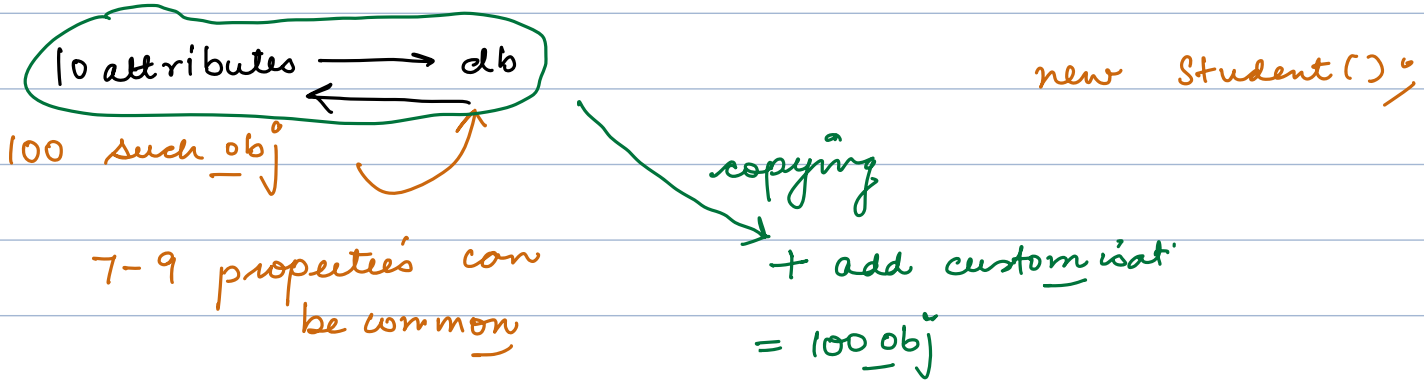
Singleton
Builder

Agenda : Prototype & Registry.

- given with an object of a class.



pubg ≡ 100 players.

55 + 45 bots.

↓ location + gun.

```
Student st = new Student();
Student st2 = st;
```

X no new obj, copying the ref.

client {

Student st = new Student();

=

=

① not reusable ← Student stI = new student();

② private attributes

↓

getters & setters.

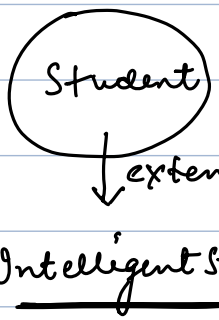
stI.id = st.id

stI.name = st.name

⋮

}

③ client will need to know the complete details of the class.



Student Intelligent

void do(Student st) {

type

~~Student st = new Student(st);~~

if(st.type == "Student")

else if(st.type == "IntSt")

=

}

Outsourcing the copying to the object itself.

copy().

```
Student st = new Student();
Student st2 = st.copy();
```

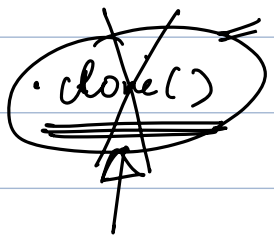
```
Student {
    copy() {
        Student x = new Student();
        x.id = this.id;
        x.name = this.name;
        :
    }
}
```

↓

```
IntelligentStudent {
    copy() {
        new IntelStr
    }
}
```

Override

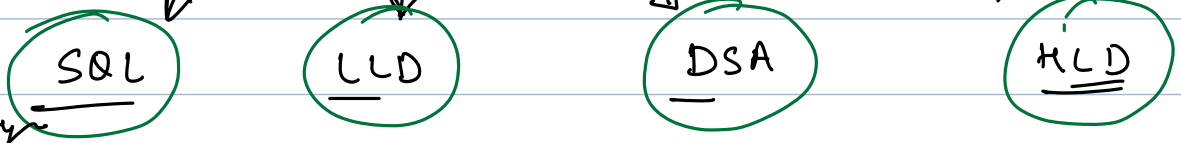
```
do(Student st) {
    st.copy();
}
```



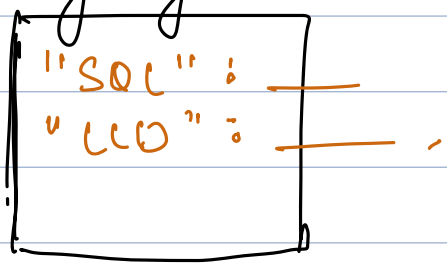
Lecture

- assignment
- homework
- pre-read
- skill
- Instructor
- Notes

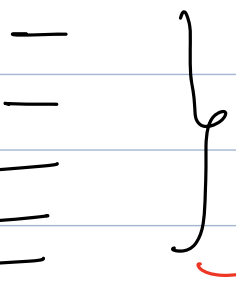
Same attributes but diff values.



Registry



Monster



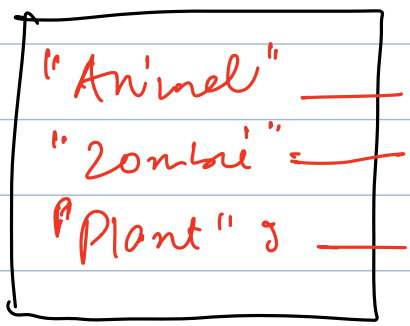
Animal

Zombie

Plant

you can have all diff types of monster by just adding diff values to attrib,

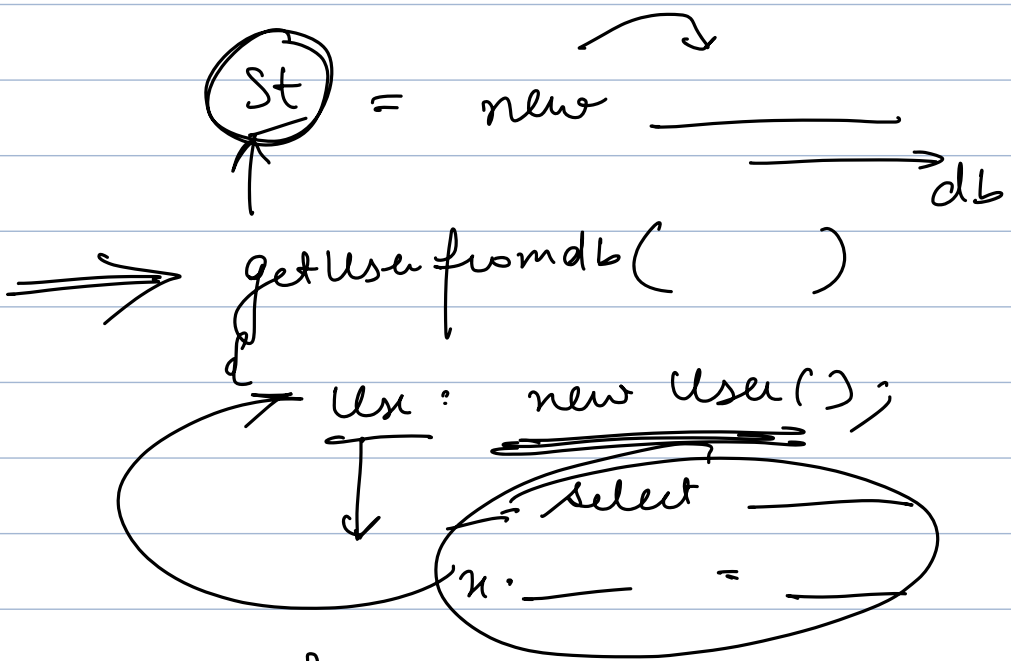
inheritance doesn't add anything



Prototype provides the flexibility of creating a new object by copying from existing one

+

storing multiple prototype to copy later on is registry.



getUserfromdb()
getUserfromdb()
getUserfromdb()
getUserfromdb()
getUserfromdb()

x = getUserfromdb()

x1 = x.copy()
x.copy()
x.copy()
x.copy()

x1 both = "Apar"

Break: 10:16 pm