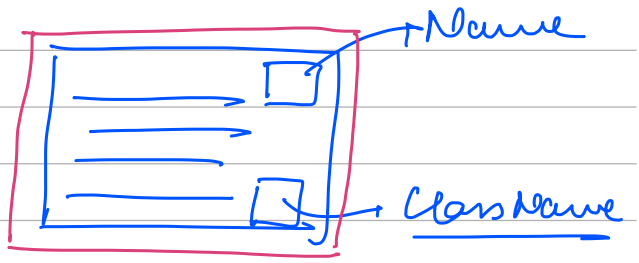
Agenda

Start @ 9.05 pm

1) Prototype design Pattern

↳ Model / sample



Problem statement

Given an object of a class, we need to create a copy of this object.

(Create a new object with exact same attributes)

eg

```
Client {
    psvm () {

        Student st = new Student (→);

        Student stcopy = new Student ();
            stcopy.name = st.name;
            stcopy.age = st.age;
```

Cons

① Client needs to know all the implementation details of student class.

② Some fields in the student class can be private & hence can't be copied

Student St = ; → Student / Intelligent Student

Student
name
age

IS
IQ

SRP
OCP

if (st instance of student)
{
St copy = new Student()
{ attrs copy }
}
elseif (st instance of IS)
{
Stcopy = new IS();
{ attrs copy }
}
elseif
{
}

Copy Constructor

Student (Student st)
{
this.name = st.name;
this.age = st.age
{ Remaining attrs }
}

Student
↑
IS

Intelligent student will also have similar Constructor

Client

Student st = (——); → Student / IS / AlumniStudent

OCP

```
Student stcopy =
if (st instanceof Student)
{
    stcopy = new Student(st);
}
else if (st instanceof IS) {
    stcopy = new IS();
}
```

decision

⇒ If client wants to create a copy of an object having the logic of creating object within client is prone to errors || issues.

Ideal soln: Client should outsource the work of creating a copy object to class itself.

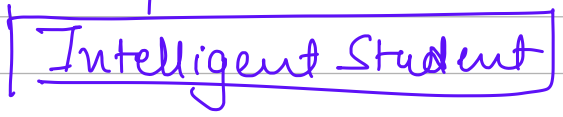
Client

```
Student st = (——);
Student stcopy = st.copy();
```

Benefits
1. Less code on client side
2. Client need not to know the impl of object class.
3. No voilation OCP

Student

```
Student copy() {
    Student s = new Student();
    s.name = this.name;
    s.age = this.age
    ____________
    ____________
    ____________ } attrs
    return s;
}
```




```
IntelligentSt copy() {
    IS is = new IS() {
        is.name = this.name();
        is.age = this.age();
        ____________
        ____________
        ____________
    return is;
}
```

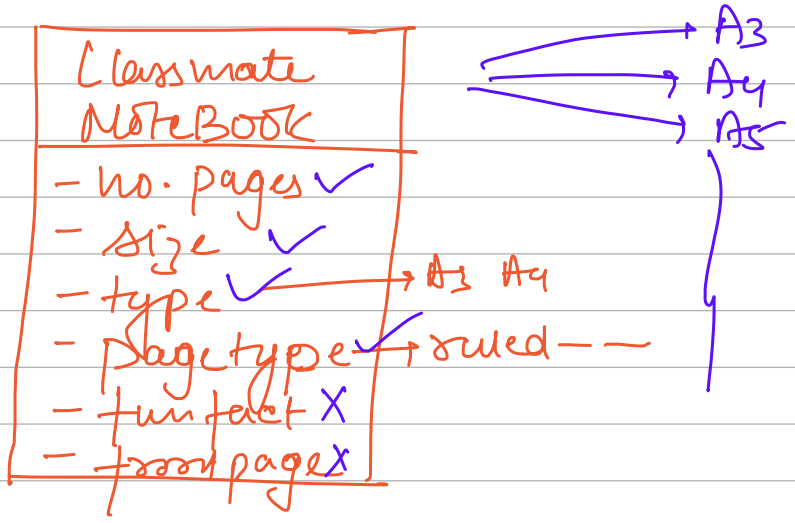
We will have to make sure that all child classes have overridden copy()

```
Student st = new Student();   // IS()
Student stcopy = st.copy();
```

} Runtime polymorphism

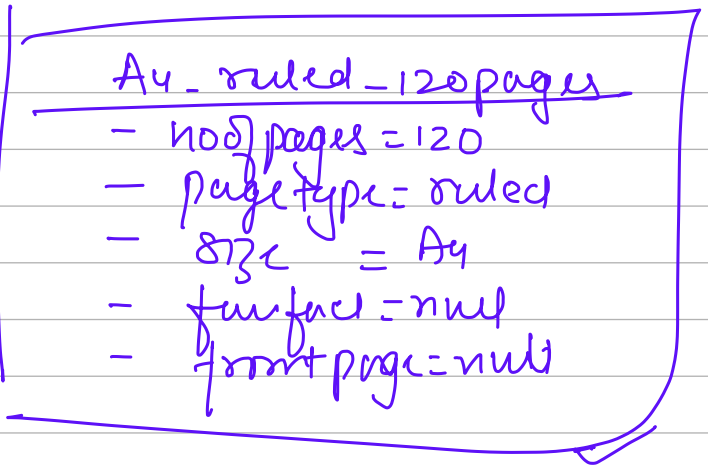
eg Classmate Notebooks

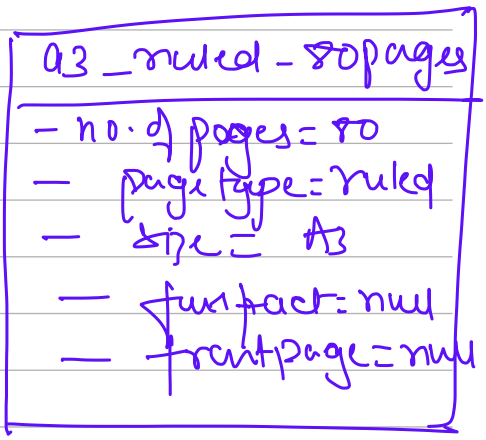
1) Classmate wants to create 1 Million notebooks of A4 size Ruled with 120 pages.



2) Let's say we create a prototype object of Classmate Notebook.

3) All the objects can be copied from the prototype & just give values of frontpage and funfact.





ClassMate NB nb = getPrototype(A4_ruled_120pages).copy();

nb.funfact = —

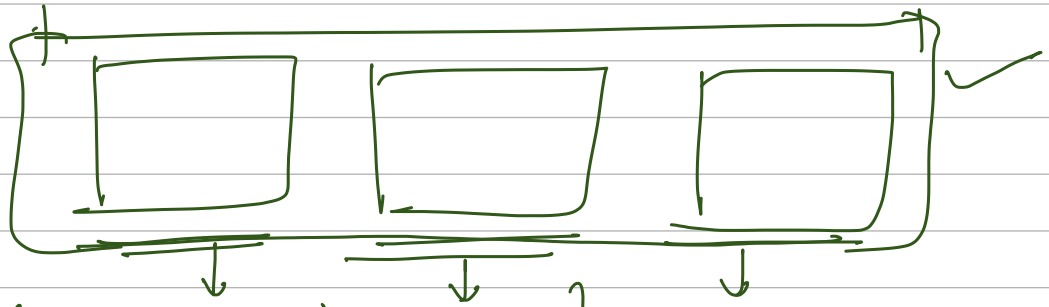
nb.frontpage = —

Prototype Pattern

Often there are scenarios where we don't have to create an object from scratch. Rather we prefer creating an object from a prototype and then setting only a required values.

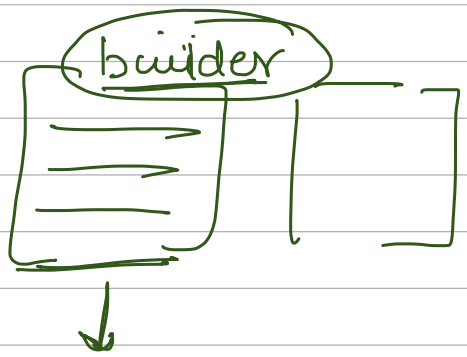
Custom

Doubts



① copy(——, ——)
copy(——, ——, ——) } explosion

② copy()



Class student {
- name
- age
- Batch
- avgBatchPsp

} Common

}

Registry

Create a prototype object & Store it somewhere

```
Student july22Batch = new Student();
july22Batch.batch = "July 22 Batch";
july22Batch.avgPsp = "80";
```

```
Student july22Batch = Student.getBuilder()
    .setBatch("july22")
    .setAvgPsp(80)
    .build();
```

Step 1 In the class for which we want to create a prototype, declare a method called CLONE() to create copy(). Note: All child classes needs to override the clone().

Step 2 Store the prototype object in registry

Student Registry

```
HashMap<String, Student> map;

get(Key) {
    return map.get(Key);
}

register(String key, Student s) {
    map.put(key, s);
}
```

(III) Client calls registry to get the prototype object

(IV) Create a copy from prototype and set custom values