Builder design pattern .

creation of complex object .

① class has a lot of attributes

```
class SoldItem {
        item
        price
        quantity
        prime
        discount
        extraDelivery
        paymetMethod
          :
          :
}
```

10 parameter
↓
2^{10} constructor

constructors

(__ , __)

(__ , __ , __)

(__) , (__) , (_ , _ , _ ,)

Setters

```
Student st = new Student();
st.setAge();
st.set__();
st.set___();
```

Student

invalid

<20

```
new Student(_, _, _, _);
```

HashMap

```
HashMap<String, Object> hm;

hm.put("name", "Mohit");
hm.put("age", 28);
⋮
```

~~"aeg"~~

"shama"

① name of the keys are not checked

② type of the values are not checked

```
class Helper {                                  object of class

    String   name
    int      age
    String   batch

}



class   Builder {

        // we can keep all
           attributes of original
           class.

 }
↑

{   Builder b = new Builder();
        b.setAge()
        b.setName()
          ⋮
  Student s = new Student(b);
```

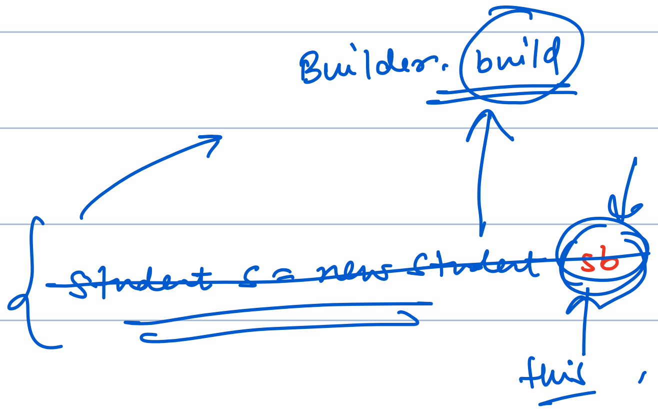
Student st = Student.getBuilder() → returns a new obj of StudentBuilder

.setAttr1() → Every setter returns the builder.

.setAttr2()

.setAttr6()

.build()


Builder.build
~~Student s = new Student~~(sb)
this