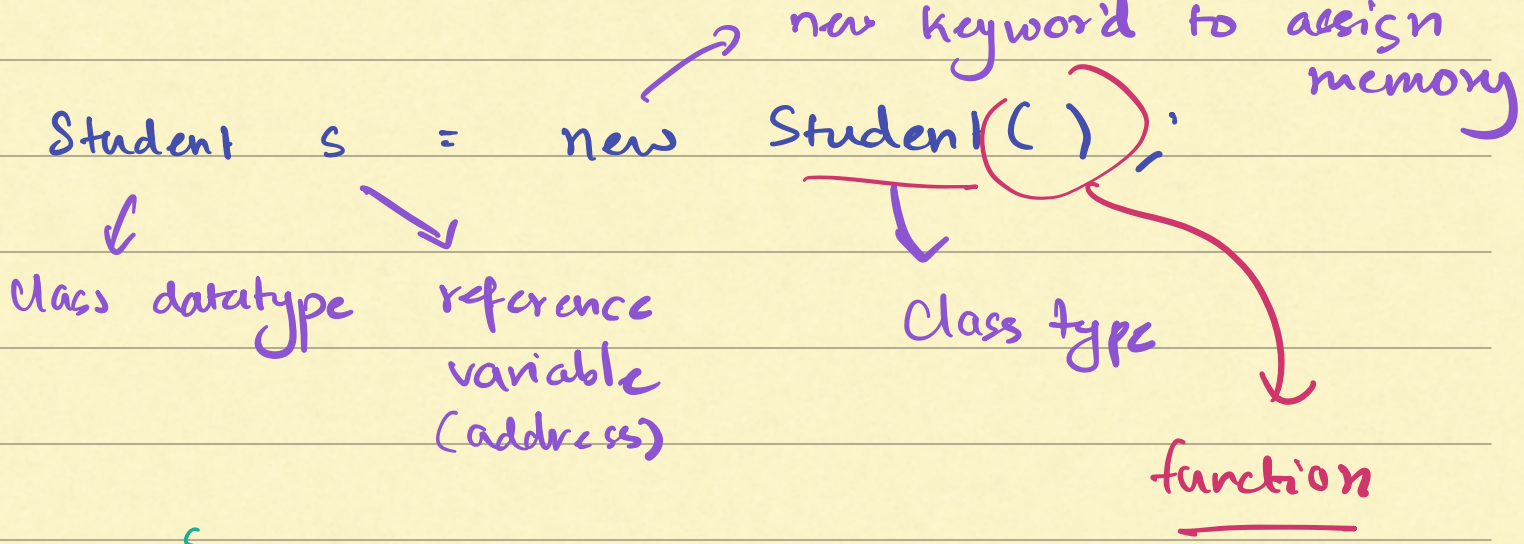
Agenda :

- Constructors
- Copy Constructors
- Deep copy vs Shallow copy
- Static

Constructors :

Class - Blueprint

Object - Instantiate a class.



s.

Constructor is a function which instantiates and initialises an object.

1. Default Constructor.

```
public Student() {
    name = "";
    age = 0;
    psp = 0.0;
}
```

```
student {
String name;
int age
float psp;
```

2. Manual constructors
   - Unparameterised
   - Parameterised.

→ no parameters.

```
public Student ( ) }

    name = "Akash";
    age = 29,
    psp = 5.0;

}
```

↳ public Student(String name, int age){

this.name = name;

this.age = age;

this.psp = 0.0

}

s.print(); this

1. Default constructors are only provided if no manual constructors are written

2. Custom Constructors does :

a. initialises the variables with default values at the start.

b. The update the variables with values provided by parameters / user.

Copy Constructors

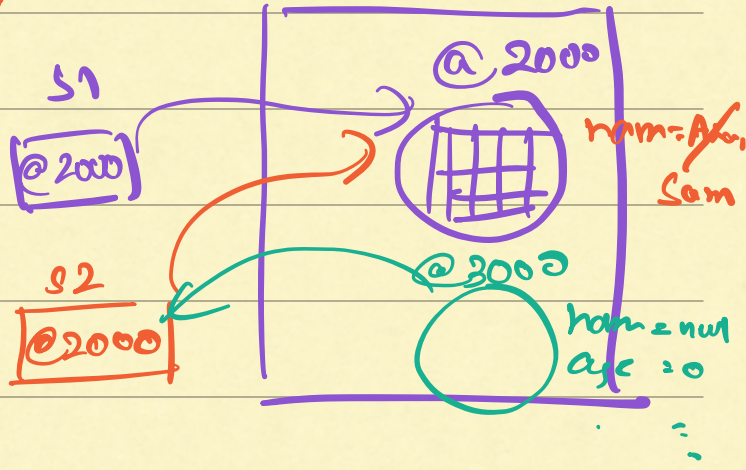
```
Student {

    String name;
    int age
    float psp;

}
```

```
Student s1 = new Student();
s1.name = "Akash";
s1.age =
. . .
```

```
Student s2 = s1;
s2.name = "Sam";
```



Way1 of copy:

```
Student s2 = new Student();

s2.name = s1.name;
s2.age = s1.age;
s2.psp = s1.psp;
```

Problems:

1. Too much manual code/ repetation
2. Private variables can't be accessed outside.

```
class Student {
    String name;
    int age
    float psp;
    ≡
    // copy constructor
    public Student(Student s) {
        this.name = s.name;
        this.psp = s.psp;
        this.age = s.age;
```

→ other constructors.

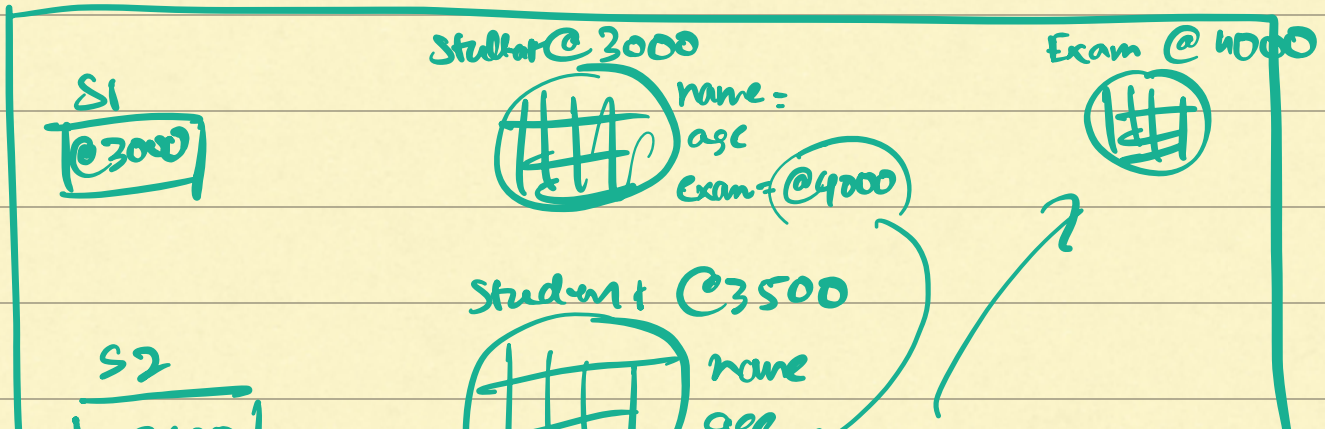
```
    }
}
```

```
Student s1 = new Student();
s1.name = "Akash";
s1.age = ...
Student s2 = new Student(s1);
```

Shallow copy

```
Student {
    String name;
    ...
    Exam exam;
}

Student s2 = new Student(s1);
```

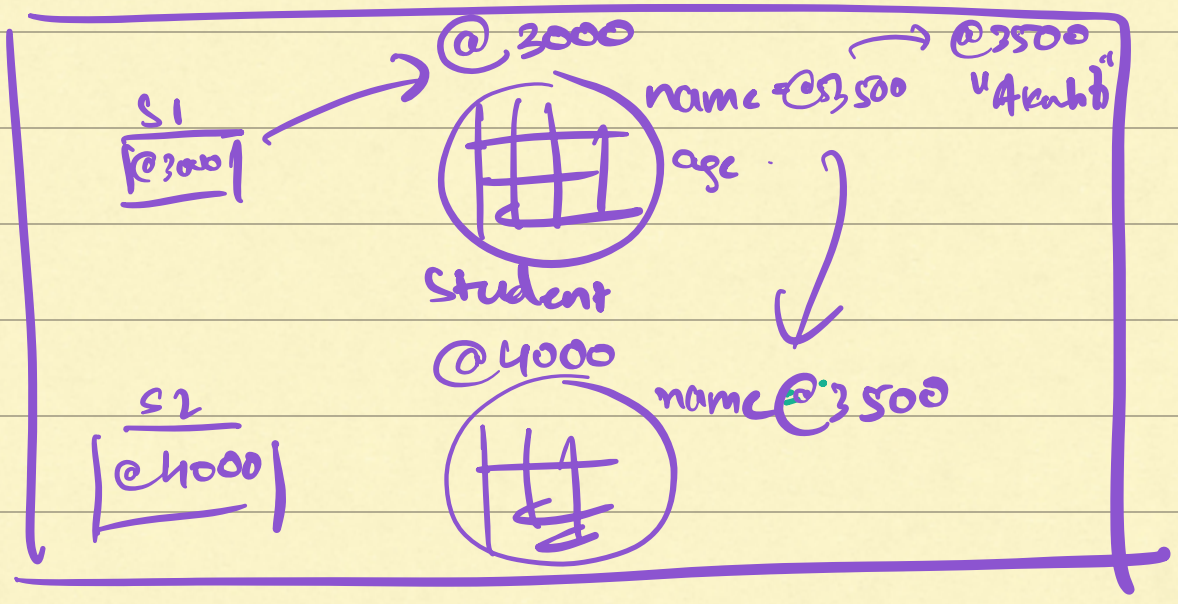


@3500

exam = @4000

this.exam = s.exam;

s2 = s1;



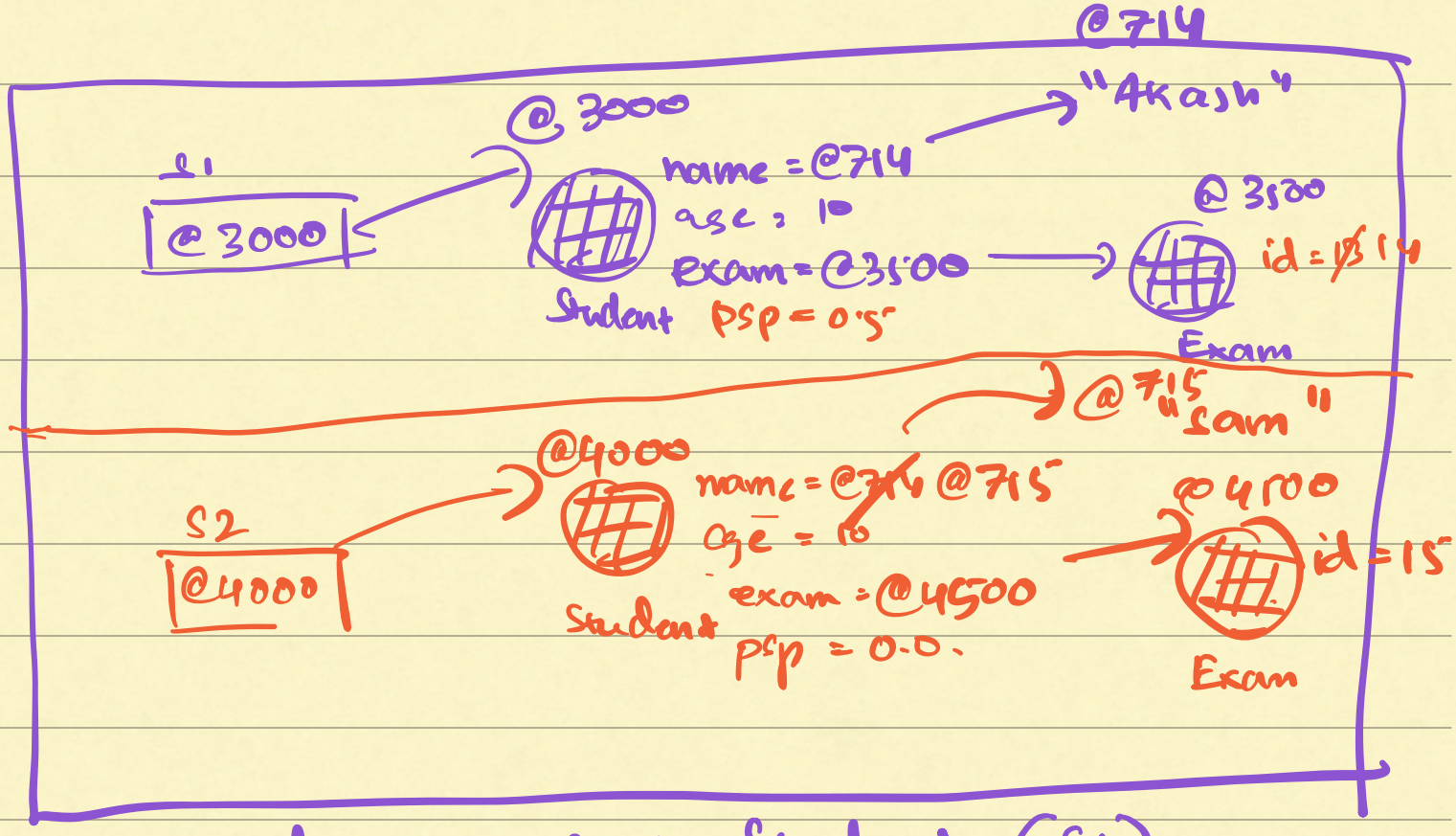
this.name = s.name;

this.name = "Akash"

this.name = "Sam";

↳ this.name = new String("Sam")

sam



Student s2 = new Student (s1);

this.name = s.name;

this.exam = s.exam; X

this.exam = new Exam (s.exam);

s2.exam.id = 15

s2.name = "Sam".

↳ s2.name = new String("Sam");

5 mins → 8:52 AM.

Pass by reference / Pass by value

```
void fun(int x) {
    x = x+10;
    sout(x) → 15
}

int x = 5;
fun(x);
sout(x); →
```

&

```
fun(Student s) {
    s.age += 10;
}
```

Student s = new Student();

s.age = 5;

fun(s);

sout(s.age) → 15

Static :