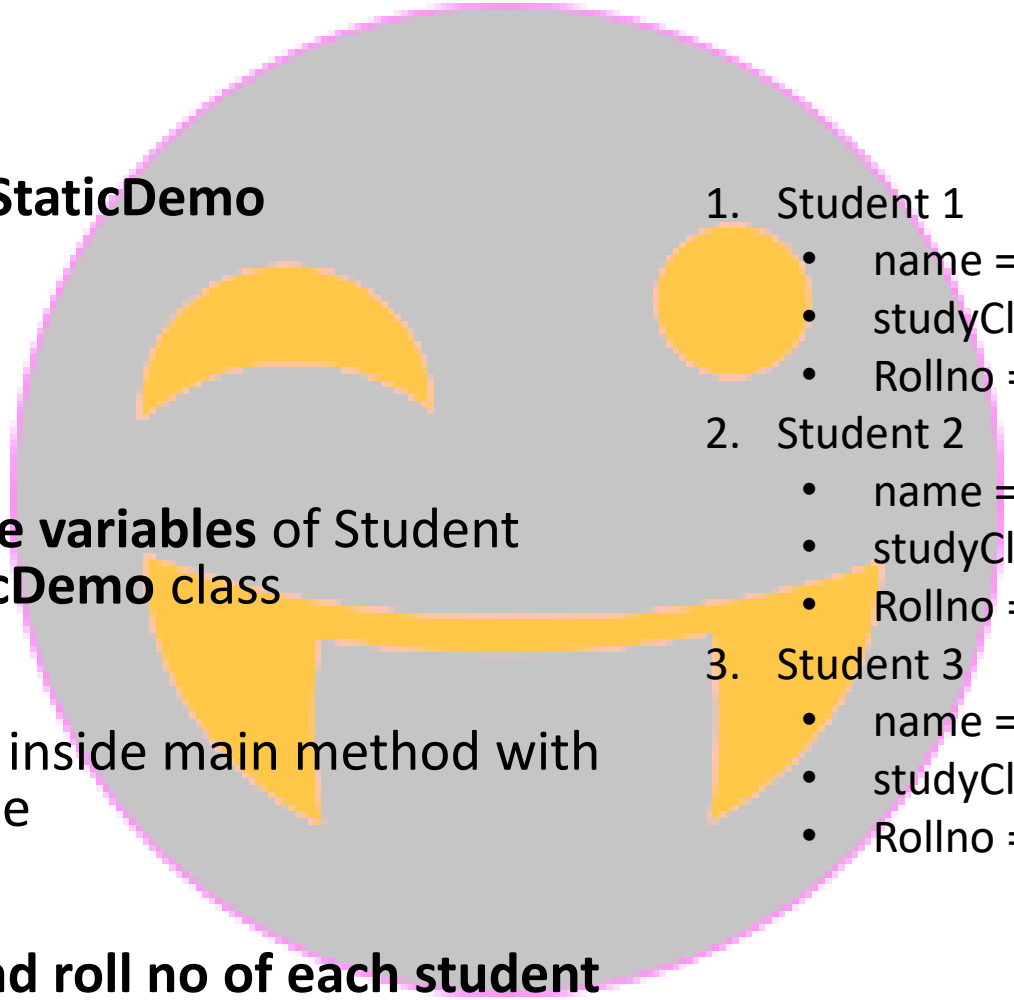
Chapter 17

static

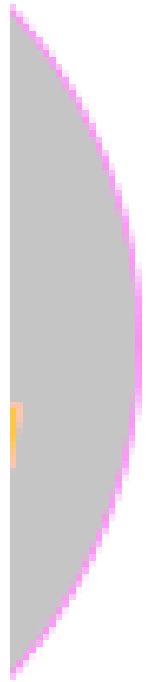
Task

- Create a class named **StaticDemo**
- Write **main method**
- Print values of **instance variables** of Student class 😄 😄 **from StaticDemo** class
- Create **three students** inside main method with details shown right side
- Display **name, class and roll no of each student just after setting the values**



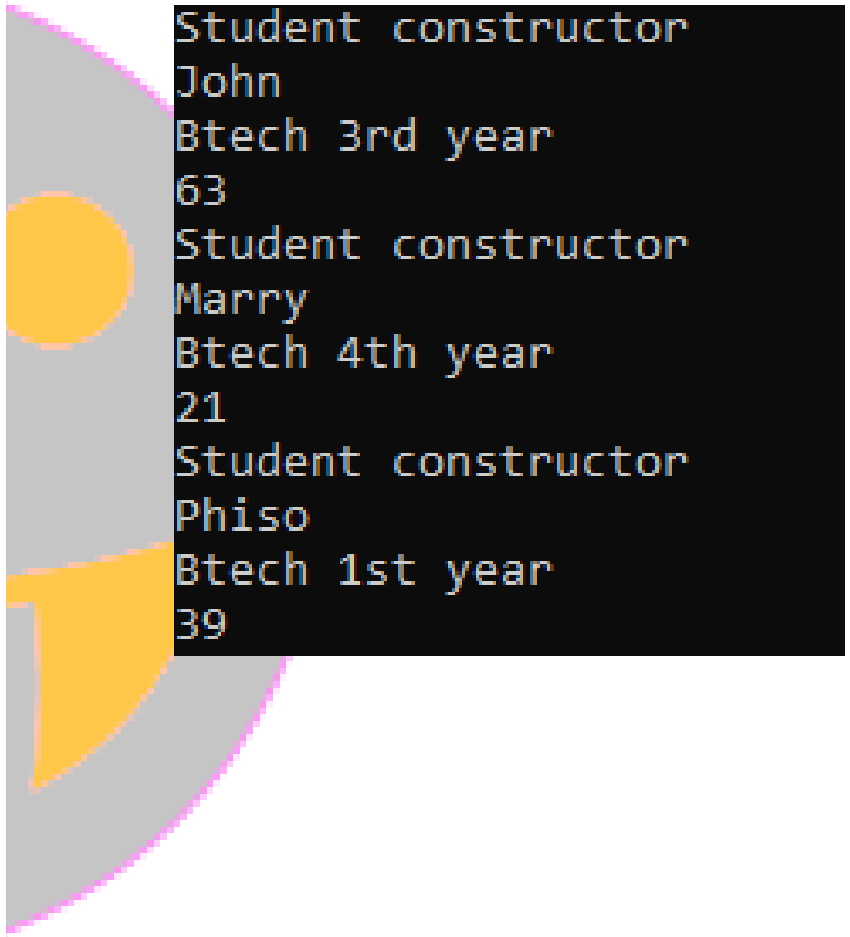
1. Student 1
   - name = John
   - studyClass = Btech 3rd year
   - Rollno = 63
2. Student 2
   - name = Marry
   - studyClass = Btech 4th year
   - Rollno = 21
3. Student 3
   - name = Phiso
   - studyClass = Btech 1st year
   - Rollno = 39

```
class StaticDemo{

    public static void main(String[] args){
        Student student1, student2, student3;
        student1 = new Student();
        student1.setName("John");
        student1.setStudyClass("Btech 3rd year");
        student1.setRollno(63);

        student2 = new Student();
        student2.setName("Marry");
        student2.setStudyClass("Btech 4th year");
        student2.setRollno(21);

        student3 = new Student();
        student3.setName("Phiso");
        student3.setStudyClass("Btech 1st year");
        student3.setRollno(39);
    }
}
```



```
Student constructor
Student constructor
Student constructor
```

```java
class StaticDemo{
    public static void main(String[] args){
        Student student1, student2, student3;
        student1 = new Student();
        student1.setName("John");
        student1.setStudyClass("Btech 3rd year");
        student1.setRollno(63);

        System.out.println(student1.getName());
        System.out.println(student1.getStudyClass());
        System.out.println(student1.getRollno());

        student2 = new Student();
        student2.setName("Marry");
        student2.setStudyClass("Btech 4th year");
        student2.setRollno(21);

        System.out.println(student2.getName());
        System.out.println(student2.getStudyClass());
        System.out.println(student2.getRollno());

        student3 = new Student();
        student3.setName("Phiso");
        student3.setStudyClass("Btech 1st year");
        student3.setRollno(39);

        System.out.println(student3.getName());
        System.out.println(student3.getStudyClass());
        System.out.println(student3.getRollno());
    }
}
```

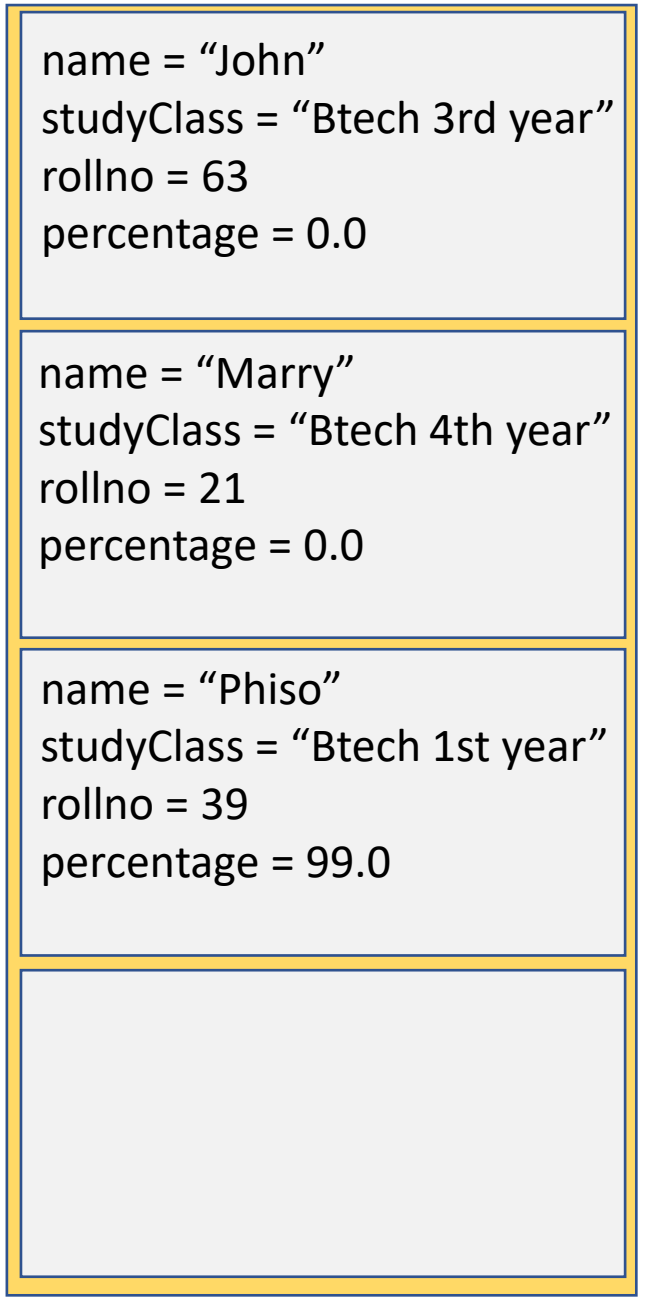


Let’s represent college name for the students

- **All** the students will **have the same college**
- **Remember name, study class, roll no may vary** but the **college name is same** for all the objects(students)
- Let’s create **a new instance variable** named **college** with getter and setter

```
class Student{
    String name;
    String studyClass;
    int rollno;
    double percentage;
    House h;
    String college;
```

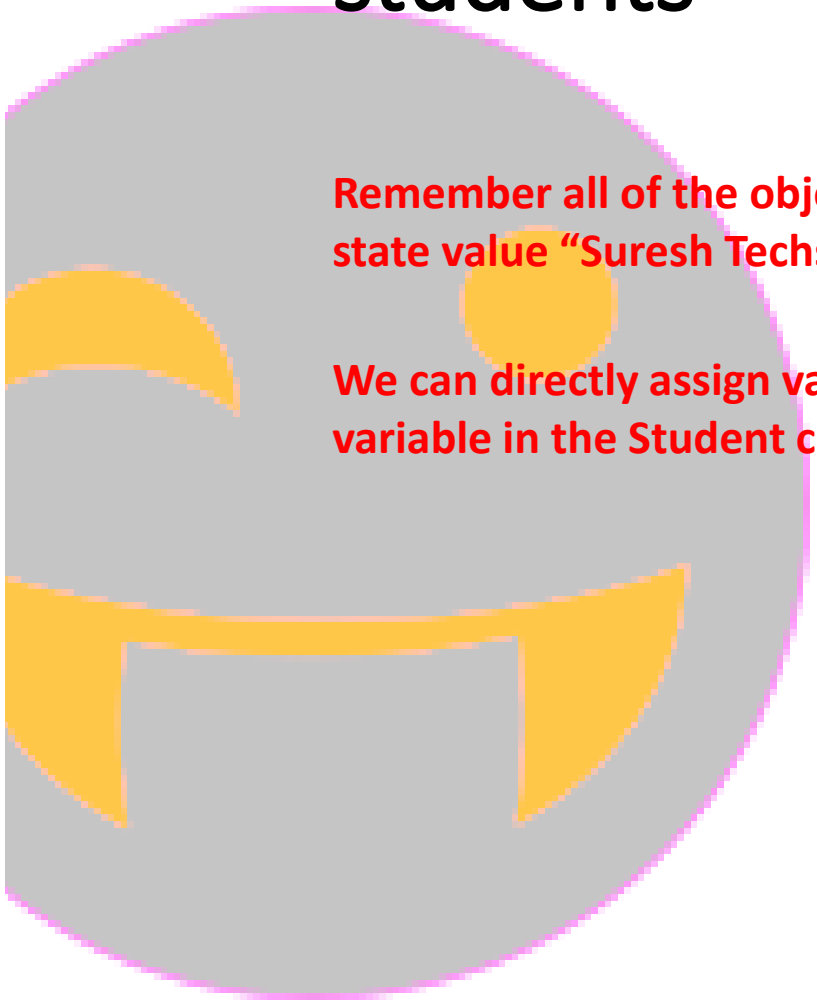
```
void setCollege(String college){
    this.college = college;
}

String getCollege(){
    return college;
}
```



Memory

```java
class StaticDemo{
    public static void main(String[] args){
        Student student1, student2, student3;
        student1 = new Student();
        student1.setName("John");
        student1.setStudyClass("Btech 3rd year");
        student1.setRollno(63);
        student1.setCollege("Suresh Techs College");

        System.out.println(student1.getName());
        System.out.println(student1.getStudyClass());
        System.out.println(student1.getRollno());
        System.out.println(student1.getCollege());

        student2 = new Student();
        student2.setName("Marry");
        student2.setStudyClass("Btech 4th year");
        student2.setRollno(21);
        student2.setCollege("Suresh Techs College");

        System.out.println(student2.getName());
        System.out.println(student2.getStudyClass());
        System.out.println(student2.getRollno());
        System.out.println(student2.getCollege());

        student3 = new Student();
        student3.setName("Phiso");
        student3.setStudyClass("Btech 1st year");
        student3.setRollno(39);
        student3.setCollege("Suresh Techs College");

        System.out.println(student3.getName());
        System.out.println(student3.getStudyClass());
        System.out.println(student3.getRollno());
        System.out.println(student3.getCollege());
    }
}
```

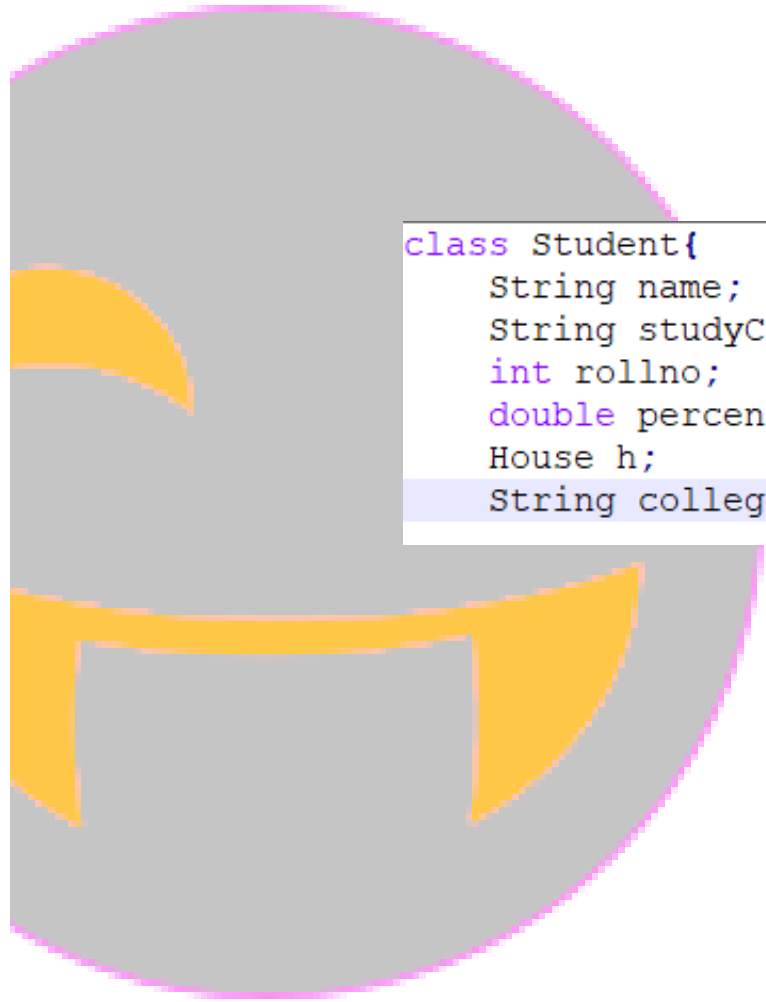
Set college name to three students



Remember all of the objects is having same state value “Suresh Techs College”

We can directly assign value to the instance variable in the Student class itself 😄 😄 😄

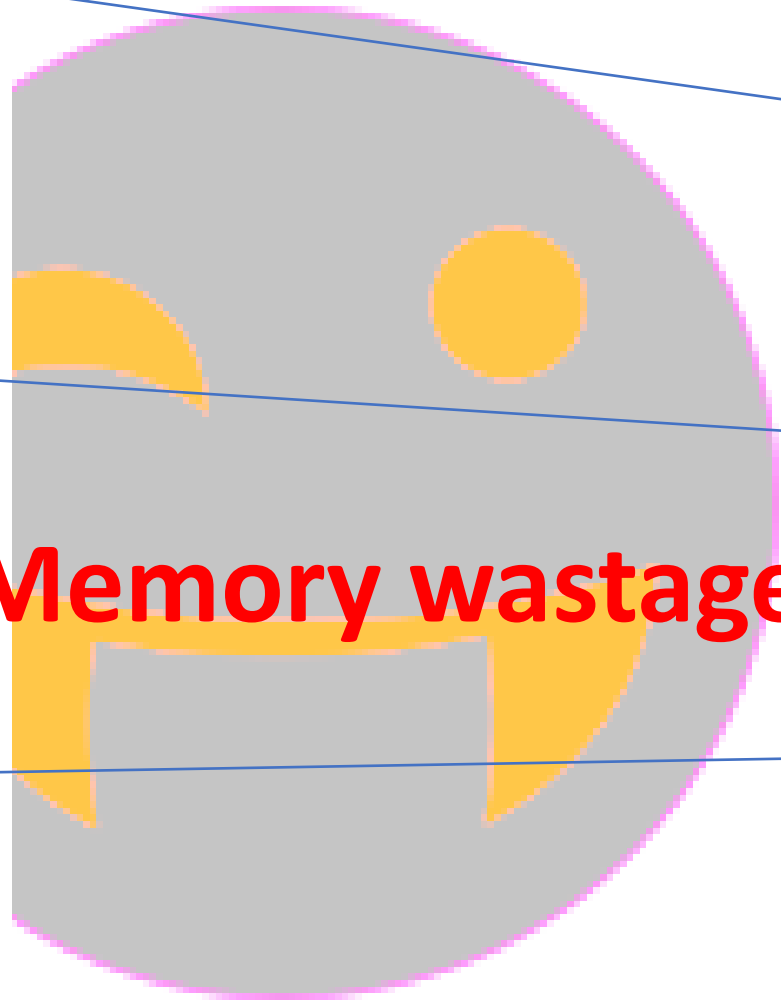
```java
class StaticDemo{
    public static void main(String[] args){
        Student student1, student2, student3;
        student1 = new Student();
        student1.setName("John");
        student1.setStudyClass("Btech 3rd year");
        student1.setRollno(63);
        //student1.setCollege("Suresh Techs College");

        System.out.println(student1.getName());
        System.out.println(student1.getStudyClass());
        System.out.println(student1.getRollno());
        System.out.println(student1.getCollege());

        student2 = new Student();
        student2.setName("Marry");
        student2.setStudyClass("Btech 4th year");
        student2.setRollno(21);
        //student2.setCollege("Suresh Techs College");

        System.out.println(student2.getName());
        System.out.println(student2.getStudyClass());
        System.out.println(student2.getRollno());
        System.out.println(student2.getCollege());

        student3 = new Student();
        student3.setName("Phiso");
        student3.setStudyClass("Btech 1st year");
        student3.setRollno(39);
        //student3.setCollege("Suresh Techs College");

        System.out.println(student3.getName());
        System.out.println(student3.getStudyClass());
        System.out.println(student3.getRollno());
        System.out.println(student3.getCollege());
    }
}
```



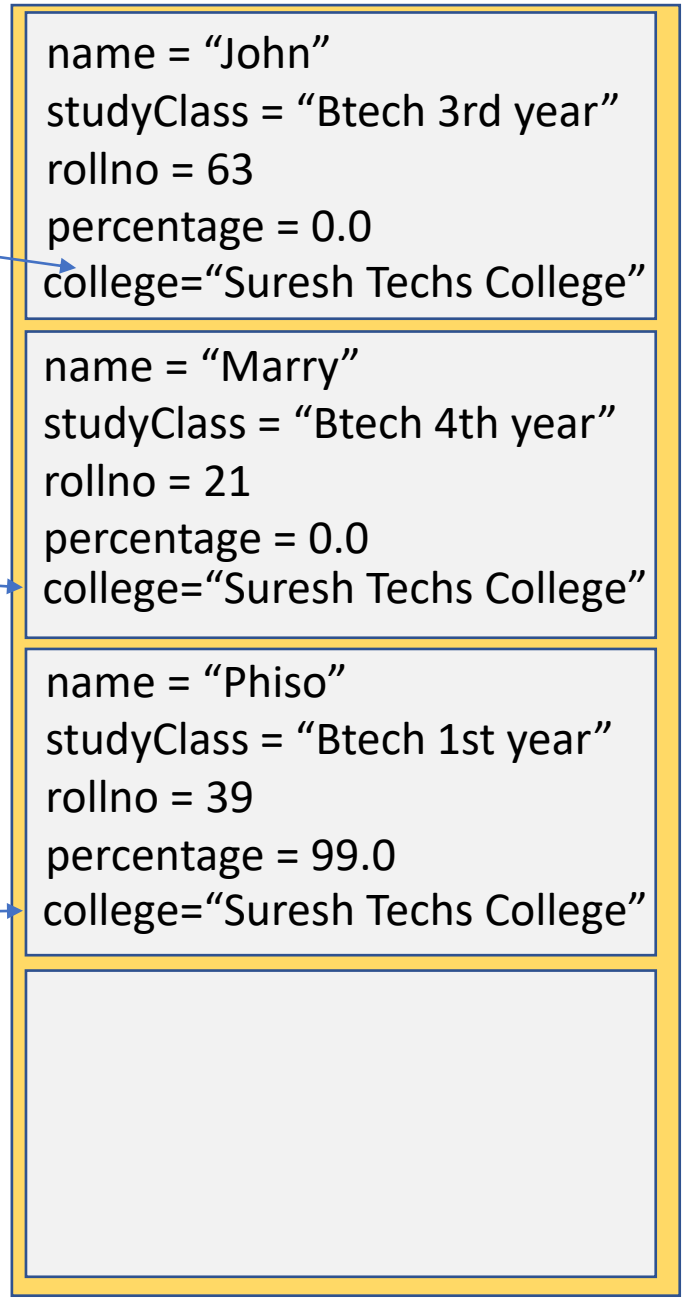
```java
class Student{
    String name;
    String studyClass;
    int rollno;
    double percentage;
    House h;
    String college="Suresh Techs College";
```

```java
class StaticDemo{
    public static void main(String[] args){
        Student student1, student2, student3;
        student1 = new Student();
        student1.setName("John");
        student1.setStudyClass("Btech 3rd year");
        student1.setRollno(63);
        //student1.setCollege("Suresh Techs College");

        System.out.println(student1.getName());
        System.out.println(student1.getStudyClass());
        System.out.println(student1.getRollno());
        System.out.println(student1.getCollege());

        student2 = new Student();
        student2.setName("Marry");
        student2.setStudyClass("Btech 4th year");
        student2.setRollno(21);
        //student2.setCollege("Suresh Techs College");

        System.out.println(student2.getName());
        System.out.println(student2.getStudyClass());
        System.out.println(student2.getRollno());
        System.out.println(student2.getCollege());

        student3 = new Student();
        student3.setName("Phiso");
        student3.setStudyClass("Btech 1st year");
        student3.setRollno(39);
        //student3.setCollege("Suresh Techs College");

        System.out.println(student3.getName());
        System.out.println(student3.getStudyClass());
        System.out.println(student3.getRollno());
        System.out.println(student3.getCollege());
    }
}
```

But the state(college) will be created for all the objects 😖 😖



Memory wastage



Memory

Local Variables

- Variables that are **declared within a method, constructor or block** are called local variables
- Local variables **doesn't get a default value**
- It is **mandatory to initiali** before use
- It **does not include any access modifiers** such as public, private, protected

```
public static void main(String[] args){
    int suitcase1;
    int suitcase2 = 2000;
    int suitcase3 = 3000;
    suitcase1 = 1000;
```

Instance Variables

- Variables that are declared **within the body of the class but outside of a method, constructor, or block** are called **Instance** variables
- Instance variables **gets a default value**
- **to initialize** instance e
- It **includes access modifiers** such as public, private, protected

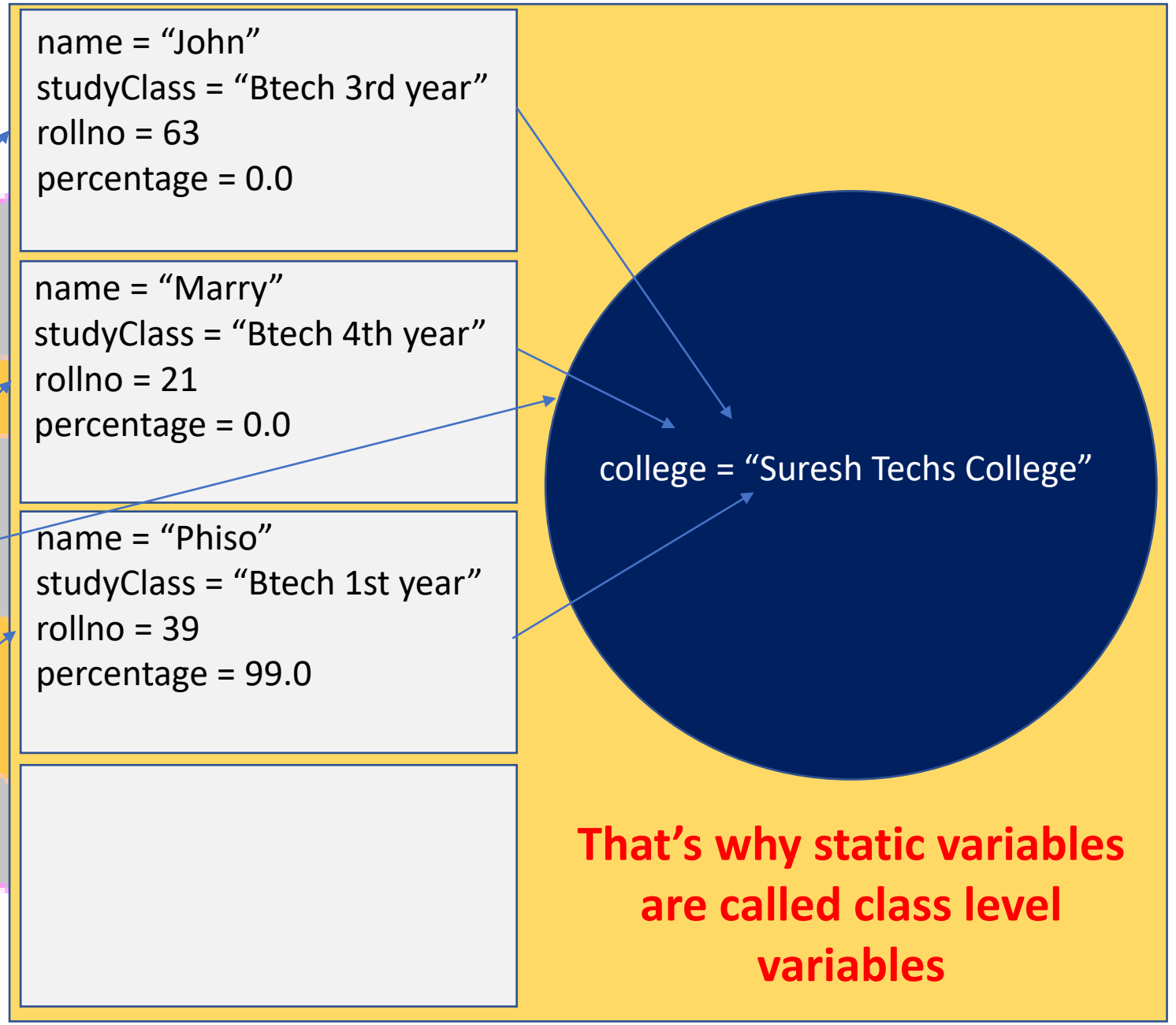
```
class Student{
    String name;
    String studyClass;
    int rollno;
    double percentage;
    House h;
```

Static variables

static

- **static** variable get's memory allocated only once during the time of class loading
- **static variables** will only have **one copy in the memory** and that is **shared across the objects**
- A variable can be **made static** by using **static** keyword

```
class Student{
    String name;
    String studyClass;
    int rollno;
    double percentage;
    House h;
    static String college="Suresh Techs College";


public static void main(String[] args){
    Student student1, student2, student3;
    student1 = new Student();
    student1.setName("John");
    student1.setStudyClass("Btech 3rd year");
    student1.setRollno(63);
//  student1.setCollege("Suresh Techs College");

    student2 = new Student();
    student2.setName("Marry");
    student2.setStudyClass("Btech 4th year");
    student2.setRollno(21);
//  student2.setCollege("Suresh Techs College");


    student3 = new Student();
    student3.setName("Phiso");
    student3.setStudyClass("Btech 1st year");
    student3.setRollno(39);
//  student3.setCollege("Suresh Techs College");
```



Accessing static variables

Accessing Instance variables

```
Student student1 = new Student();
student1.name = "John";
System.out.println(student1.name);
```

Accessing static variables

```
Student student1 = new Student();
student1.name = "John";
System.out.println(student1.name);
System.out.println(Student.college);
```

- We **don't need an object** to **access a static variables** since it is a class level variable and it **gets memory during the class loading** it self
- So, **we can directly access the static variables using class name**

Can’t we access a static variable by using object?

- **static variables** are **class level variables** and since there is no relation with the objects, it is better to access them using class name
- **Classname**.variablename

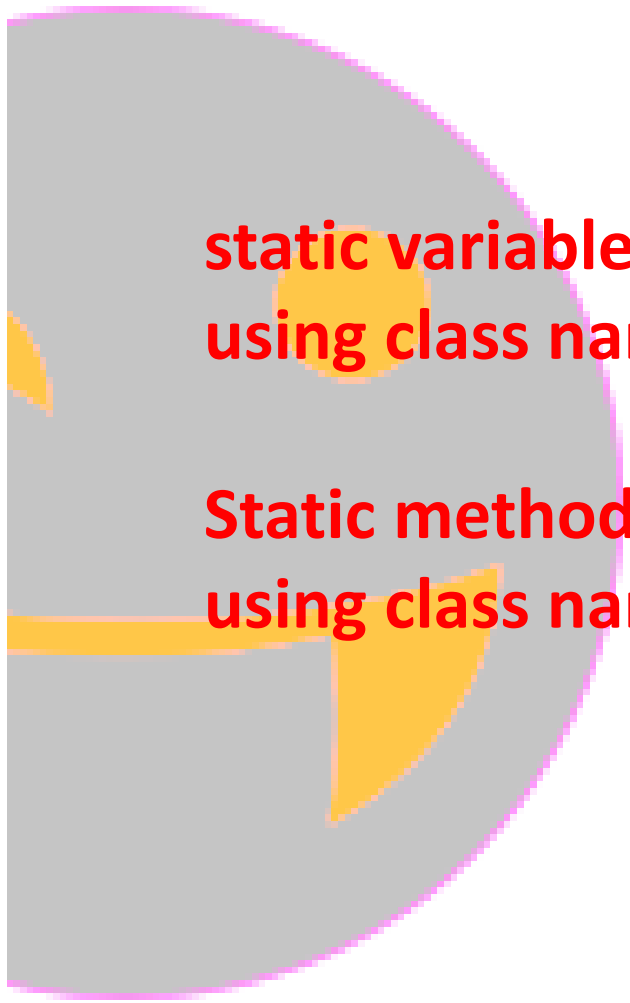
```
System.out.println(student1.getName());
System.out.println(student1.getStudyClass());
System.out.println(student1.getRollno());
System.out.println(student1.getCollege());
```

```
System.out.println(student1.getName());
System.out.println(student1.getStudyClass());
System.out.println(student1.getRollno());
System.out.println(Student.college);
```

Try to access instance variables(name, studyClass, rollno using classname)

Instance variables can only be accessed using an object from outside of the class

```java
class StaticDemo{
    public static void main(String[] args){
        Student student1, student2, student3;

        student1 = new Student();
        student1.setName("John");
        student1.setStudyClass("Btech 3rd year");
        student1.setRollno(63);

        System.out.println(student1.getName());
        System.out.println(student1.getStudyClass());
        System.out.println(student1.getRollno());
        System.out.println(Student.college);

        student2 = new Student();
        student2.setName("Marry");
        student2.setStudyClass("Btech 4th year");
        student2.setRollno(21);

        System.out.println(student2.getName());
        System.out.println(student2.getStudyClass());
        System.out.println(student2.getRollno());
        System.out.println(Student.college);

        student3 = new Student();
        student3.setName("Phiso");
        student3.setStudyClass("Btech 1st year");
        student3.setRollno(39);

        System.out.println(student3.getName());
        System.out.println(student3.getStudyClass());
        System.out.println(student3.getRollno());
        System.out.println(Student.college);
    }
}
```

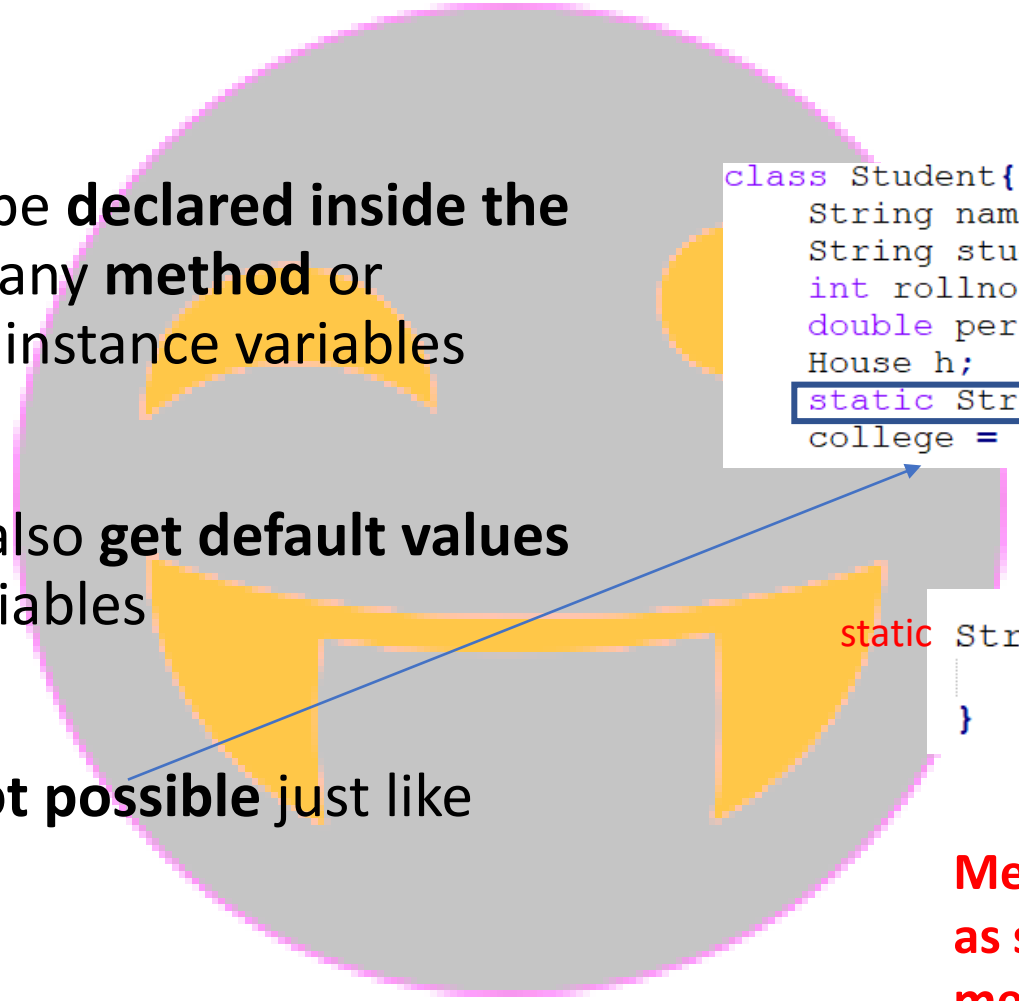


static variables can be accessed using class name

Static methods can also be accessed using class name

Static

- **Static** variables will be **declared inside the class** but **outside** of any **method** or **constructor** just like instance variables
- Static variables will also **get default values** just like instance variables
- **Reinitialization** is **not possible** just like instance variables
- We **can't make a local variable as static**



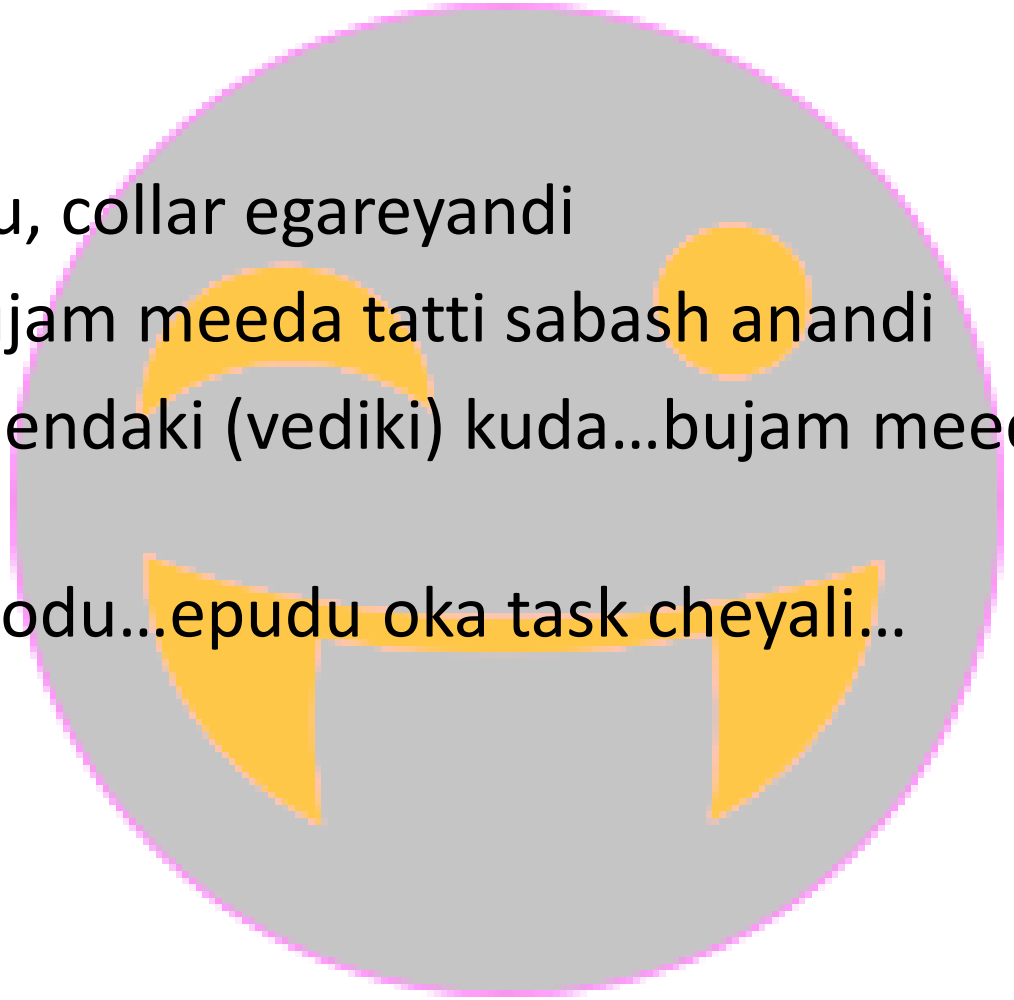
```
class Student{
    String name;
    String studyClass;
    int rollno;
    double percentage;
    House h;
    static String college="Suresh Techs College";
    college = "STC";
```

```
static String getCollege(){
    return college;
}
```

Methods that are declared as static are called static methods

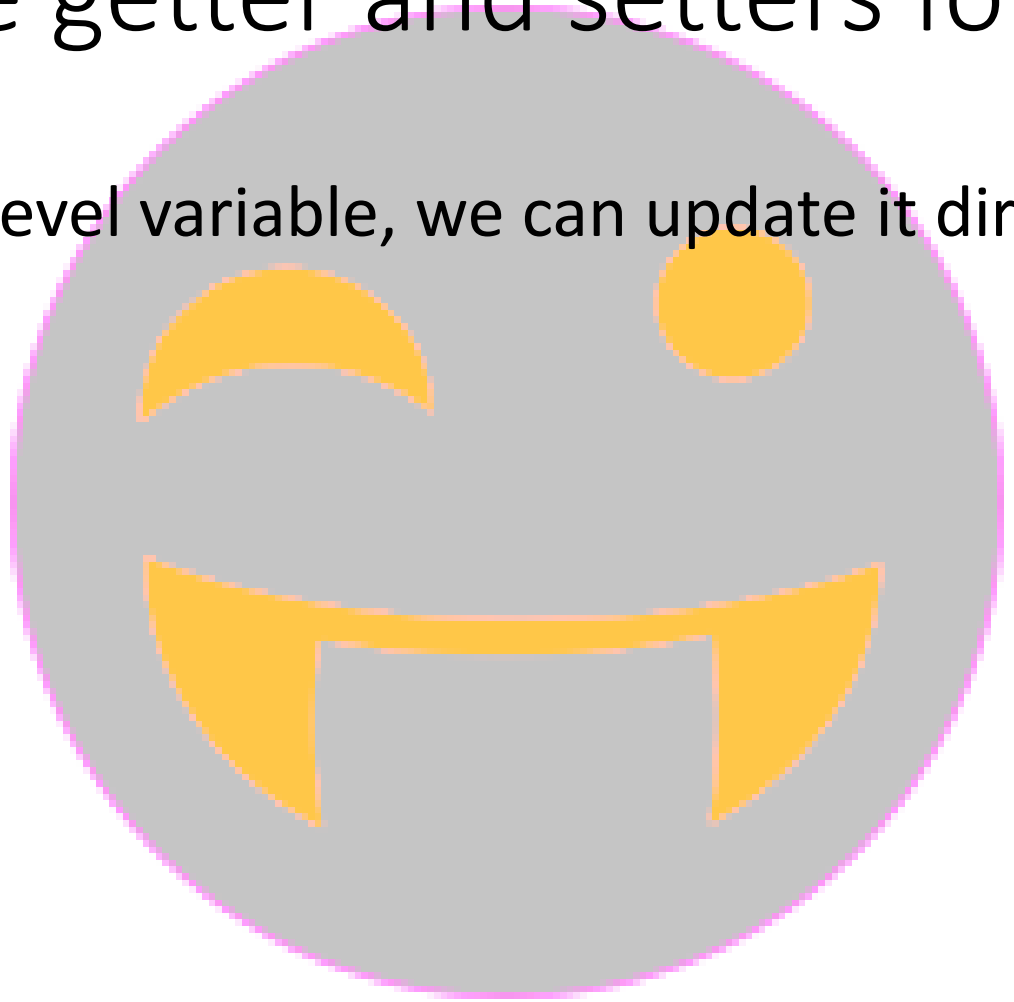
Static methods can also be accessed using class name

- Edi ardamina vallu, collar egareyandi
- Collar leni velli bujam meeda tatti sabash anandi
- Emi leni varu…ee endaki (vediki) kuda…bujam meeda cheyi vesi sabash anandi…
- Sabash ante saripodu…epudu oka task cheyali…



Let’s remove getter and setters for college

- Since it is a class level variable, we can update it directly using class name



Task

- Create an **instance variable** named **marks** of type **int** in Student class
- Create an **instance variable** named **totalStudents** of type **int** in Student class
- Set **marks 90 for student1**, **87 for student2**, **76 for student3** and **display the marks for each student at the bottom**
- **Compile and run the program**
- Set **totalStudents to 1** on **student1** object, **totalStudents to 2** on **student2**, **totalStudents to 3** on student3 and **display totalStudents for each student**
- **Compile and run the program**

```
class StaticDemo{
    public static void main(String[] args){
        Student student1, student2, student3;

        student1 = new Student();
        student1.setName("John");
        student1.setStudyClass("Btech 3rd year");
        student1.setRollno(63);
        student1.marks = 90;

        System.out.println(student1.getName());
        System.out.println(student1.getStudyClass());
        System.out.println(student1.getRollno());
        System.out.println(Student.college);

        student2 = new Student();
        student2.setName("Marry");
        student2.setStudyClass("Btech 4th year");
        student2.setRollno(21);
        student2.marks = 87;

        System.out.println(student2.getName());
        System.out.println(student2.getStudyClass());
        System.out.println(student2.getRollno());
        System.out.println(Student.college);

        student3 = new Student();
        student3.setName("Phiso");
        student3.setStudyClass("Btech 1st year");
        student3.setRollno(39);
        student3.marks = 76;

        System.out.println(student3.getName());
        System.out.println(student3.getStudyClass());
        System.out.println(student3.getRollno());
        System.out.println(Student.college);

        System.out.println("student 1 marks: "+student1.marks);
        System.out.println("student 2 marks: "+student2.marks);
        System.out.println("student 3 marks: "+student3.marks);
    }
}
```

```
class StaticDemo{
    public static void main(String[] args){
        Student student1, student2, student3;

        student1 = new Student();
        student1.setName("John");
        student1.setStudyClass("Btech 3rd year");
        student1.setRollno(63);
        student1.marks = 90;
        student1.totalStudents = 1;

        System.out.println(student1.getName());
        System.out.println(student1.getStudyClass());
        System.out.println(student1.getRollno());
        System.out.println(Student.college);

        student2 = new St
        student2.setName
        student2.setStudyClas                    ");
        student2.setRollno(21);
        student2.marks = 87;
        student2.totalStudents = 2;

        System.out.println(student2.getName());
        System.out.println(student2.getStudyClass());
        System.out.println(student2.getRollno());
        System.out.println(Student.college);

        student3 = new Student();
        student3.setName("Phiso");
        student3.setStudyClass("Btech 1st year");
        student3.setRollno(39);
        student3.marks = 76;
        student3.totalStudents = 3;

        System.out.println(student3.getName());
        System.out.println(student3.getStudyClass());
        System.out.println(student3.getRollno());
        System.out.println(Student.college);

        System.out.println("student 1 marks: "+student1.marks);
        System.out.println("student 2 marks: "+student2.marks);
        System.out.println("student 3 marks: "+student3.marks);

        System.out.println("Total students: "+student1.totalStudents);
        System.out.println("Total students: "+student2.totalStudents);
        System.out.println("Total students: "+student3.totalStudents);
    }
}
```

```
int marks;
static int totalStudents;
```

That's why static variables are referred to as global state variables

All the objects are sharing the same memory location 😄 😄 of the variable

```
Student constructor
John
Btech 3rd year
63
Suresh Techs College
Student constructor
Marry
Btech 4th year
21
Suresh Techs College
Student constructor
Phiso
Btech 1st year
39
Suresh Techs College
student 1 marks: 90
student 2 marks: 87
student 3 marks: 76
Total students: 3
Total students: 3
Total students: 3
```

```
Student constructor
John
Btech 3rd year
63
Suresh Techs College
Student constructor
Marry
Btech 4th year
21
Suresh Techs College
Student constructor
Phiso
Btech 1st year
39
Suresh Techs College
student 1 marks: 90
student 2 marks: 87
student 3 marks: 76
Total students: 1
Total students: 2
Total students: 3
```

Let's make totalStudents to a static variable and the run the program

Let’s modify our code to increment totalStudents on object creation

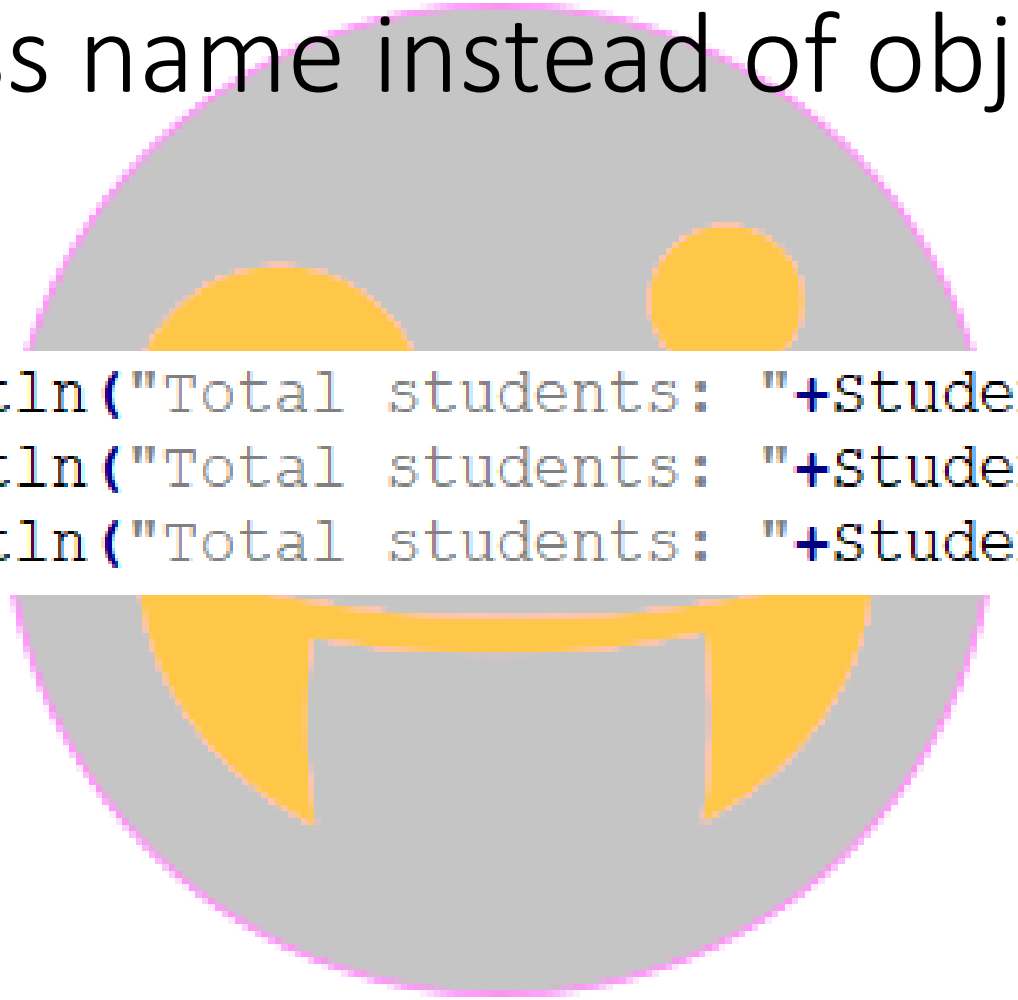
```
static int totalStudents;

Student(){
    System.out.println("Student constructor");
    totalStudents = totalStudents+1;
}
```

```
System.out.println(Student.totalStudents);
```

```
student1 = new Student();
student1.setName("John");
student1.setStudyClass("Btech 3rd year");
student1.setRollno(63);
student1.marks = 90;
//  student1.totalStudents = 1;

System.out.println(student1.getName());
System.out.println(student1.getStudyClass());
System.out.println(student1.getRollno());
System.out.println(Student.college);

student2 = new Student();
student2.setName("Marry");
student2.setStudyClass("Btech 4th year");
student2.setRollno(21);
student2.marks = 87;
//  student2.totalStudents = 2;

System.out.println(student2.getName());
System.out.println(student2.getStudyClass());
System.out.println(student2.getRollno());
System.out.println(Student.college);

student3 = new Student();
student3.setName("Phiso");
student3.setStudyClass("Btech 1st year");
student3.setRollno(39);
student3.marks = 76;
//student3.totalStudents = 3;
```

No need to increment manually 😍 😍

Let’s modify our code to access totalStudents by using class name instead of object



```
System.out.println("Total students: "+Student.totalStudents);
System.out.println("Total students: "+Student.totalStudents);
System.out.println("Total students: "+Student.totalStudents);
```

Static Variables

- A **static variable** is a **property of a class**
- A **static variable** is created **only once when the class loader loads the class**
- A **static variable** is used when you want to store a value **that represents all the instances like sum, average, totalStudentsCount etc**

```
static int totalStudents;
```

- **Local** variables
- **Instance** variables
- **Static** variables

Instance Variables

- An **Instance variable** is a **property of an instance(object)**
- An **Instance variable** is created **every time an instance is created**
- An **Instance variable** is used to store a value that **represents property of a single instance(object)**

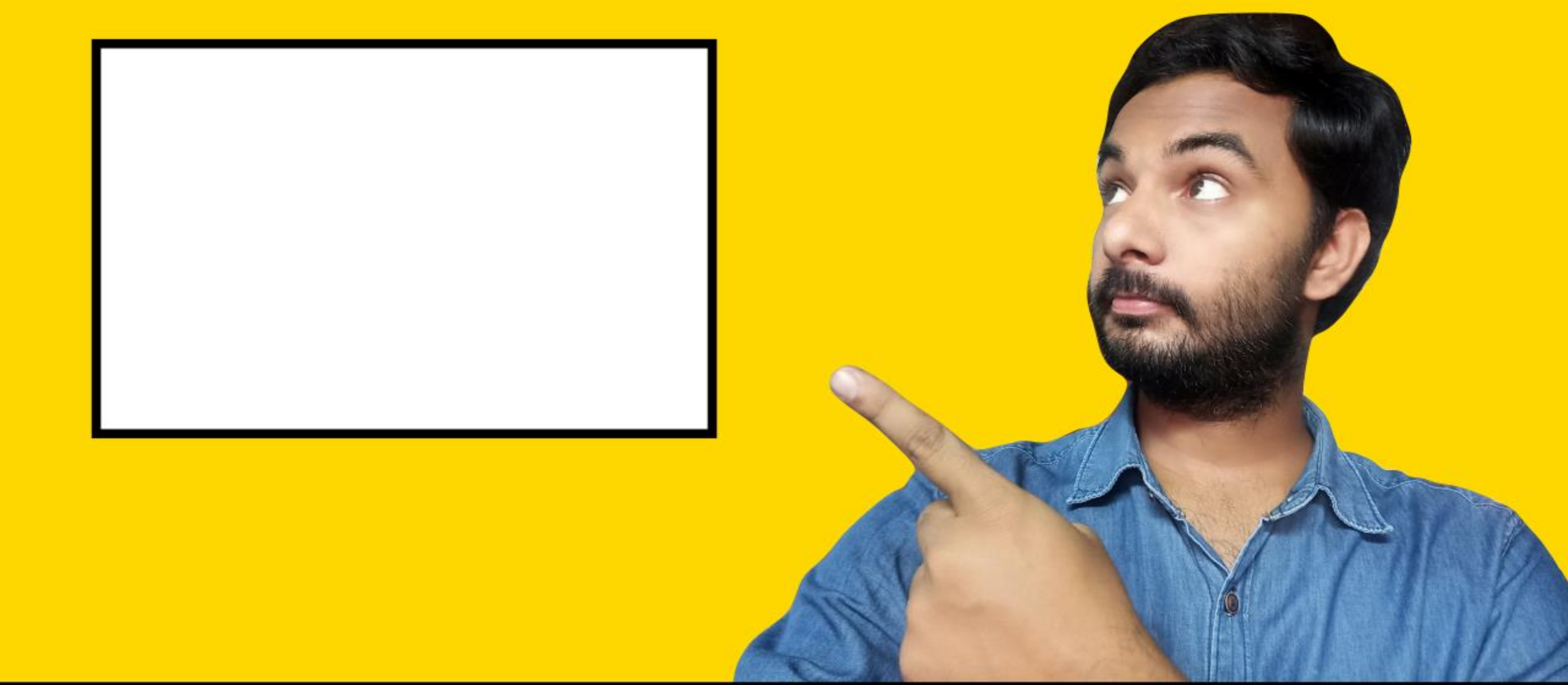
```
int marks;
```

```
static String getCollege(){
        return college;
}
```

Static methods

What next?

Static methods



చిన్న బ్రేక్ చిటికలో వచ్చేస్తా