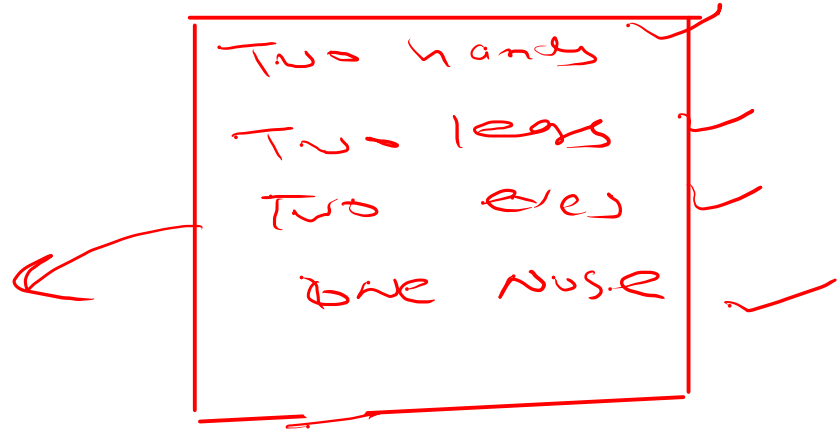
① Class object

class is a named group of properties & functions

Physical

Human



Constructor

Initialisation

1. default constructor
2. Parametrized constructor

Points

1. they don't have any return type
2. No Need to call the constructor
3.

```
class Emp {
    String name;
    int age;

    Emp(String name, int age) {
        this.name = name;
        this.age = age;
    }
}
psvm() {
    Emp e1 = new Emp();
    sysout(e1.name);
    sysout(e1.age);
```

Ranu
22

Default

Overloading

When the method Name are Same but different in arguments.

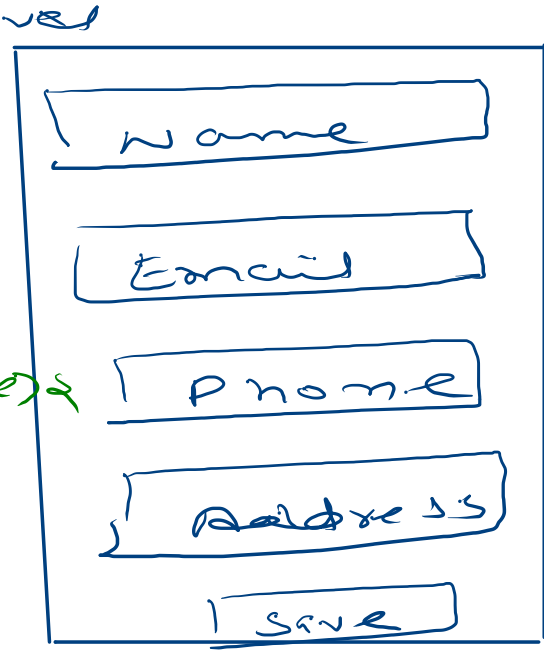
1. When the arguments are different.
2. Data types are different.
3. order are Not Same

(2)

```
Survey (String name, String email){
    this.name = name;
    this.email = email;
}
```

(3)

```
Survey (String name, String email, int phone){
    this.name = name;
    this.email = email;
    this.phone = phone;
}
```



2 = = 3

Survey (String name, String Email) {

this.name = name;

this.Email = email;

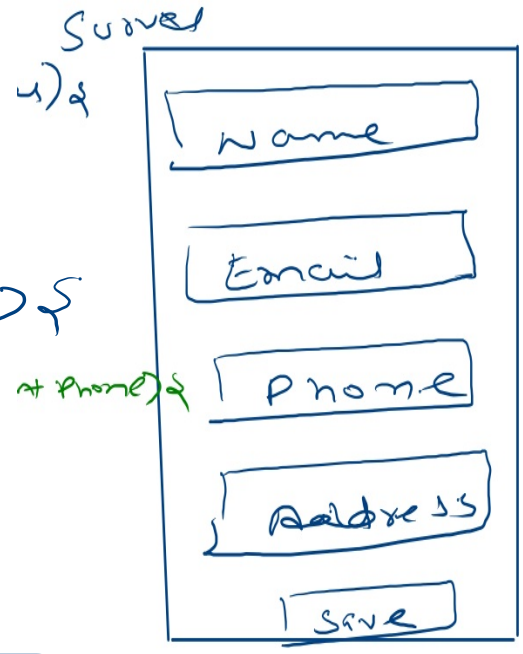
}

Survey (String email, int phone) {

this.email = email;

this.phone = phone;

}



2 == 2

+ve (or) false

```java
public class Emp {
    2 usages
    String name;
    4 usages
    String email;
    3 usages
    int phone;
    1 usage
    String address;

    no usages
    public Emp(){

    }
```

```java
public Emp(String name,String email) {
    this.name = name;
    this.email = email;

}
1 usage
public Emp(String email,int phone) {
    this.email = email;
    this.phone = phone;
}
no usages
public Emp(String name, String email, int phone, String address) {
    this.name = name;
    this.email = email;
    this.phone = phone;
    this.address = address;
}
```

```
no usages
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    String name = sc.next();
    String email = sc.next();
    int phone = sc.nextInt();

    Emp e1 = new Emp(name,email);
//      System.out.println(e1.name);
//      System.out.println(e1.email);
//      System.out.println(e1.phone);
//      System.out.println(e1.address);

    Emp e2 = new Emp(email,phone);
    System.out.println(e2.email);
    System.out.println(e2.phone);
}
```

H.W

class movie {

Name; — String

Category; — String

Rating; — int

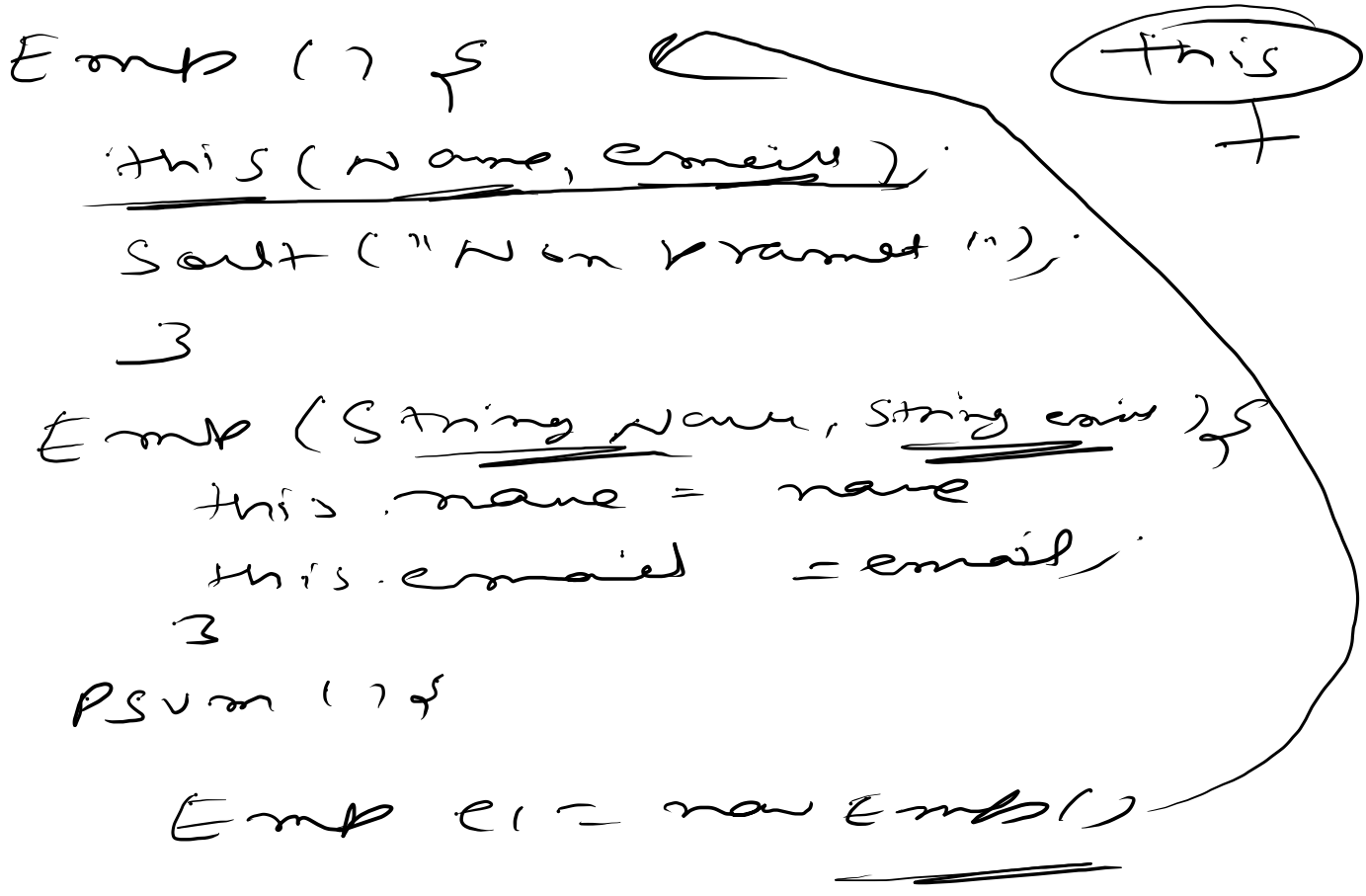
Collection — int

}

Constructor chaining

```
static void fun greeting() {
    sysо("Hello");
}
void fun welcome() {
    greeting();
    sysо("welcome");
}
```

Output: Hello
welcome


Emp () {
this(name, email);
Sout("No parameter");
}
Emp (String name, String email) {
this.name = name
this.email = email;
}
PSVM () {
Emp e1 = new Emp()
this