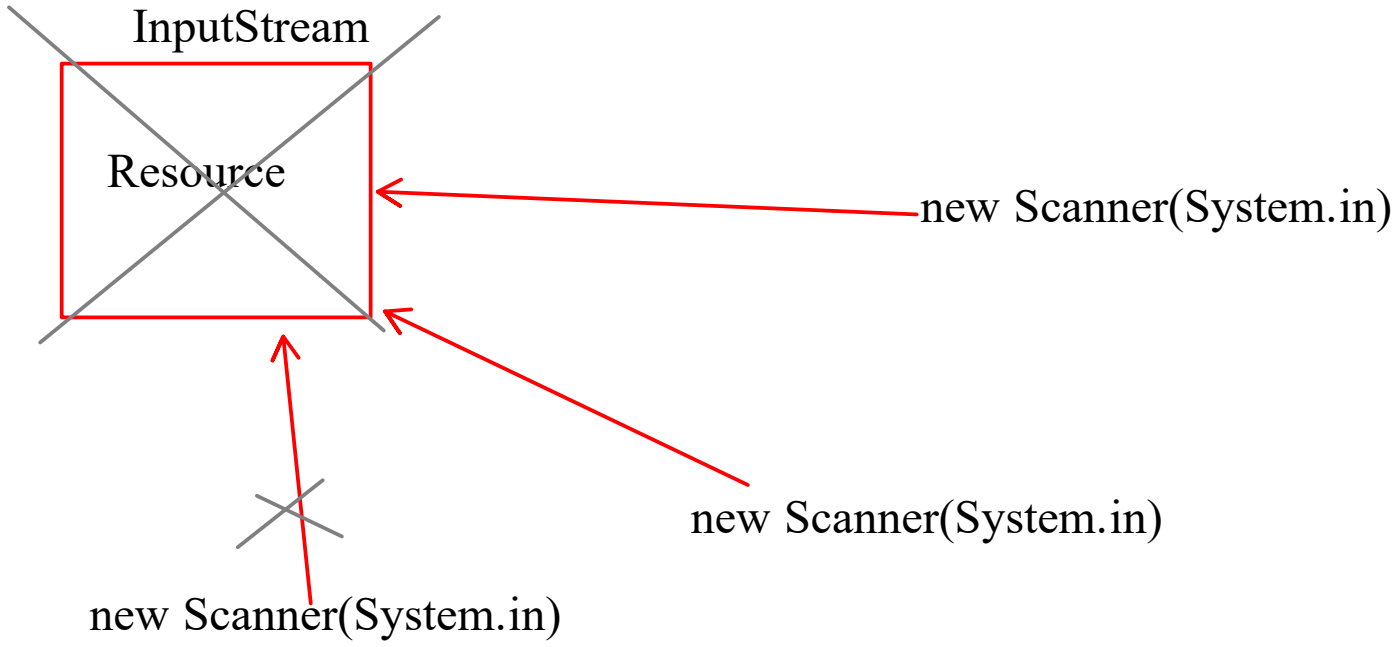
```
Point2D
{
int x;
int y;

public boolean isEqual(Point2D p){

}

public String getDetails(){
return x + "," + y;
}

calculateDistance(Point2D p){

}
}
```

this reference
Types of methods
Pass by Value, Pass by reference
Array

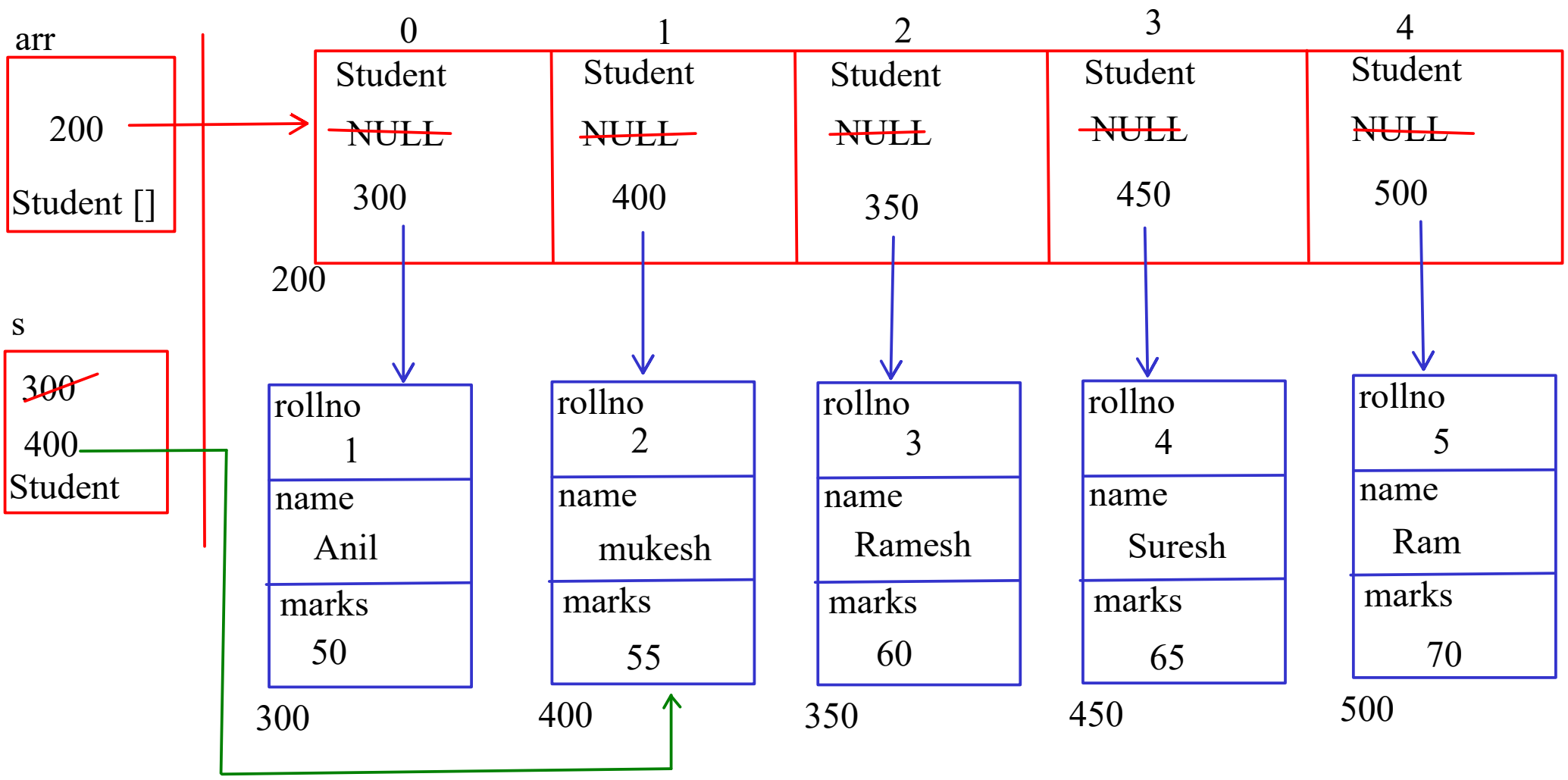
Array

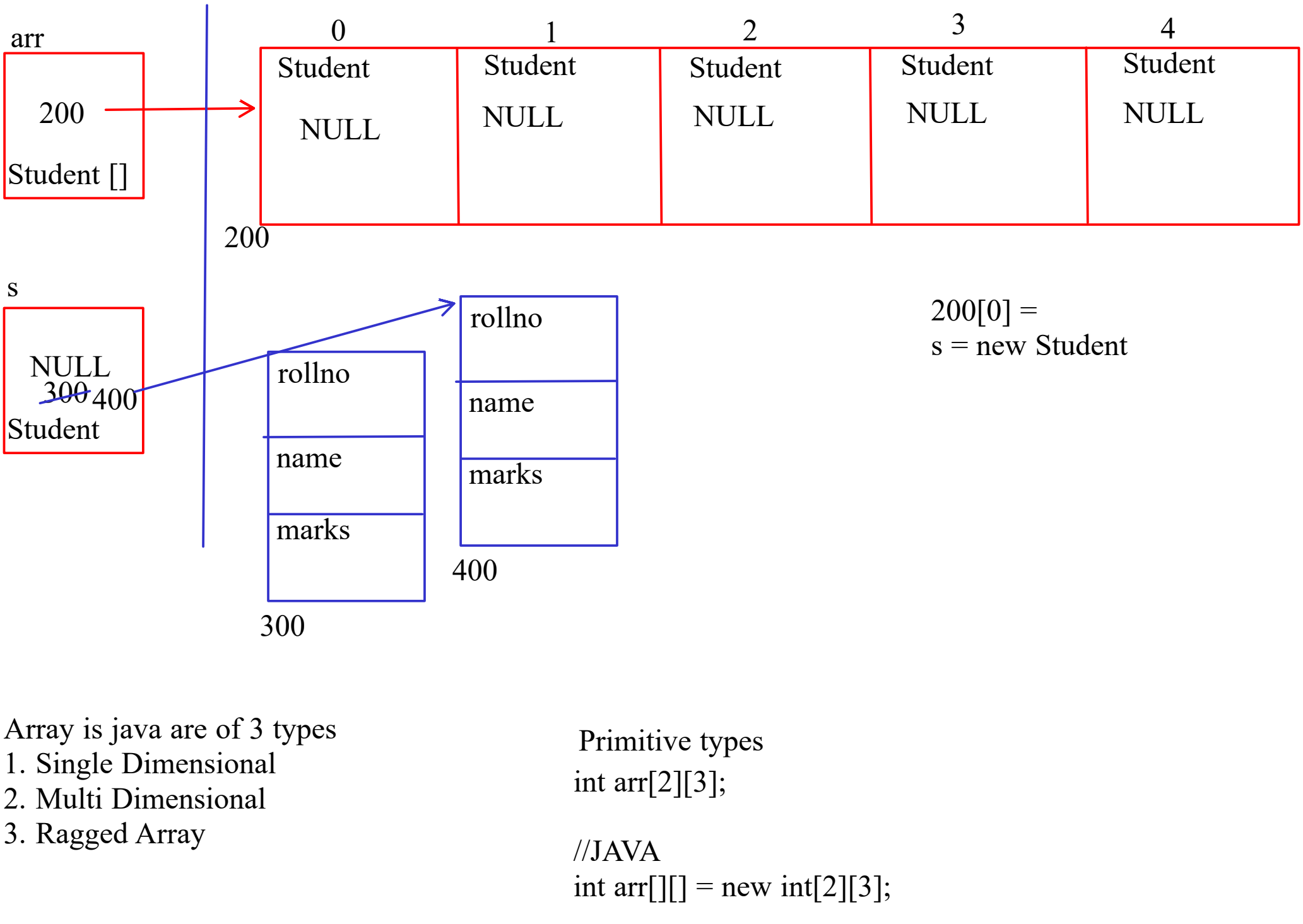
```
int arr[] = new int[5];
arr[0] = 10;
int element = arr[0];

for-each
for(int element:arr){
sysout(element)
}
```

```
Student **ptr = new Student*[5];
Student arr[] = new Student[5];

Student arr[5];
Student *ptr = new Student[5];
Student **ptr = new Student*[5];
```



sc.close();



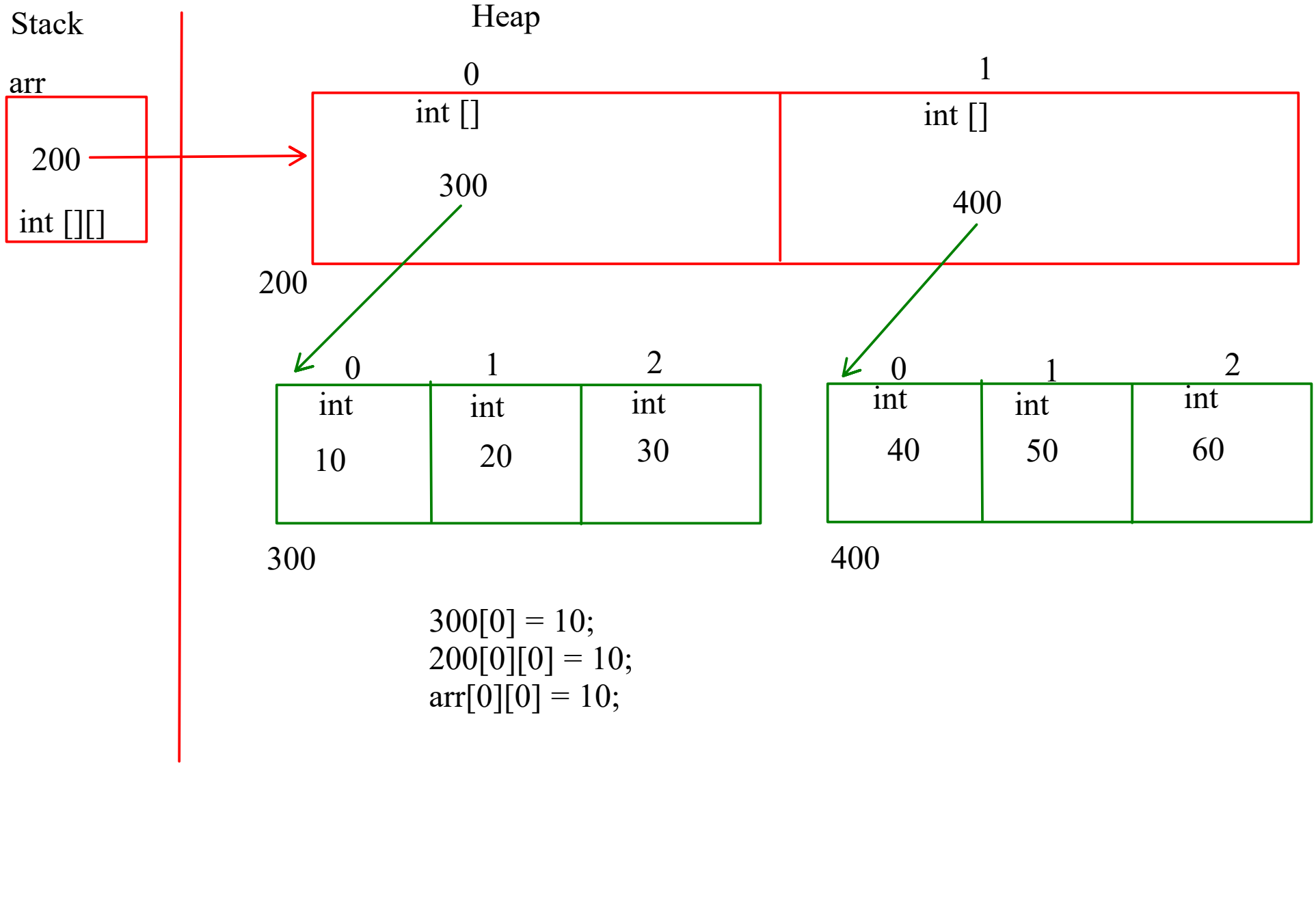


Array is java are of 3 types
1. Single Dimensional
2. Multi Dimensional
3. Ragged Array

Primitive types

```
int arr[2][3];

//JAVA
int arr[][] = new int[2][3];

//CPP
int **ptr = new int*[2];
arr[0] = new int[3];
arr[1] = new int[5];
```

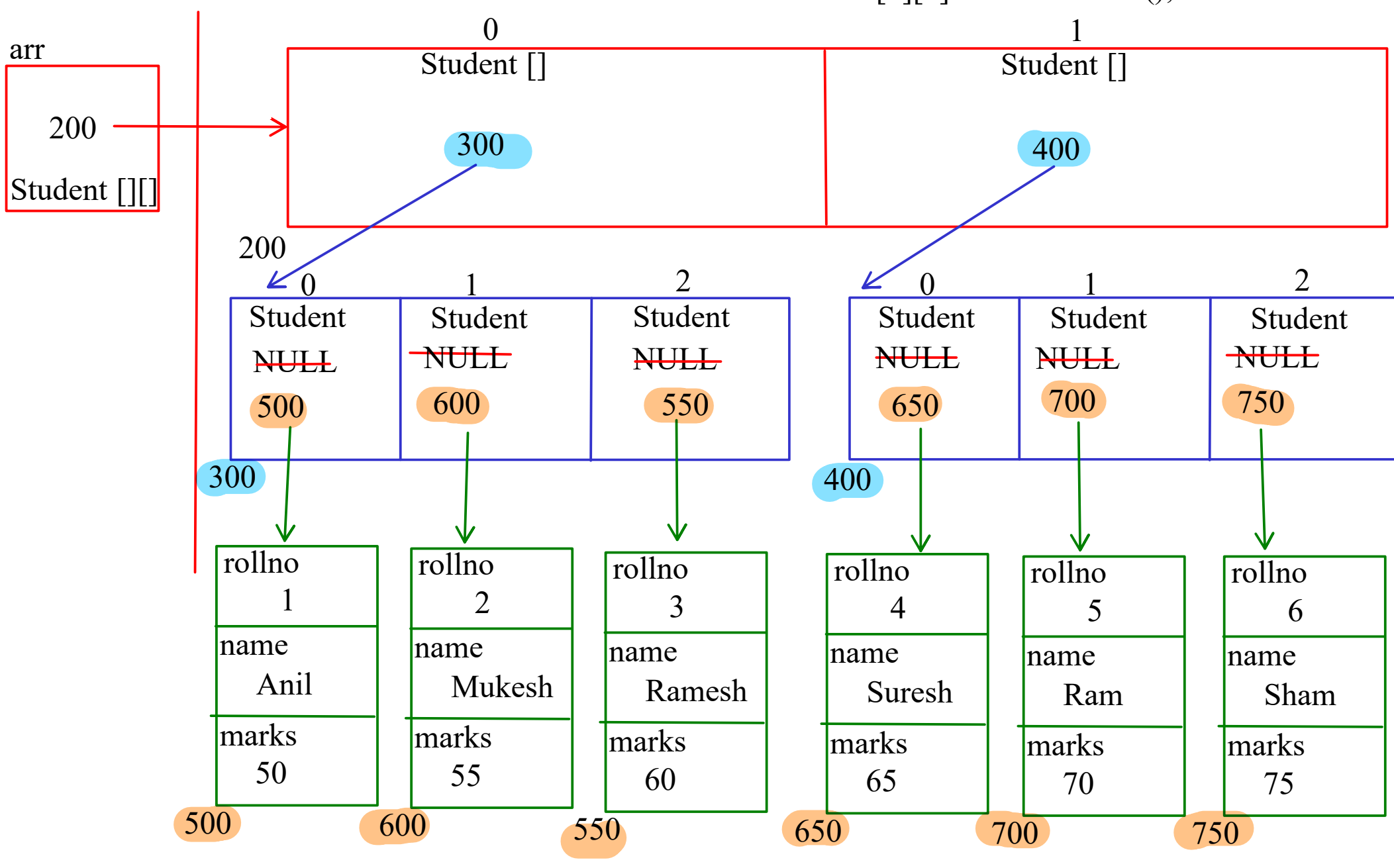
int arr[2][3];



300[0] = 10;
200[0][0] = 10;
arr[0][0] = 10;

```
Student[][] arr = new Student[2][3];
```

```
Student ***arr = new Student**[2];
arr[0] = new Student*[3];
arr[1] = new Student*[3];
arr[0][0] = new Student();
```




```
for(i)
        for(j)
arr[i][j].display();
```

```
for(Student [] ele : arr)
        for(Student s : ele)
                s.display();
```

```
int arr[] = new int[5];
arr[0] = 10;

int arr[][] = new int[2][3];

int arr[][] = new int[2][];
arr[0] = new int[3];
arr[1] = new int[5];
```

```
int arr[5];
arr[0] = 10;

int *ptr = new int[5];
ptr[0] = 10;

int **ptr = new int*[5];
ptr[0] = new int(10);
```

```
int arr[2][3];
arr[0][0] = 10;

int **ptr = new int*[2];
ptr[0] = new int[3];
ptr[1] = new int[3];
ptr[0][0] = 10;

int ***ptr = new int**[2];
ptr[0] = new int*[3];
ptr[1] = new int*[5];
ptr[0][0] = new int(10);
```

final Keyword

1. Variables
    - It can be initialized after the declaration
    - Once initialized we cannot change the value inside it.
2. Fields
    - We can declare the fields of the class as final,
    - they can be initialized in field initializer or object initializer or constructor
    - Once initialized we cannot change the value inside it.
    - we cannot initialize the final fields in the setters, getters or facilitators
3. Methods
4. Class

```
final Student s;
s = new Student();
s = new Student();
```

Non-Primitive (Reference type)
array
class
interface
enum

```
enum EMenu{
}
```

```
interface Shape{
}
```

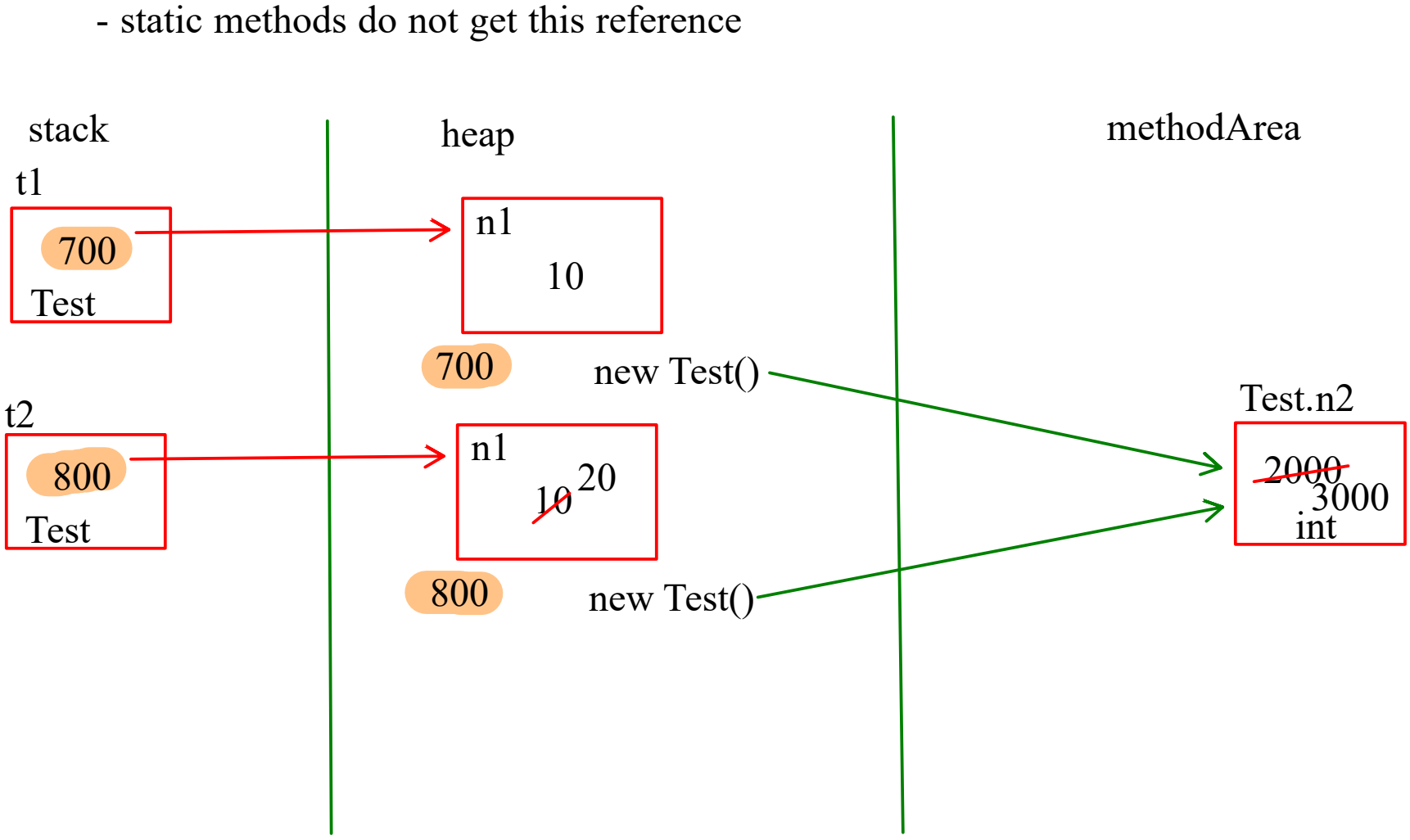
```
class Student{
}
```

```
Student [ ] arr = new Student[5];
```

Static

- We can declare the

1. Fields
    - If fields are made static then the memory for the fields will be given on the method area only once at the time of class loading.
2. Methods
    - If methods are static then they are designed to be accessed on classname using . operator
    - They can access only the static fields of the class, they cannot access the non static fields of the class
    - static methods do not get this reference



Lab ->