Package -> It is a container used to
1. Organize the code
2. Avooid the ambugity

```
package p1;
```

directory p1 -> .class

Package p2; -> import p1.*;

company domain name in reverse.project.module

Access Modifiers
1. private -> Only in the same class
2. default -> In the declared package
3. protected -> In the delclared package and in other package in sub class only
4. public-> everywhere

```
public class Employee{

}
```

```
Employee e1 = new Employee();
e1.acceptEmployee();
e2.acceptEmployee();
```

```
void acceptEmployee() // Employee this
{

}
```

Types of Methods
Constructor -> To initialize the state of an object
Setters
Getters
Facilitators

constructor
1. Paramterless/Default
2. Paramterized

```
Employee e1 = new Employee(); // without arguments
Employee e2 = new Employee(1,"e2",2000);
                                    // with Arguments
```

class Fields -> private
outside the class to provide write and read operation for such fields
we provide setters and getters

```
void setFiledName(Type value)
```

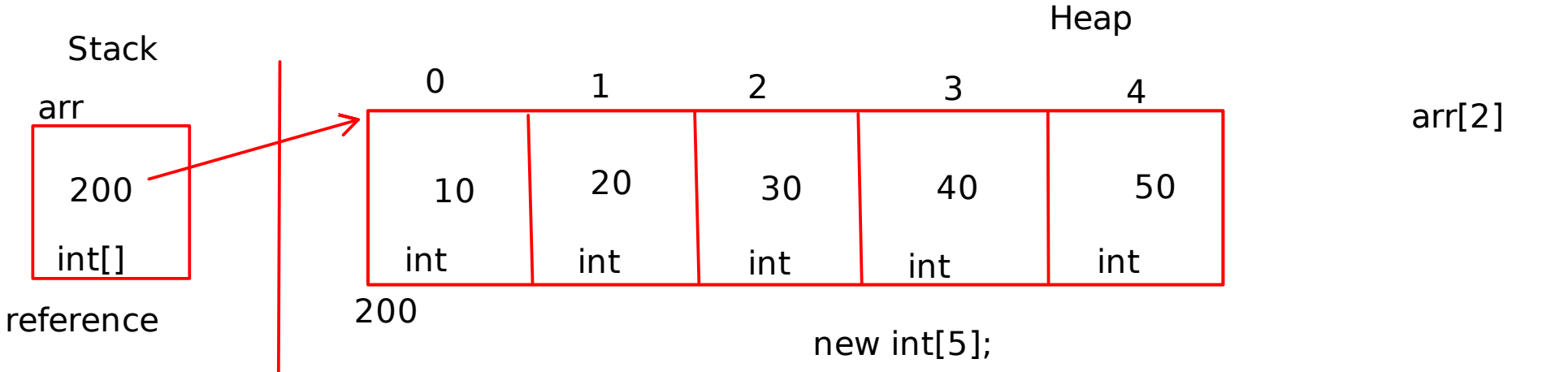
```
type getFileName(){
return field;
}
```

Constructor Chaning
calling a constuctor of a class from another constructor is called as ctor chaning
this() // this statement -> first statement in the ctor body

Array
data structure that is used to store smiliar type of data in contigious memory location
size of array is fixed
arry is a reference type in java

```
int arr[]; // reference
arr = new int[5];
```

```
int *ptr;
ptr = new int[5];
```



element
20
int

```
// for-loop
for(int i = 0; i<5 ; i++){
arr[i] = 10*(i+1);
}
```

```
// for-each
int i=1;
for(int element : arr){
element = 10*i++;
}
```

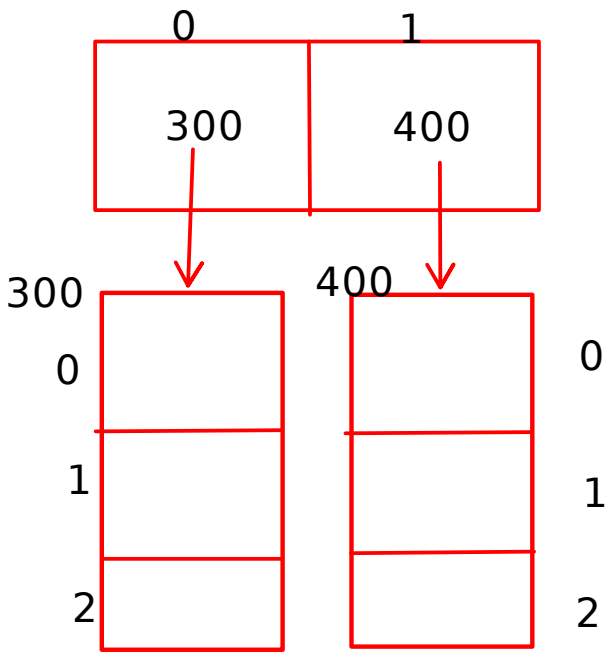
```
// for-loop
for(int i = 0; i<5 ; i++){
int element = arr[i];
sysout(element);
}
```

```
for(int element : arr){
sysout(element);
}
```

Array
3 types of array
1. Single Dimension
2. MultiDimension -> 2D Array
3. Ragged

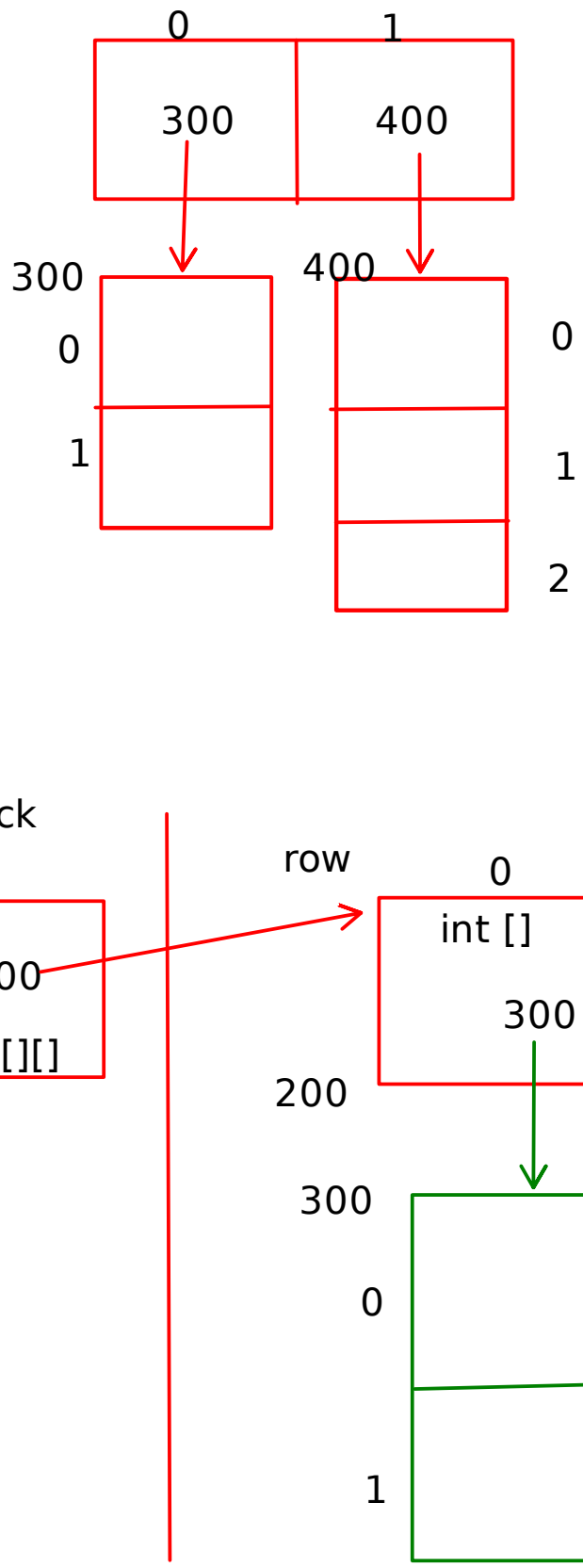
int arr1 = new int[5];

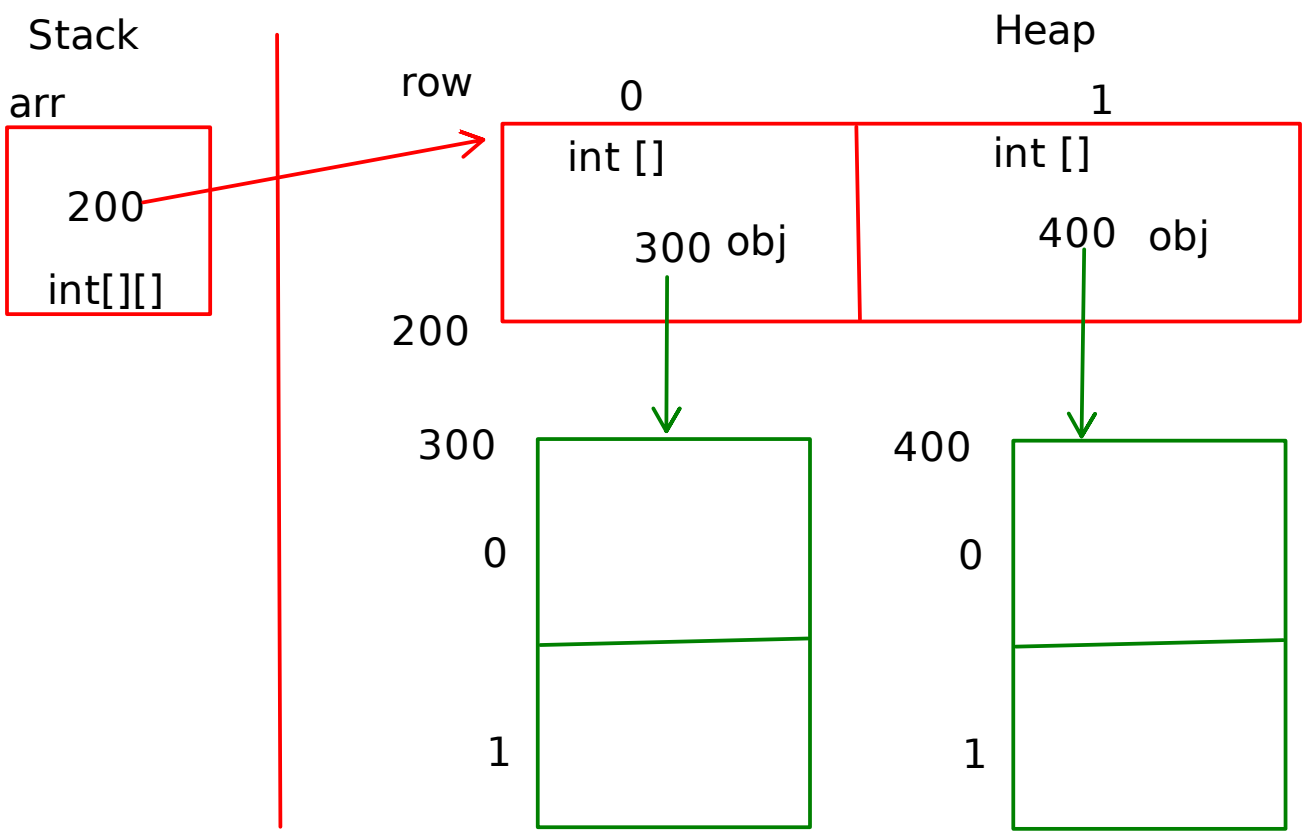
int arr2[][] = new int[2][3];



int arr3[][] = new int[2][];
arr3[0] = new int[2];
arr3[1] = new int[3];

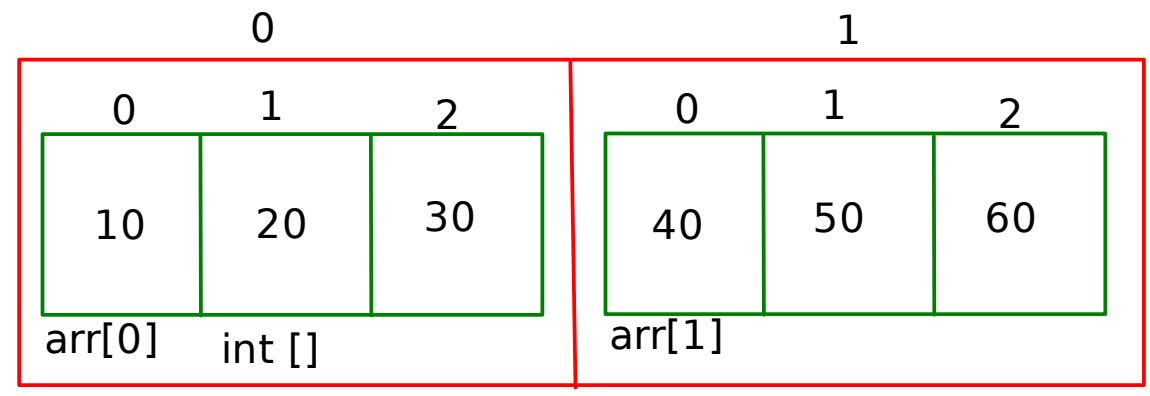
Ragged





new int[2][2]

	col0	col1
row 0	10	20
row1	30	40



arr

arr[0] [0] = 10
arr[0] [1] = 20
arr[0] [2] = 30

arr[1] [0]
arr[1] [1]
arr[1] [2]

arr[0]-> obj.length

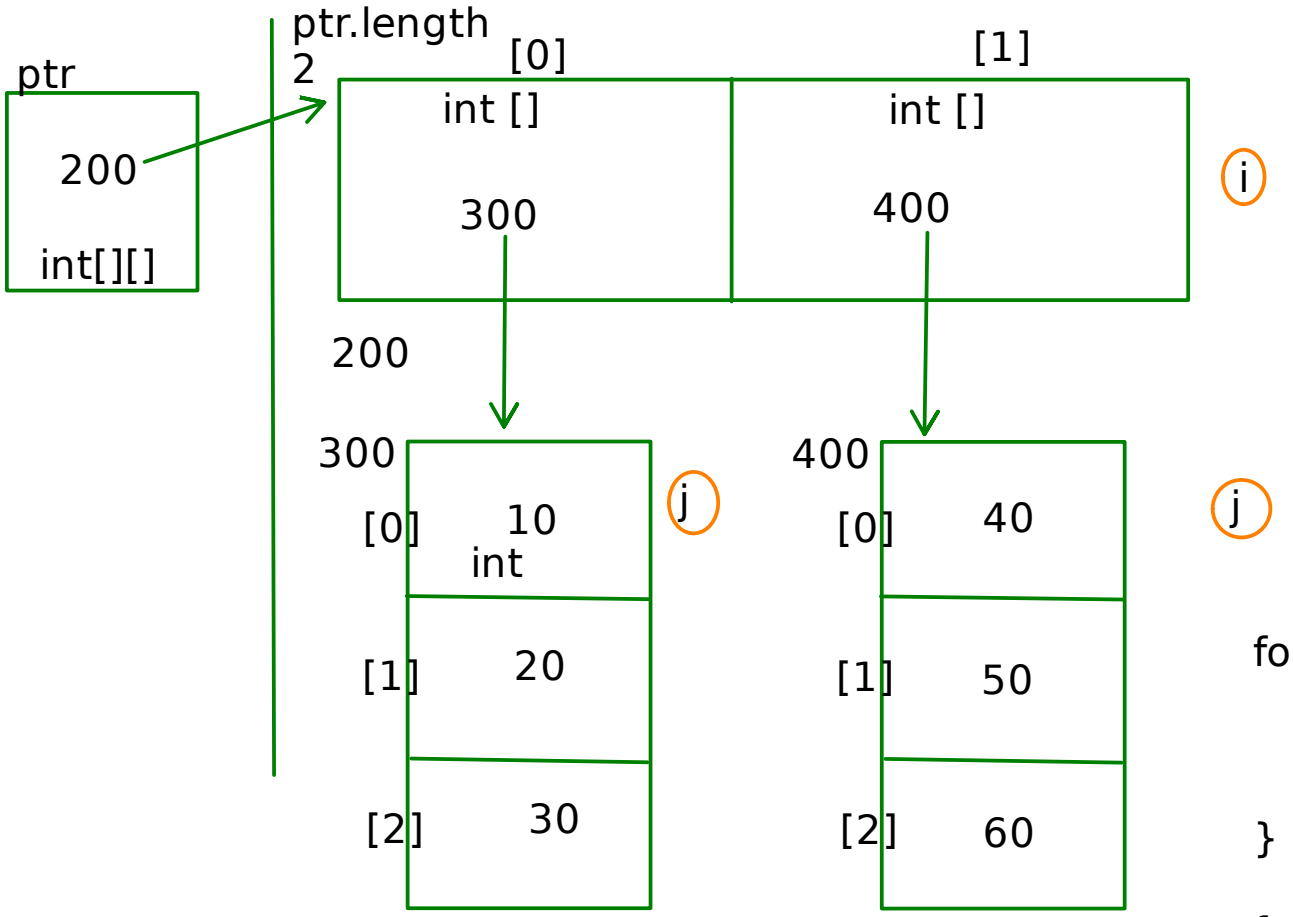
arr.length
200.length

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

for -> termination
length array -> ?

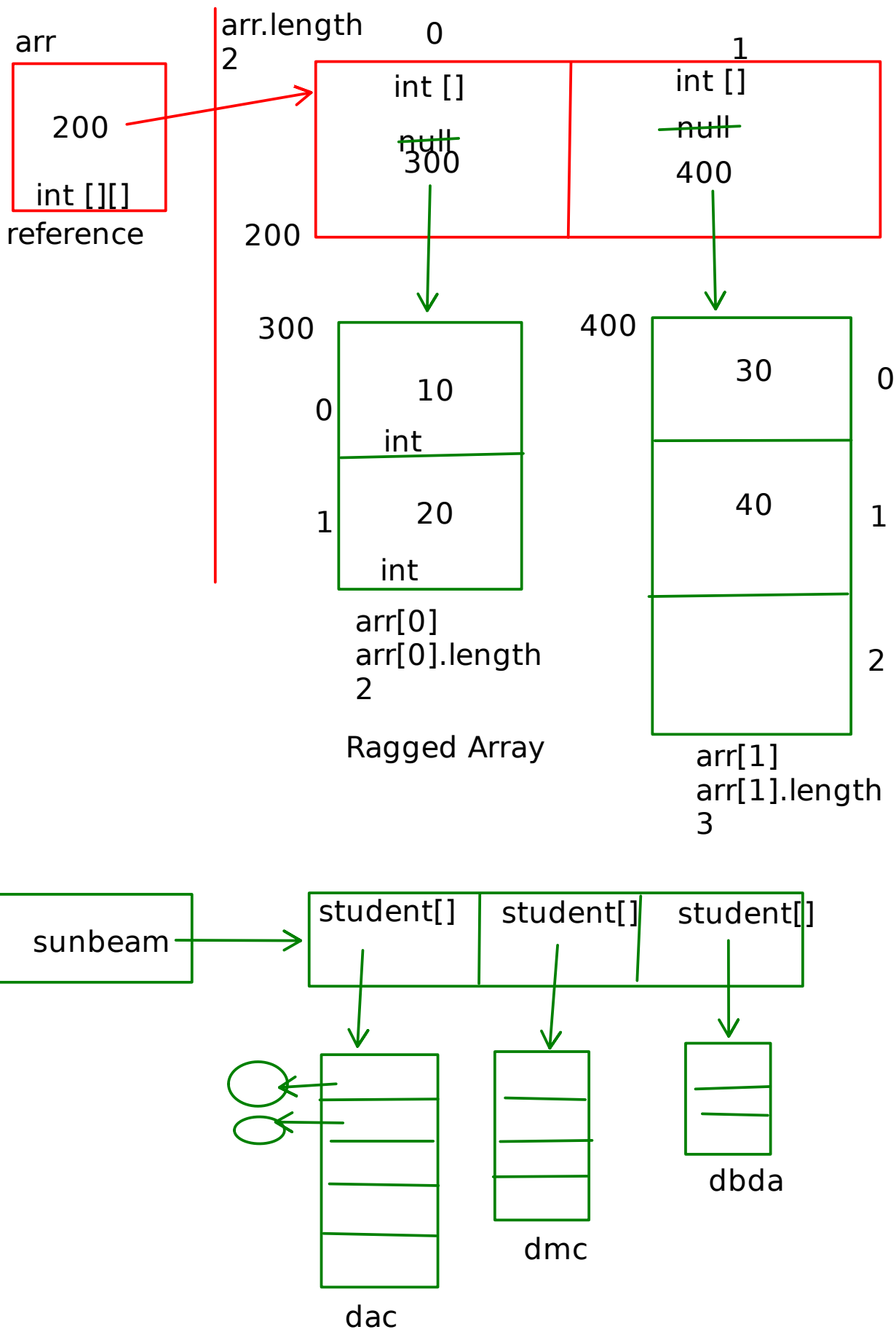
arr[0].length
arr[1].length

arr.length



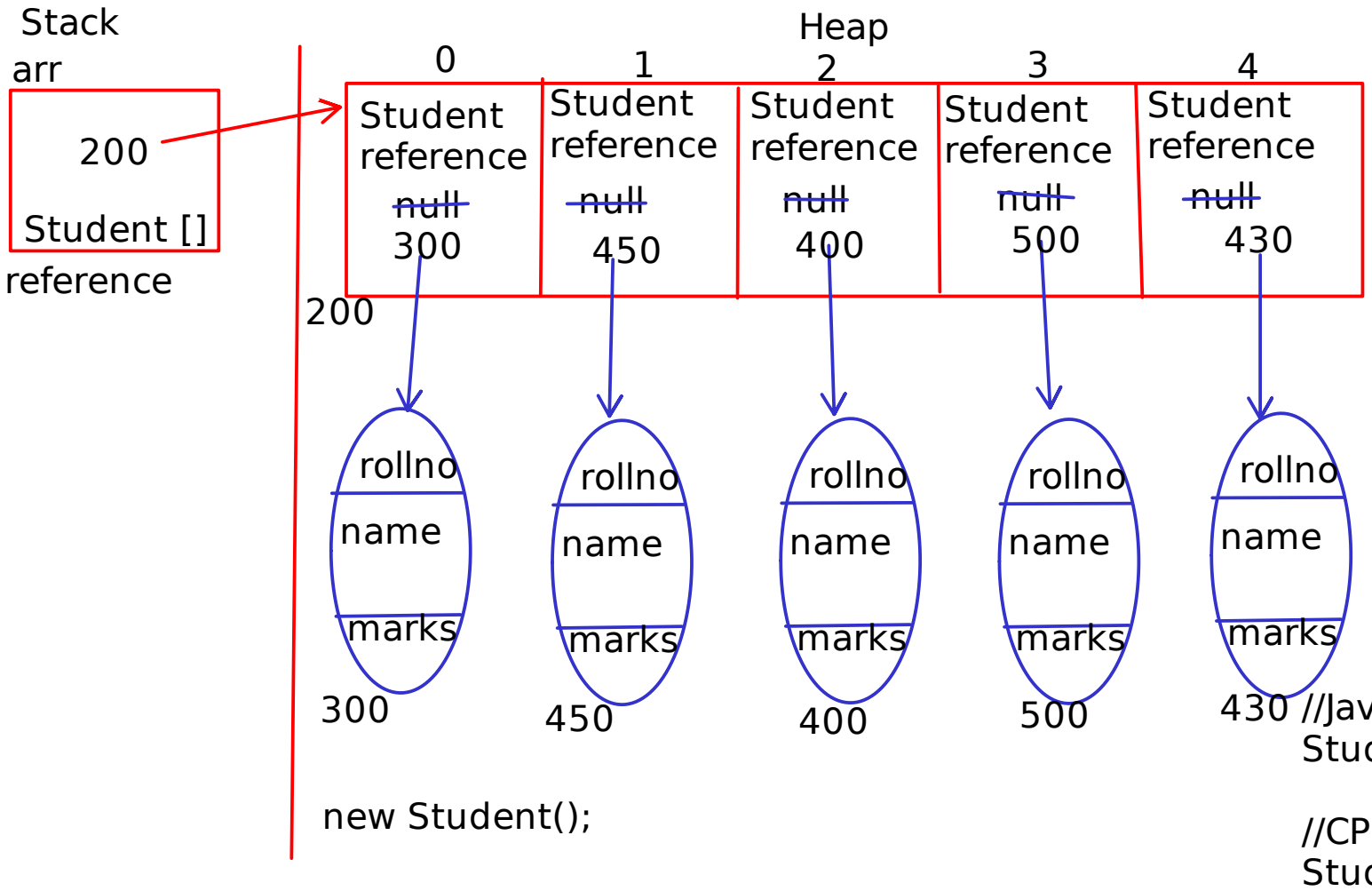

```
int ptr[][] = new int [2][3]
// ptr[0] = new int[3];
// ptr[1] = new int[3];

      ptr[0][0] = 10
      ptr[0][1] = 20
      ptr[0][2] = 30
      ptr[1][0] = 40
      ptr[1][1] = 50
      ptr[1][2] = 60
```

```
for(int i=0 i<ptr.length ; i++){
        for(int j=0;j<ptr[i].length;j++){
                sysout(ptr[i][j])
        }
}

for(int[] earr : ptr)
        for(int element :earr)
                sysout(element);
```




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

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


arr[0][0] = 10;
arr[0][1] = 20;

arr[1][0] = 30;
arr[1][1] = 40;
arr[1][2] = 50;
```

```
for(int i=0;i<arr.length;i++)
{
        for(int j=0;j<arr[i].length;j++){
                sysout(arr[i][j])
}
}

 for(int [] earr : arr)
        for(int element : earr)
                sysout(element)
```




```
Student[] arr;
arr = new Student[5];
arr[0] = new Student();
```

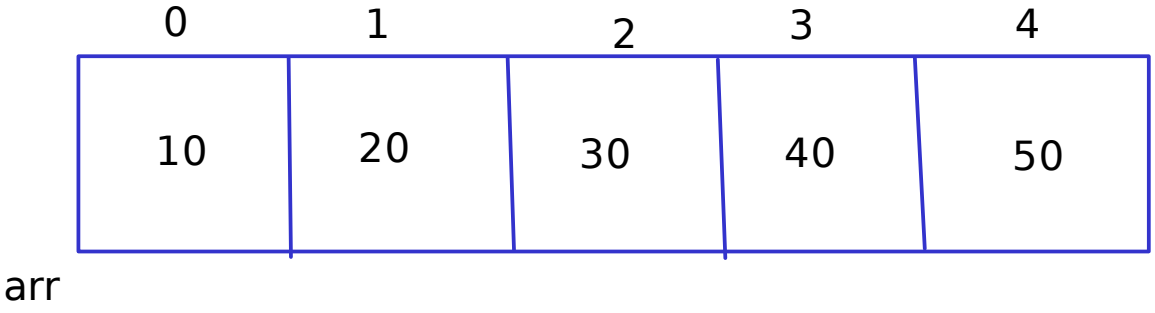
```
//Java
Student[] arr = new Student[5];

//CPP
Student**ptr = new Student*[5];
```

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

int arr[][] = new int[2][];
arr[0] = new int[3];
arr[1] = new int[4];
```

```
for( : ){

}
```

Method Overloading
- Defining the method multiple time with same name but different signature is called as method overloading
- Rules of method overloading
1. No of parameters should be different
2. If no of parameters are same then their type of parameters should be different
3. If no and type are same then the order of paramters should be different

int arr[] = new arr[5]; single dimension array



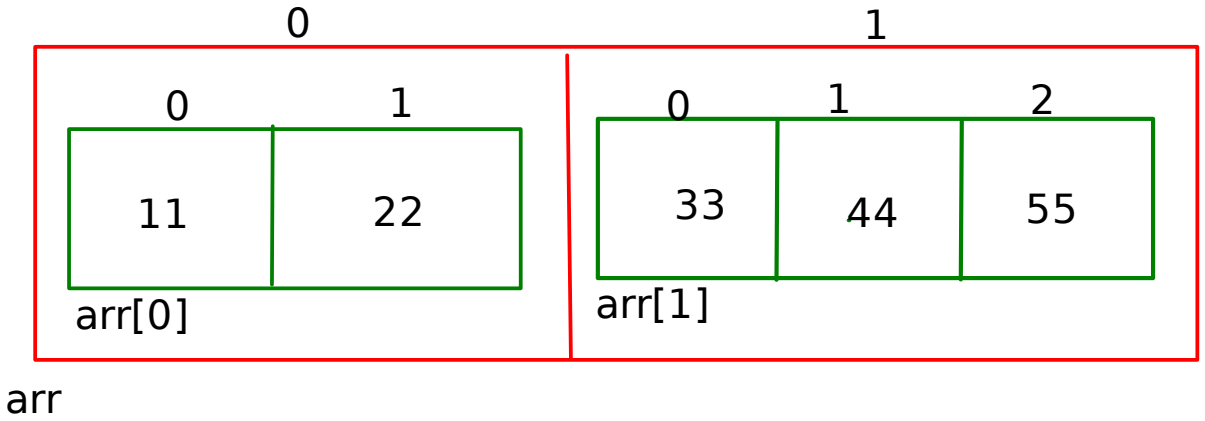

```
arr[0] = 10;
arr[1] = 20;
arr[2] = 30;
arr[3] = 40;
arr[4] = 50;
```

int arr[][] = new int[2][2]


```
arr[0][0]=11;
arr[0][1]=22;
arr[1][0]=33;
arr[1][1]=44;
```

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

Ragged Array




```
arr[0][0]=11;
arr[0][1]=22;
arr[1][0]=33;
arr[1][1]=44;
arr[1][2]=55;
```

```
Employee []arr;
arr = new Employee[3];
```

```
arr[0] = new Employee();
arr[1] = new Employee();
arr[2] = new Employee();
```