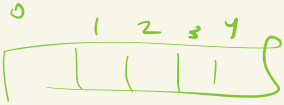
Arraylist (Dynamic memory array)

int [] arr = new int [5]

↓

static

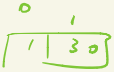


Syntax

ArrayList < Integer > al = new ArrayList<>();

functions

add al.add(1); al.add(15)



get int val = al.get(1) al.get(idx);

set al.set (idx, val); al.set (1, 30)

size al.size();

Remove al.remove (idx) → O(n) → shifting of indexes

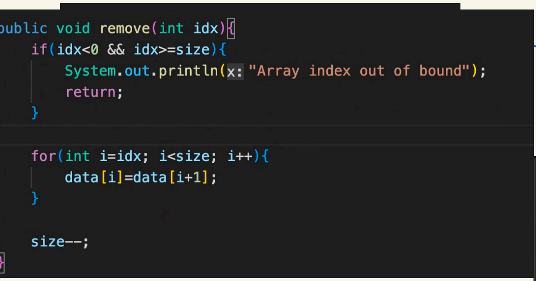
⇒ Implement Arraylist

→ add → size

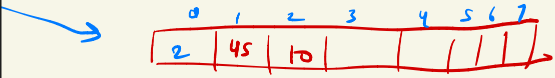
→ get → remove

→ set

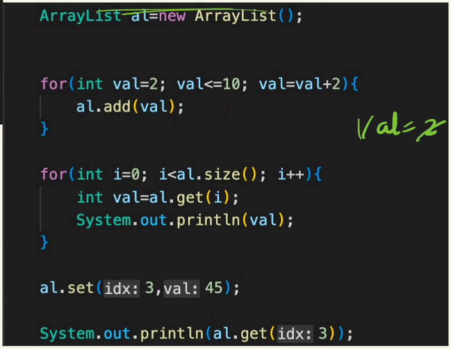
① al.add(75) ⑥



```
ublic void remove(int idx){
    if(idx<0 && idx>=size){
        System.out.println(x: "Array index out of bound");
        return;
    }

    for(int i=idx; i<size; i++){
        data[i]=data[i+1];
    }

    size--;
}
```

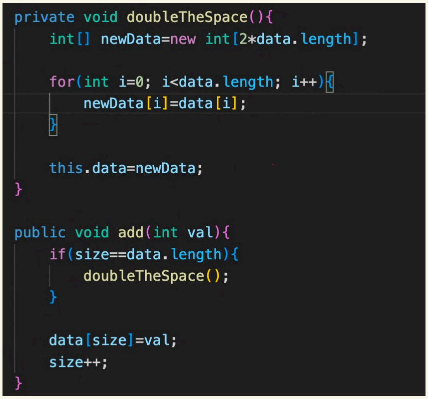


size = ~~0~~ ~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ 4 ~~3~~ 4

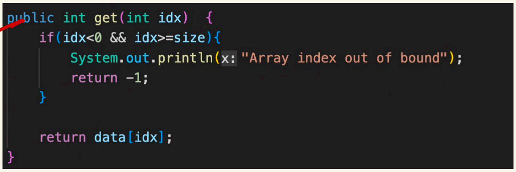


```
ArrayList ~~al=new~~ ArrayList();


for(int val=2; val<=10; val=val+2){
    al.add(val);
}

for(int i=0; i<al.size(); i++){
    int val=al.get(i);
    System.out.println(val);
}

al.set(idx: 3,val: 45);

System.out.println(al.get(idx: 3));
```

val=~~2~~ ~~4~~ ~~6~~ ~~8~~ 10



```
private void doubleTheSpace(){
    int[] newData=new int[2*data.length];

    for(int i=0; i<data.length; i++){
        newData[i]=data[i];
    }

    this.data=newData;
}

public void add(int val){
    if(size==data.length){
        doubleTheSpace();
    }

    data[size]=val;
    size++;
}
```



```
public int get(int idx)  {
    if(idx<0 && idx>=size){
        System.out.println(x: "Array index out of bound");
        return -1;
    }

    return data[idx];
}
```