C-(1)

Linear Array

it means creating

creation

(2) `Foo [] x = new Foo [100];`

// creating an array name x of Foo type
// and have a length of 100.

`Foo[] y = x;`

// Here it is shown that (you may assume)
// an new array name y type Foo is created
// but it's not it's another (Foo) type of array
// which is referenced to another array (Foo type)
// X.

Iteration

(2)

while loop

```
int i=0;
while (i<x.length) {
    S.O.P (x[i]);
    i++;
}
```

For loop

```
for (int i=0; i<x.length; i++){
    S.O.P (x[i]);
}
```

For each loop :-

```
for (Foo v : x) {
    S.O.P (v);
}
```

Foo type of x

x is the array

v is where the elements of x array been send and it catches and print the element and iteration in (1) every time. That's the limitation

Copy of an array :-

(a)

```
public static Object[] copyArray(Object[] source){
    Object[] copy = new Object[source.length];
    for(int i=0; i<source.length; i++){
        copy[i] = source[i];
    }
    return copy;
}
```

//Built in method :-

```
public static Object[] copyArray(Object[] source)
    Object[] copy = java.util.Arrays.copyOf(source, source.length);
    return copy;
}
```

8) Resizing :-

```
public static Object[] resize(Object[] oldArray,
    Object [] newArray = new Object[newCapacity]; int newCapacity)
for(int i=0 ; i<oldArray.length; i++){
        newArray[i] = oldArray[i];
}
    return newArray;
}
```

//Built in method

```
Object[] data= new Object [10];
for(int i=0; i<10; i++){
    data[i] = new String(String.valueOf(i));
}
 data = resize (data, 20);
for (int i=0; i<20; i++){
    data[i] = new String(String.valueOf(i));
}
```

→ Out of place approach

(C) Reversing an array :-

[return type void এর কেস]

```
public static void reverse (Object[] array){

    Object tempArray = new Object [array.length];

    int i=0;
    int j= tmp.Array.length-1;
    while (i< array.length) {
        tempArray [j] = array[i];
        i++;
        j--;
    }

    for (int i=0; i<array.length; i++){
        array[i] = tempArray [i];
    }
```

```
int j = source.length
        -1
for(int i=0;
    i<source.len/2;
    i++)
    int temp = source[i];
    source[i] = source[j]
    source[j] = temp;
```

i দিলে কাজ করবে না

but প্রতি iteration এ যদি k দিলে কাজ করবে।

[out of place approach মানে নতুন করে array create করে সেই array তে কাজ গুলো manipulate করা।]

Reversing an array in place approach :-

work like palindrome :- (in place approach)

```
public static void reverse (object[] array){
    int i=0;
    int j= array.length-1;
    while (i<j) {
        object temp = array[i]
        array[i] = array[j];
        array[j] = temp;
    }
    i++;
    j--;
}
```

10, 20, 30, 40, 50
i j

↳ i<j,
50, 20, 30, 40, 10
i j

↳ i<j; i++; j--;
50, 40, 30, 20, 10
i j

↳ i<j (x) → array reverse

→ one for even length

10, 20, 30, 40, 50, 60
i j

↳ i<j
60, 20, 30, 40, 50, 10,
i j

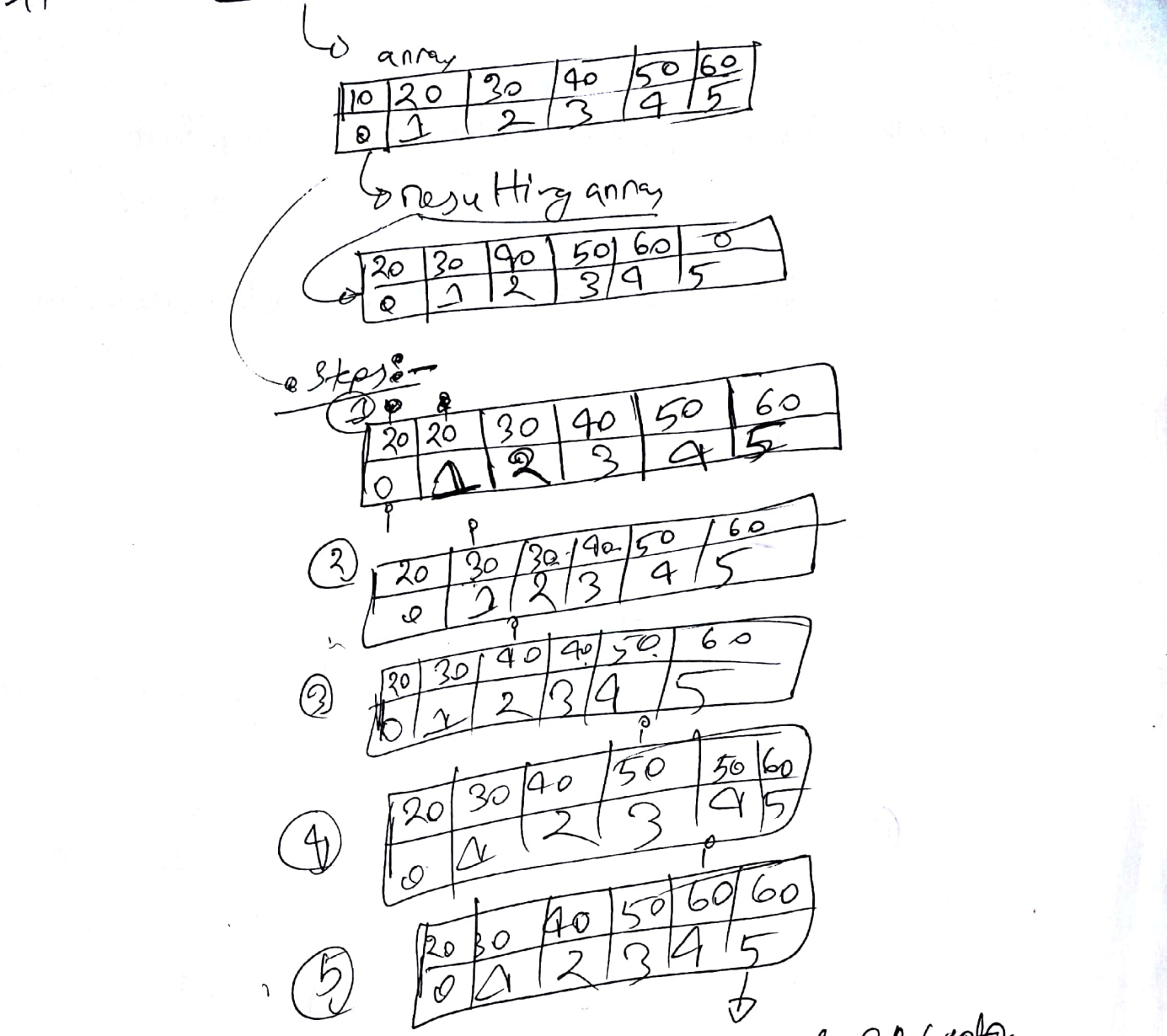
↳ i<j, i++, j--;
60, 50, 30, 40, 20, 10
i j

i<j, i++, j--;
60, 50, 40, 30, 20, 10
i j

i<j (x) → stopped

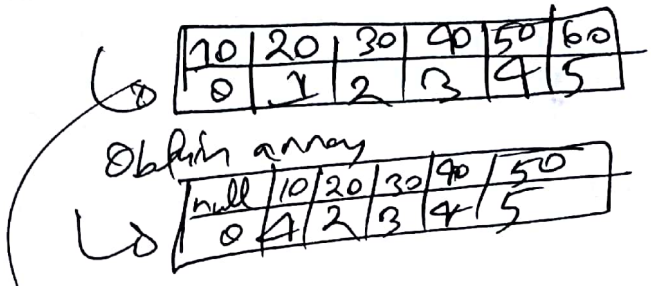
array reversed

Left shift :-




```
public static void LeftShift(Object a[]){
    for(int i=0; i<a.length-1; i++){
        a[i]=a[i+1];
    }
    a[a.length-1]=null;
}
```

② Rightshift :-



Steps :-


```
public static void rightShift(Object a[]){
    for(int i = a.length-1; i>0; i--){
        a[i] = a[i-1];
    }
    a[0] = null;
}
```

10

Right shift by K places :-

→ [এখানে value K বার বামে চলে যাবে]

10	20	30	40	50
0	1	2	3	4

i

Let k=2

→ output would be

0	4	3	3	4

10

Left shift by K places

K=3

10	20	30	40	50	60
0	1	2	3	4	5

↳

30	40	50	0	0	0
0	1	2	3	4	5

zero পাওয়া যাবে

public ~~static~~ int[] shiftLeft(int[] source, int k){

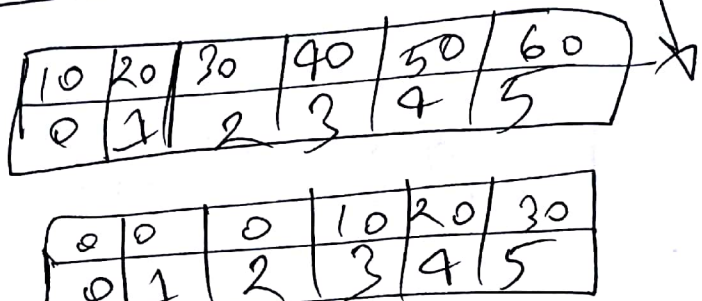
for(int j=0; j<k; j++){

~~source~~

```
for(int i=0; i<source.length-1; i++){
    source[i] = source[i+1];
}
source[source.length-1] = 0;
```

11

Right shift by K places :-



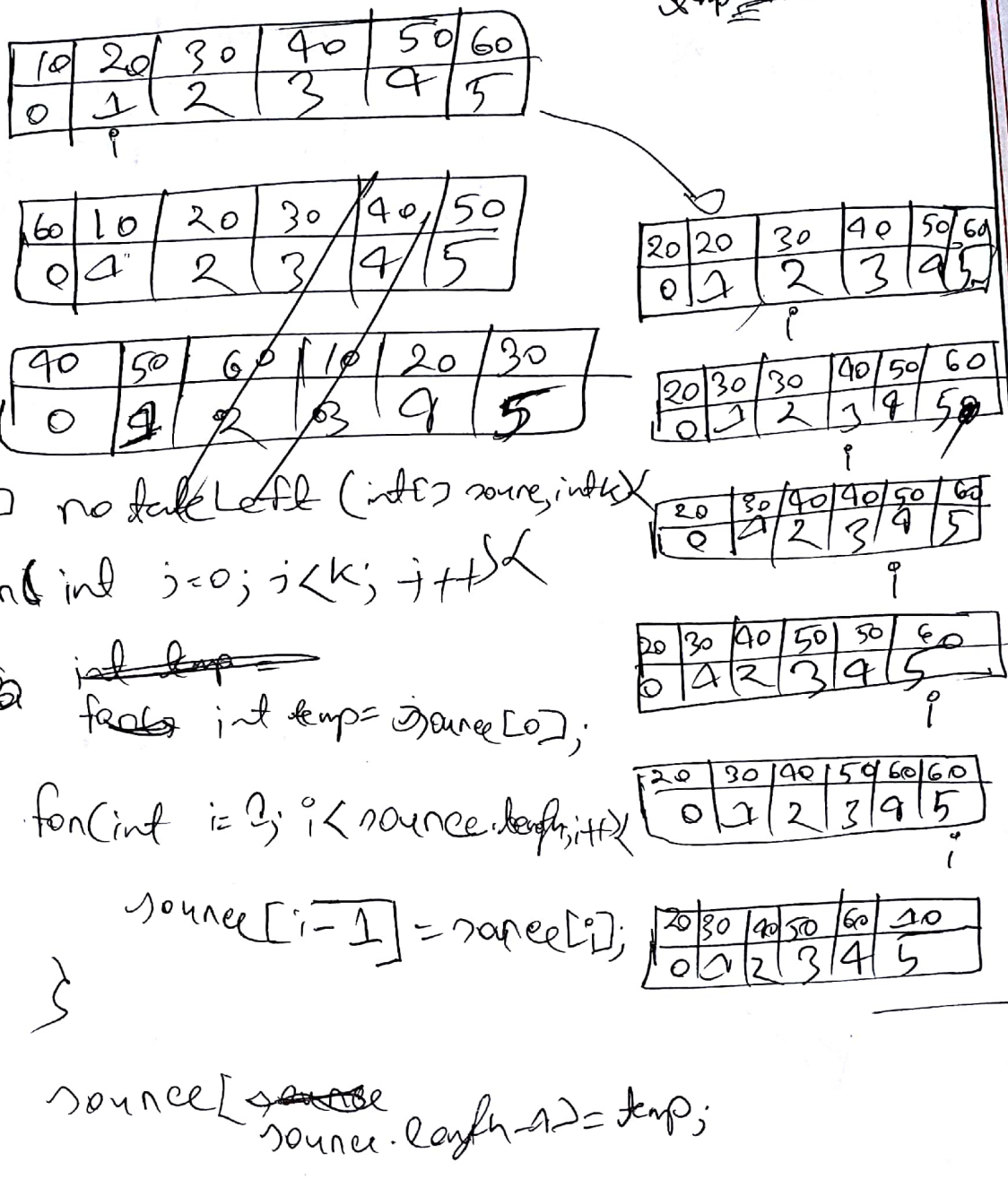
last index

```
int[] shiftRight(int[] source, int k){
for(int j=0; j<k; j++){
for(int i= source.length-1; i> 0; i--){
source[i] = source[i-1];
}
source[0]= 0;
}
return source;
}
```

Scanned by CamScanner

13

Rotate Left

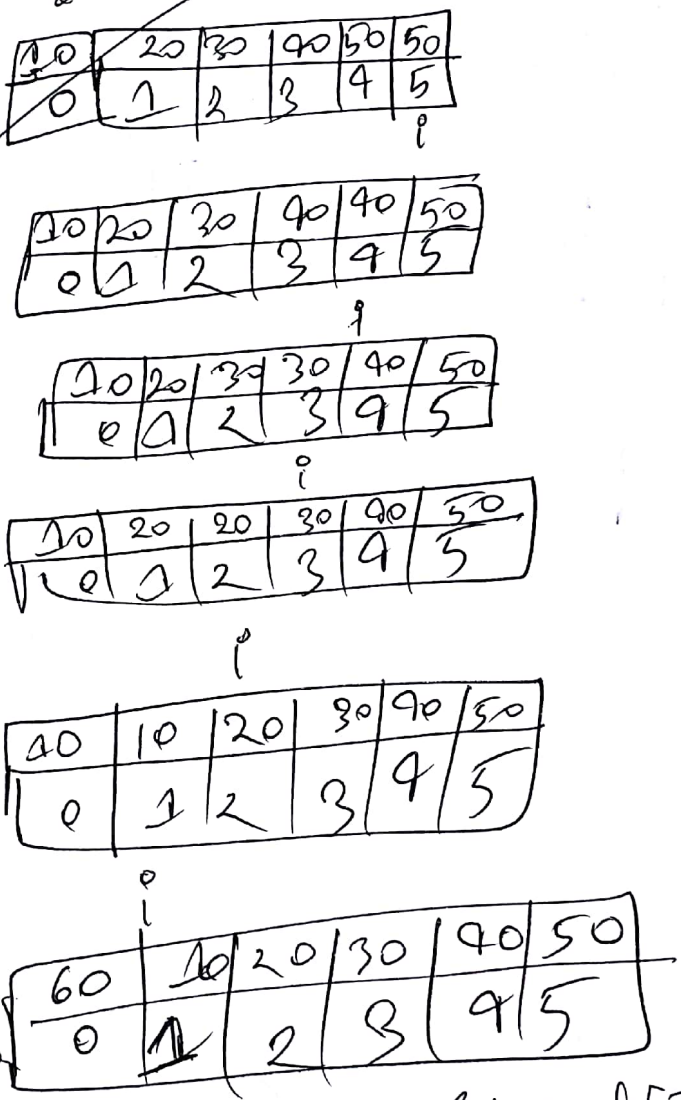


```
int[] rotateLeft(int[] source, int k){
    for(int j=0; j<k; j++){
        int temp = source[0];
        for(int i=1; i<source.length; i++){
            source[i-1] = source[i];
        }
        source[source.length-1] = temp;
    }
    return source;
}
```

13 Rotate Right

10	20	30	40	50	60
0	1	2	3	4	5

source[source.length-1] = temp ; // 60




```
int[] rotateright(int[] source, int k){
for(int j=0; j<k; j++){
int temp = source[source.length-1];
for(int i=source.length-2; i>0; i--){
source[i] = source[i-1];
}source[0] = temp;
```

14

Inserting an element to an array

10	20	30	40	50	0
0	1	2	3	4	5

→ size = 5
i=

insert 60 at index 3

(i)

10	20	30	40	50	50
0	1	2	3	4	5

(ii)

10	20	30	40	40	50
0	1	2	3	4	5

shifting [illegible]
i > index পর্যন্ত
a[i-1] এর value
i তে [illegible]

(iii)

10	20	30	60	40	50
0	1	2	3	4	5

→ insert

```
boolean insert (int[] arr, int size, int elem, int index){
    boolean flag = false;
    if(size==arr.length){
        s.o.p("No space left");
        return flag
    }
    else{
        if(index<0 || index>size){
            s.o.p("invalid index");
            return flag;
        }
        else{ for(int i=size; i>index; i--){
            arr[i]=arr[i-1];}
            arr[index] = elem;
            flag = true;
            return flag
        }
    }
```

Remove 30

(arr[i] == elem) এমন

10	20	30	40	30	50
0	1	2	3	4	5

i = 0

newArr

0	0	0	0	0	0
0	1	2	3	4	5

j = 0

10	0	0	0	0	0
0	1	2	3	4	5

j = 1

10	20	0	0	0	0
0	1	2	3	4	5

j = 2

10	20	40	0	0	0
0	1	2	3	4	5

j = 3

10	20	40	60	0	0
0	1	2	3	4	5

value এবার assign হবে

Then j++

10	20	30	40	50	60
0	1	2	3	4	5

i

10	20	30	40	30	60
0	1	2	3	4	5

i

10	20	30	40	30	60
0	1	2	3	4	5

i

10	20	30	40	30	60
0	1	2	3	4	5

i

10	30	30	40	30	60
0	1	2	3	4	5

i

এবার remove হওয়া element
এর array index দাও।
যেহেতু new Array তে শুধু element
গুলো copy করব।

arr

10	20	40	60	0	0
0	1	2	3	4	5

code removeAll

```
boolean removeAll(int[] arr, int size, int elem){
    boolean flag = false;
    if(size == 0){
        s.o.p("nothing to remove");
        return flag;
    }
    else{
        int[] newArray = new int[arr.length];
        int j = 0;
        for(int i=0; i<arr.length; i++){
            if(arr[i] == elem){
                flag = true;
                continue;
            }
            else{
                newArray[j] = arr[i];
                j++;
            }
        }
        for(int i=0; i<arr.length; i++){
            arr[i] = newArray[i];
        }
    }
    return flag;
}
```

নতুন [illegible]

এর মাধ্যমে remove করা element বাদে বাকি সব copy হলো নতুন Array তে।

// [illegible]

কেননা remove হওয়ার পরে যে নতুন Array [illegible] copy করবো। Array তে copy করবো।