Swap w/o third variable.

$x = 25 \rightarrow 23$

$y = 23 \rightarrow 25$

$x = x + y \Rightarrow 25 + 23 = \underline{48} \rightarrow x$

$y = x - y \Rightarrow 48 - 23 = 25 \rightarrow y$] — o/p

$x = x - y = 48 - 25 = 23 \rightarrow x$

$x = 23, \quad y = 25.$

Reverse a 3 digit no.

$rn = 0$

$n = 3\,4\,2$

$100 \quad 10 \quad 1$

$2 \quad 4 \quad 3$

$digit = n \% 10 = 2$

$rn \times 10 + digit$

$rn = 0 \times 10 + 2 = \underline{2}$

$= n/10 \rightarrow 34$

$= 34 \% 10 \rightarrow 4$

$rn = 2 \times 10 + 4 = 24$

$= n/10 = 34/10 = 3$

$= n \% 10 = 3 \% 10 = 3$

$rn = 24 \times 10 + 3 = \underline{243}$

```
// take input

Syso(reverse(n));

}

public static int reverse(int num){
    int rn = 0;
    while(num > 0){
        int digit = num % 10;
        rn = rn * 10 + digit;
        num = num / 10;
    }
    return rn;
}
```

$TC \rightarrow O(n)$

Reverse n- numbers.

TC → O(n+n)

```java
    // input
    Scanner s = new Scanner(System.in);
    int n = s.nextInt();

    int number = formNumber(n, s);
    System.out.println(number);
    System.out.println(reverse(number));

}

    public static int formNumber(int n, Scanner s){
        int number = 0;

        for(int i = 0; i < n; i++){
            int digit = s.nextInt();
            number = number * 10 + digit;
        }
        return number;
    }

    public static int reverse(int number){
        int rv = 0;

        while(number != 0){
            int digit = number % 10;
            rv = rv * 10 + digit;
            number /= 10;
        }
        return rv;
    }
```

n

n

HW_Check Palindrome

TC = O(n)

```
    Scanner s =
        itn t =
        for(){
            int num = s.nextInt();
            if(isPalindrome(num)){
                Syso("YES");
            }
            else{
                Syso("NO");
            }
        }

}

public static int reverse(int number){

}

public static boolean isPalindrome(int number){
    return number == reverse(number);
}
```

— n

O(1)

123456 1234,56

% 10 → 6

last digit ✓ % 100 → 56

% 1000 → 456

% 10000 → 3456

/10 → 12345

first digits ✓ /100 → 1234

/1000 → 123

num = last digit × 10000 + first digit;

return num;

560000
+ 1234
561234

Transform a no.

Arrays → Collection of same data types.

int arr

1	2	3	4

4000 4004 4008 4012

int - 4 byts

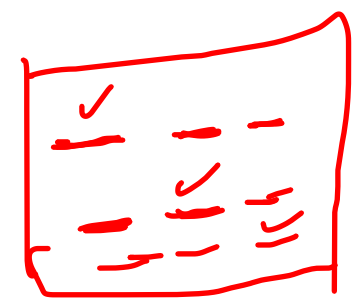
1 - 4 byts

4000 - 4003

5 × 4 = 20 byts

size of array = datatype × no. of elements

= 4 × 4 = 16 bytes.



arrays are always stored continuously acc. to data type.

Array indexing starts from 0.

length of array → 5

arr →

1	2	3	4	5
0	1	2	3	4

function = .length

arr.length; // → 5

index of array → n-1

Declaration of array

int arr[] = new int[10];

- int → type of array
- arr[] → name of array
- [] → array representation
- int → data type
- 10 → array size
- new → brings continuous memory for array.

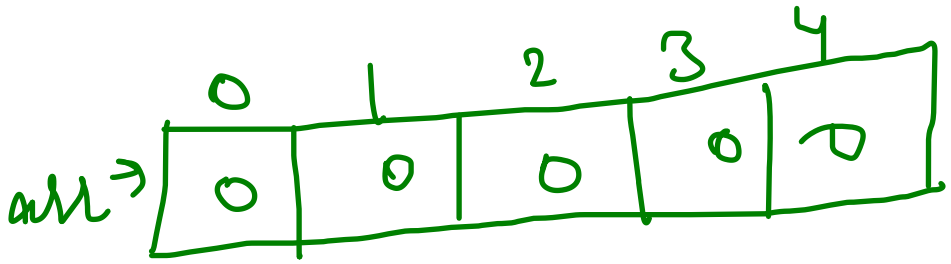
name[] ✓
[] name[] ✓

RHS = 40 bytes of continuous memory.
LHS = type and name of array.

char ch[] = new char[10]; → character type
double [] d = new double [10]; → decimal number.
boolean b[] = new boolean[10]; → boolean type.

int arr[] = new int[5];

arr[6] → error




```
for( int i = 0; i < arr.length; i++){
    arr[i] = s.nextInt();
}
System.out.print(arr[i]);
```

```
// Use this editor to write, compile and run your Java
import java.util.Scanner;
class Main {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);

        int arr[] = new int[5];

        // take input
        for(int i = 0; i < arr.length; i++){
            arr[i] = s.nextInt();
        }

        // print output
        for(int i =0; i < arr.length; i++){
        System.out.print(arr[i] + " ");
        }
    }
}
```

Output

```
1
2
3
4
5
1 2 3 4 5
=== Code Execution Success
```

Arrays works on <u>references</u>

* If we try to access an "invalid" index, then an error out of bound index.

* Indexing from [0 - (n - 1)]

* Indexing can't be negative.

* By default all the indexes of the array are 0 (int)

* Double → 0.0 (by default)

* char array → null

* boolean array → false.

How data stored in arrays.

Datatypes

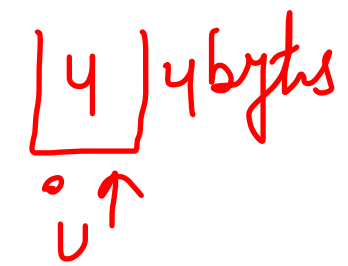
primitive
- → int
- → char
- → boolean
- → double

etc ..

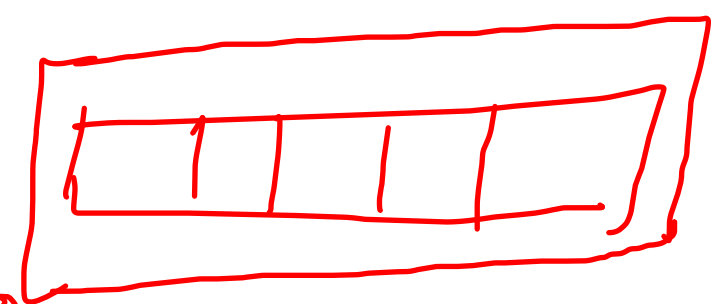
Non-primitive
- → arrays
- → String
- → Scanner

etc ..

int i = 4;



int arr[] = new int[5];



arr → is reference. not direct name.

Class 10, → 2nd bench

arr → address of array.

[I@42a72fb31

array

int

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

System.out.println(arr);
System.out.println(arr[4]);
System.out.println(2);
```

Output

```
[I@659e0bfd
0
2

=== Code Execut
```

Class 10 → arr

1 —
2 —
3 —
4 —

0	1	2	3
1	2	3	4

arr[i] % 2 == 0

sysoo(i);

[1, 3]

arr → address

arr[i] → element present at index i

i → index number.