1) Comparators

2) Problems on Sorting using comparators.

No ass./H.W. ⇒ follow up.

< 70 ??

24th & 2nd ⇒ Break.

23rd & 24th ⇒ 5 problems/day.

25th Chr → +5 → 10

(26-30) 2-3 ⇒ 10-15 problems

31st & 1st ⇒ 5 problems (31st)

20-25 quesn

1st June 2024 ⇒ 7-10

32-40 problems

10 classes

Monday (18th) ⇒ Binary Search

50 % DSA Back Greedy > DSA2

↓

DSA 3 & 4

~~Backlog~~

1) Attended lectures → Pending problems.

2) Pending lectures → Recording → 2X 1.75X

↓

Assignment.

3) Additional problems

4) List of Questions on completed topics

TBS ⇒ Ayush.

DSA Contest 2 →

R2 17th Dec 11:59 PM

↓

Comparables

class Student {

String name;
double age;
double psp;
int yoe;

.....

}

Asc

List<Student> students

Arrays.sort(students);

Comparable interface.

int compareTo(T other)

class Student implements Comparable<Student> {

String name;
Integer age;

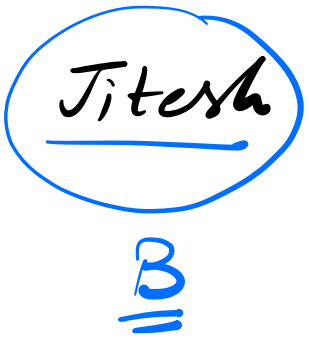
```
double psp;
int yoe;

int compareTo(Student other) {
    return Integer.compare(this.age, other.age);
}
```

A B

Natural Ordering / Logic of sorry

```
if (A > B) {
    return 1;
} else if (A < B) {
    return -1;
}
else {   // A == B
    return 0;
}
```



int Vivek.compareTo(Jitesh) {

1 ⇒ Vivek is Elder

-1 Jitesh is elder

0 Both are of equal age.

Arrays.sort (students)

int A[N] ⇒ Value

Custom Comparator

function

C++ Python Java

C++

```
bool compare(int A, int B) {
```

If in sorted order, A should come before B

```
    return true;
```

If B should come before A
return false;

}

Java, Python, JS, C#, Ruby

int compare (int A, int B) {

If in sorted order, A should come before B
return -ve value (-1);

If B should come before A
return +ve value (1);

If both are equal
return 0;

}

Given an integer array of size N.
Sort the data in asending order of count of factors.

A = [9, 3, 10, 6, 4]
(0) (1) (2) (3) (4)
(3, 2, 4, 4, 3)

O/P: [3, 9, 4, 10, 6]
(2) (3) (3) (4) (4)

NO: [3, 4, 6, 9, 10]

Collections.sort(A);
Arrays.sort(A);
A.sort(); } Natural order of integers

When do we want A to come before B??
count of factors of A < count of factors of B.

```
int compare(int A, int B)

    int factorsA = getFactors(A);
    int factorsB = getFactors(B);

    if (factorsA < factorsB)
        return -1;
    else if (factorsB < factorsA)
```

```
            return 1;
    }
    else {
            return 0;
    }
}
```

---

int A, int B

char A char B

String A string B

ayush

mayo

cat — > lexicographical order
cattle —

Given an integer array of size N.

Sort the data in asending order of count of factors.

If count of factors are equal, sort them on the basis of their actual value

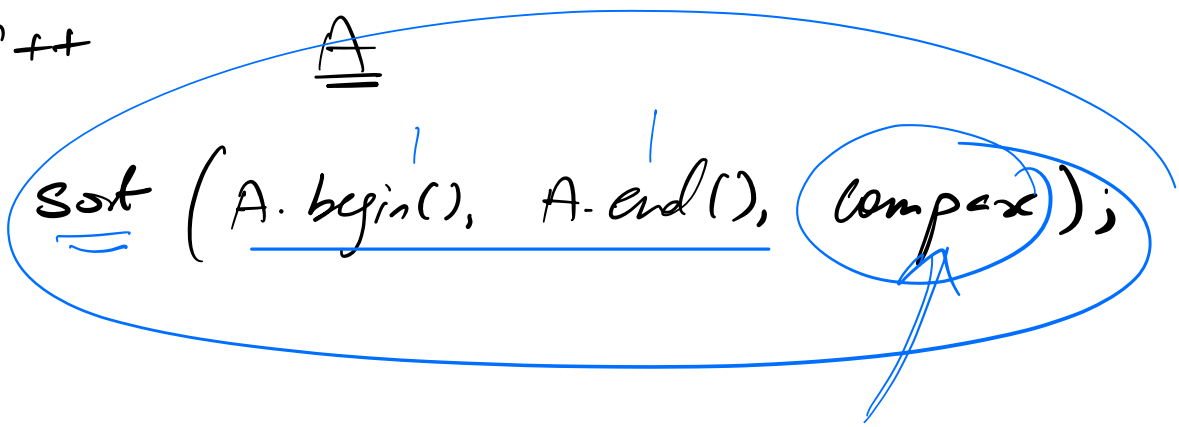
$$A = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 9, & 3, & 10, & 6, & 4 \\ 3, & 2, & 4, & 4, & 3 \end{bmatrix}$$

$$O/P : \begin{bmatrix} 3, & 4, & 9, & 6, & 10 \\ 2 & 3 & 3 & 4 & 4 \end{bmatrix}$$

```
int compare( int A, int B) {
    int factorsA = getFactors(A);
    int factorsB = getFactors(B);
    if (factorsA < factorsB) {
        return -1;
    }
    else if (factorsB < factorsA) {
```

```
        return 1;
    }
    else {
        if (A < B) {
            return -1;
        } else if (B < A) {
            return 1;
        }
        else {
            return 0;
        }
    }
}
```

C++



Python

```
A = sorted(A, key = functools.cmp_to_key(comparx))
```

---

class factorSort implements Comparator<int>{

```
public int compare (int A, int B) {
    int factorsA = getFactors(A);
    int factorsB = getFactors(B);
    if (factorsA < factorsB) {
        return -1;
    }
    else if (factorsB < factorsA) {
        return 1;
    }
    else {
        if (A < B) {
            return -1;
        } else if (B < A) {
            return 1;
        }
        else {
            return 0;
        }
    }
}
```

Arrays.Sort(A, _____)

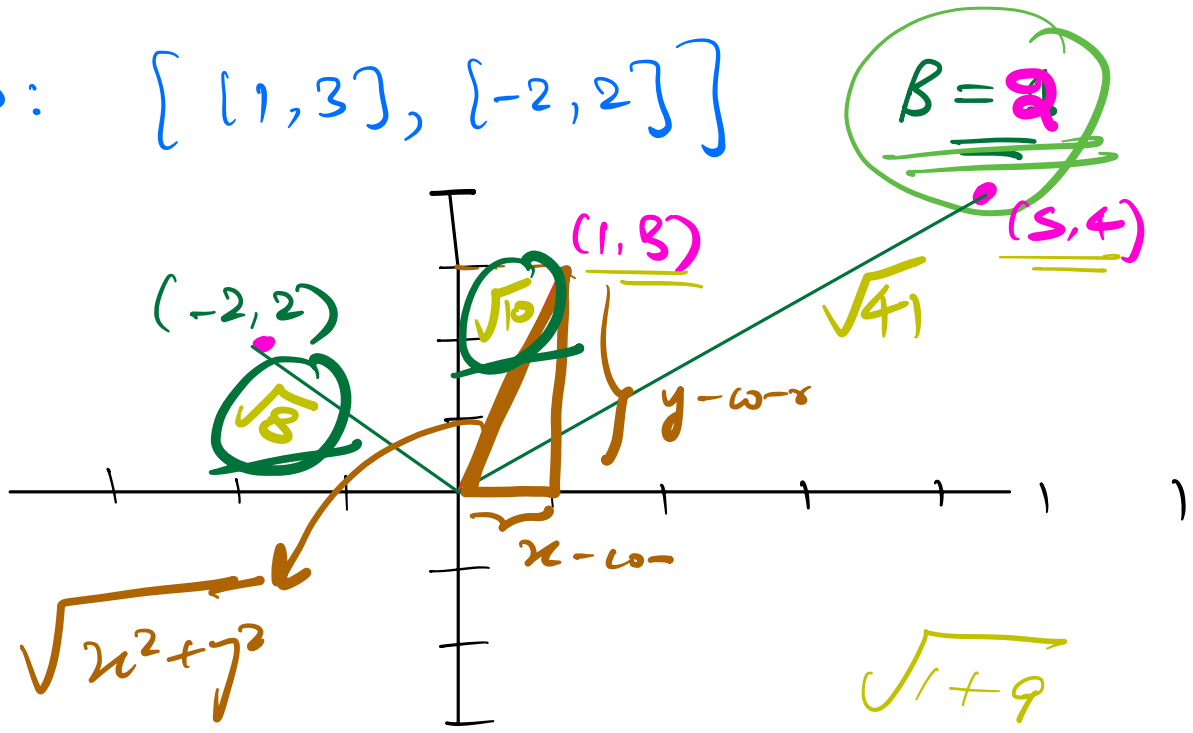
Collections.sort(A, new factorsSort());

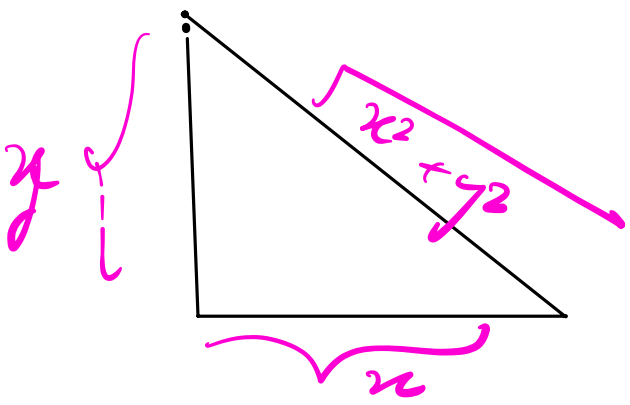
Q Given an array of points.

points[i] = [xi, yi]

Return B closest points to origin.

I/p: [[1,3], [-2,2]]





$\sqrt{1+9}$

$\sqrt{4+4}$

$= 2\sqrt{2}$

$\sqrt{25+16}$

Sol^n ⇒ Sort all points on the basis of distance from origin

Sort B elements ⇒ ans

Given a list of non negative nos.

Arrange them in such a way that they form the largest number.

Return the no

A = [3, 30, 34, 5, 9]
0 1 2 3 4

⇒ [9, 5, 34, 3, 30]

9534330

9534303

3 30 34 5 9

95433

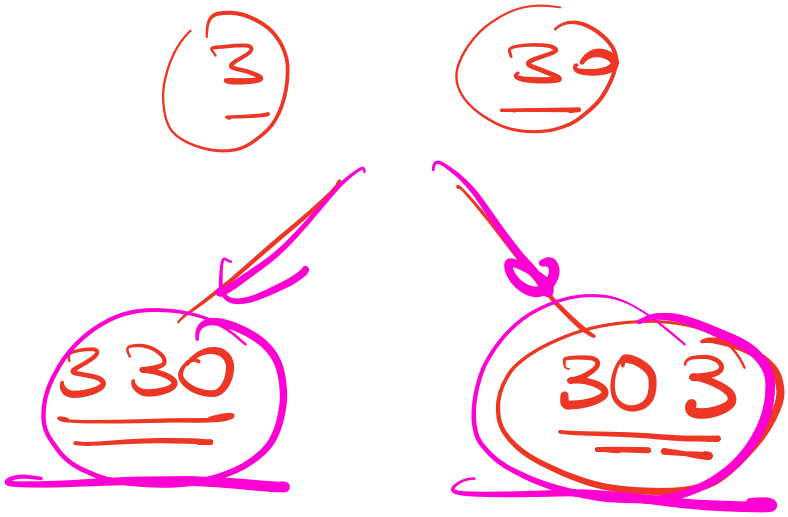
95033

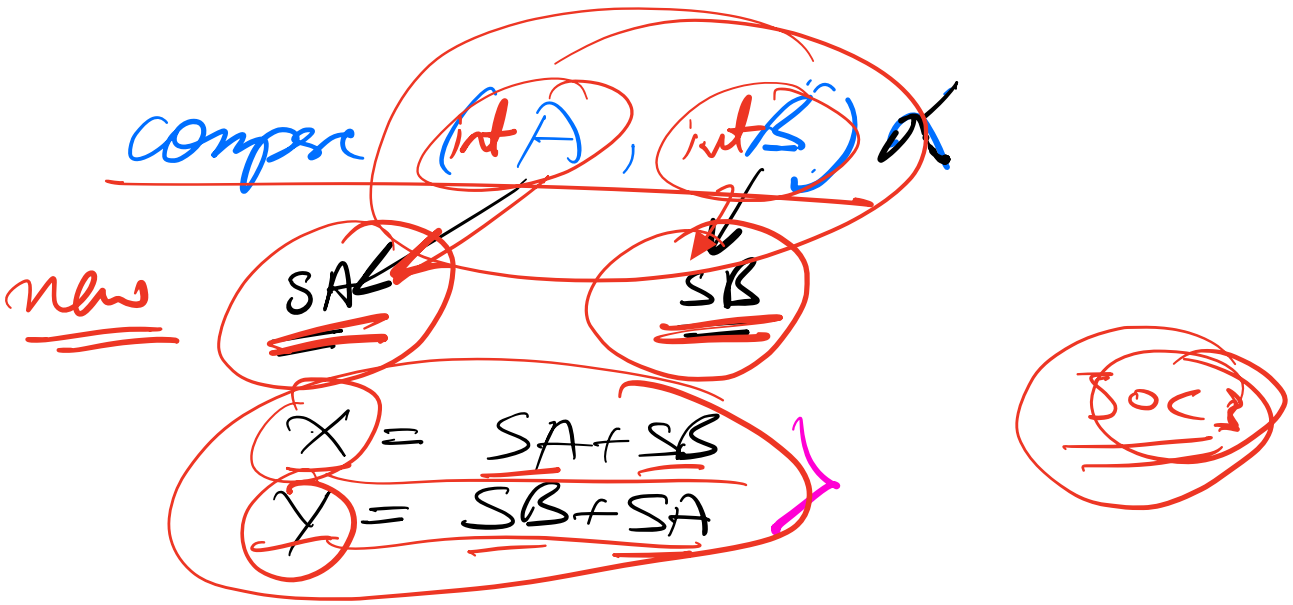
95, 8

30 3

958

3432, 521, 9435536, 99.

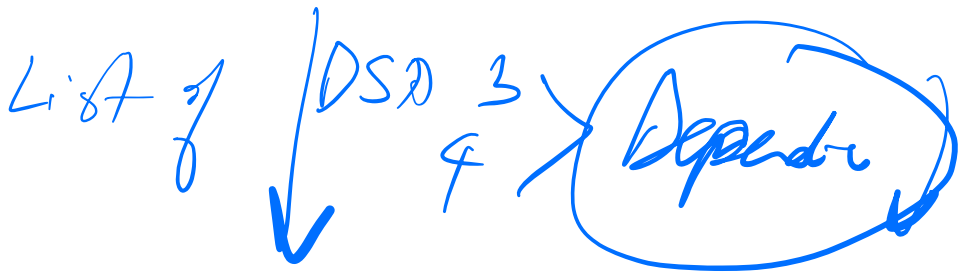




-1
1
0

4


H/SA
CS
L N
Ready


List of
DSD 3
Dependency