Problem Solving - 3

Question: Maximum Unsorted Subarray

Array of Integers,

Find Min Subarray s.t if we sort this subarray, the whole array gets sorted

Ex:

A =	1	4	2	3	5	6
	0	1	2	3	4	5

$\boxed{[1, 3]}$

A' = 1 2 3 4 5 6

(Smallest Subarray)

[1, 5]

A'' = 1 2 3 4 5 6

Ex:

A =	1	3	2	4	5
	0	1	2	3	4

$\boxed{[1, 2]}$

A' = 1 2 3 4 5

[0, 2]

$\boxed{\text{Ex}}$

0	1	2	3	4
2	4	1	5	6

1 2 4 5 6

(n)

Brute Force:

(L, R)

Consider all subarrays $O(n^2)$

→ Sort this subarray.
→ check the whole array is sorted
→ Find min length subarray

T.C: $\underbrace{n^2}_{} \times [\underbrace{n\log n}_{} + n]$

$\boxed{\text{T.C: } O(n^3 \log n)}$

Optimised Approach:

A =	1	4	3	2	5	6
A' =	1	2	3	4	5	6

Duplicating

→ $\boxed{[2, 4]}$

O(nlogn) ← 1) Sort the array
O(n) ← 2) Find first Mismatch from start (i)
O(n) ← 3) Find first mismatch from last (j)

$\boxed{[i, j]}$

$\boxed{\begin{array}{l} \text{T.C: } O(n\log n) \\ \text{S.C: } O(n) \end{array}}$

	0	1	2	3	4	5	6
Ex: A =	1	10	9	5	2	3	20
A =	1	2	3	5	9	10	20

[1, 5]

2 - Pointer ⇒ T.C: O(n)
S.C: O(1)

	0	1	2	3	4
A =	2	1	4	6	5

[0, 4]

Question:

Given some Info, construct an array of size k

1) Array should be sorted in ↑
2) All elements are distinct & positive (≥ 1)

d = 0 ✗

3) → Difference between 2 consecutive elements should be same (A.P)

$a_1 \; a_2 \; a_3 \; a_4 \; a_5 \ldots$

4) x, y should be there in the array

Objective: The last element of array is as minimum as possible

[x < y]

k = 5, x = 20, y = 50

A1 = 20 50 80 110 140

a = 20
d = 30

A' = 10 20 30 40 50 → Y

60 $A'' =$ 20 30 40 80 60 $\boxed{60}$

Ex2: $K = 4$, $x = 20$, $y = 40$

$A_1 =$ 20 40 60 80

$A_2 =$ 10 20 30 $\boxed{40}$ y

Ex3: $K = 4$, $x = 2$ $y = 3$

$A =$ 1 2 3 4

Brute Force:

Consider all possible A.P and choose the one with min last element

AP: a, a+d, a+2d, a+3d . . . . . $\boxed{a+(n-1)d}$

1st 2nd 3rd 4th

n terms

kth $a+(k-1)d$ $a+(k-1)d$

a : Starting term
d : common difference

Range of a : $[1, x]$

≥ 1

Sorted array

[. . . . x | . . y . . . .]

$(x < y)$

$(x < y)$

Range of d :

$[1, y-x]$

$[a_1 \; a_2 \; a_3 \; x \; a_4 \; a_5 \; a_6 \; y \; a_7 \; a_8]$

d

(ud)

$d < (y-x)$

$d < (y-x)$

k = 5

for(i=0; i<k; i++)

$[a_1 \; a_2 \; a_3 \; x \; y \; a_4 \; a_5 \; a_6 \ldots]$

$d = (y-x)$

a = 2
d = 3

2 5 8 11 14

(x y)

$d = y-x$

$d > y-x$?

a : [1, x]

d : [1, y-x]

K

```
ans;
min_so_far = INT_MAX;
{ for( a = 1; a ≤ x; a++) {
    for (d = 1; d ≤ y - x; d++) {
        for( i = 0; i < k; i++)
            temp[i] = a + i.d;

        if( X is there && Y is there &&
            temp[k-1] < min_so_far) {
            min_so_far = temp[k-1];
            ans = temp;
    }
}
}
```

T.C : O(X . (Y) . K)

T.C : O(X . Y . K)

Accepted

$1 \le x, y, k \le 50$

$50 \times 50 \times 50 = 125000$

50×50... $\sqrt{1.2\times10^{9}}$

Efficient Approach:

AP: $a \quad a+d \quad a+2d \quad a+3d \quad a+4d \ldots$

(x above $a+2d$, Y above $a+4d$)

$x = a + P_1 \cdot d$ — P_1 is integer

$Y = a + P_2 \cdot d$ — P_2 is integer

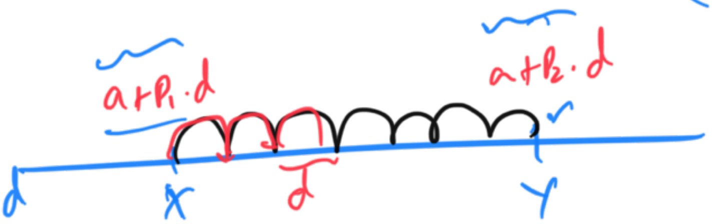
$P_1, P_2 \geq 0$

$(P_1 < P_2)$

$(Y-x) = (P_2 - P_1)\cdot d$

d has to be a factor of $(Y-x)$ (divisor)

$(P_2 - P_1)$



$(P_2 - P_1)$ hops

$a+P_1 d + (P_2-P_1)d$
$= a + P_2 d$

d

$x = 3 \quad Y = 9$ $d=3$

3 6 9 12 15 18 ...

$a+0\cdot d \quad a+2\cdot d$

$(2-0)$

(P_2-P_1) hops

$Y, x, d \ldots$

$(P_2 - P_1 + 1)$

No. of ele $= (P_2 - P_1 + 1)$

$= \left(\frac{Y-x}{d}\right) + 1$

$x = 3, \quad Y = 9, \quad d = 2$

1 3 5 7 9 11 13

$a+d$ $a+ud$

$(4-1)$

$\frac{Y-X}{d} = \frac{9-3}{2} = \frac{6}{3} = 3$

3+1 =

No. of terms $= \left(\frac{Y-X}{d}\right) + 1$

Factor of (y-x)

$\boxed{k = 6}$ $x = 3$ $Y = 9$

$(Y-X) = 6$

$\boxed{d = 1, 2, 3, 6}$

7 terms

3 4 5 6 7 8 9 . . .

X

d = 1

No. of terms : $\left(\frac{Y-X}{d}\right) + 1$

$= \left(\frac{9-3}{1}\right) + 1 = \boxed{7}$

1

k = 6, x = 3, Y = 9

d = 2

1 3 5 7 9 11 . . .

No of terms : $\left(\frac{9-3}{2}\right) + 1$

$= 3 + 1 = \boxed{4}$

remaining_terms = $6 - 4 = \boxed{2}$

= 1

ans

```
for(i=0; i<k; i++)
ans[i] = temp[i]
```

[. . . X . . . Y . . .]

Y to be as right as possible

$d = 3$ $x = 3,\ Y = 9,\ k = 6$

3 6 9 . 12 15 18

No. of terms $= \left\lfloor \frac{9-3}{3} \right\rfloor + 1$

$= \boxed{3}$

remaining $= \boxed{3}$

$d = 6$

3 9 15 21 27 33

No. $\left\lfloor \frac{9-3}{6} \right\rfloor + 1 = \boxed{2}$

remaining $= \boxed{4}$

T.C: $O(f \times K)$
size of array

Factors

20: 1 2 4 5 10 20

$20 \% f == 0$

```
vector<int> factors;
for( i = 1 ; i <= N; i++) {
    if( n % i == 0) {
        factors.push_back (i);
    }
}
```

a, d

T.C: O(n)

Factors in sorted order.

{a, b}

$\frac{n}{P_1} = P_2$

$\frac{P}{P_2} = P_1$

n 4 25 $\frac{n}{4} = x$

$\frac{n}{x} = y$

$\{x, y\}$ both are factors

1) $x < y$

$n/2$

2) $x = y$ $\sqrt{12} = 3.4$

12 = 1 2 3 4 6 12

$(\sqrt{n})$

16 = 1 2 4 8 16

$$\boxed{\begin{array}{l} a \cdot b = N \\ a \le \sqrt{N} \\ b \ge \sqrt{N} \end{array}}$$

$(i * i \le N)$
$i \le sqrt(N)$

$\boxed{sqrt(n)} \Rightarrow \boxed{\log n}$

$\boxed{(\log n)}$

```
for(i=1; i*i <= N; i++){
    if(N % i == 0){
        factors.push(i);
        factors.push(n/i);
    }
}
```

$$\boxed{T.C: \; O(\sqrt{N})}$$

$O(d \log d)$

factors are not sorted

$X = \alpha + P_1 \cdot d$

$Y = \alpha + P_2 \cdot d$

$Y - X = (P_2 - P_1)\, d$

$\boxed{Y = X + (P_2 - P_1)\, d}$

$(Y - X =) \quad (P_2 - P_1)\, d$

Y/d

(Integer)

$Z = X \cdot Y$

Question: Given 'M' puzzles each has certain no. of pieces

→ choose 'N' puzzles such that

if A = Max no. of pieces in puzzle

B = Min no. of pieces in puzzle

(A − B) is minimized

(M ≥ 8)

	0	1	2	3	4	5
A =	10	15	6	9	13	8

M = 6, N = 4 (Input)

N = 4

A1 = {6, 15, 13, 9} → A = 15, B = 6 } = 9

|15 − 6| = $\boxed{9}$ minimum so

= {13, 8, 6, 9} A = 13, B = 6 13 − 6 = $\boxed{7}$

A = 13 $\boxed{5}$

A' = {13, 9, 8, 10}   B = 8
                                  A = 10

Ans = {6, 8, 9, 10}   B = 6 $\boxed{4}$

Suggested Approach 1

sort Array & take first N ele

N = 3

Eg 1   100   101   102   103

[1, 100, 101]   [101, 102, 103]
A - B   A - B
101 - 1 = 100   103 - 101 = 2

take last N element

N = 3

1   2   3   100   101

A = 3   A = 101, B = 3
B = 1   Δ = 98
Δ = 2

Brute Force:

Among M ele, choose N

$\binom{M}{N}$ → Exponential

$O(^{M}C_N \times M)$

(Back Tracking)

... Approach :

Efficient Approach

A = 10 15 6 9 13 8

M = 6, N = 4

(MAX) (Minimum A − B)

A = 10, B = 6

{10, 6, 8, 9}

A = 15, {8, 15, 9, 10}

→ {9, 15, 10, 13}

A − B = 15 − 8 = 7

A = 15
B = 9
15 − 9 = $\boxed{6}$

Sorting:

M = 6, N = 4

A =	6	8	9	10	13	15
	0	1	2	3	4	5

{6, 10, 13, 15} = 9

diff = ~~6~~ ~~5~~ 4

A = 15, B = 9

A = 13, B = 8

A = 10, B = 6 ⇒ B = 4

(i + N − 1)
0 + 4

M = 6
N = 4

M − N
6 − 4

6	8	9	10	13	15
0	1	2	3	4	5

i

9 10 13 15

Algo:

Sort Array

1) sort
2) sliding window diff = INT_MAX

for(i = 0; i <= M-N; i++){
    Δ = arr[i+N-1] - arr[i];
    diff = min(Δ, diff).
}

i+N-1

$O(m\log m + M) \Rightarrow \boxed{O(M\log m)}$ 50%

Question:

Split the array into 2 non-empty subsets such that

max(1 subset) - min(other subset) is **minimum**

M

{6, 4} {1, 2, 3}
(6 - 1) = $\boxed{5}$

Ex: 3 1 2 6 4

{3, 6} {1, 2, 4}
|6 - 1| = $\boxed{5}$

{3, 4} {1, 2, 6}
|4 - 1| = 3

4

S1 S2

{1} {2, 3, 6, 4}
|1 - 2| = $\boxed{1}$

{2} {1, 3, 6, 4} = 1

{1, 2} = $\boxed{1}$

1) max(S1) min(S2)
2) max(S2), min(S1)

[4, 1] = $\boxed{3}$

Suggested Approach:

Sort & take 1st 2 elements.

1)

[1 | 3 4 5 6]

(3 - 1) = $\boxed{2}$

{1, 3, 4} {5, 6}

2) Sort & take last 2 elements

→ 1 2 3 4 6

6 - 4 = 2

{1, 2, 3} {4, 6} = 1

{1, 3, 5} {6, 9}

1 3 5 6 9

{1, 9} {3, 5, 6}

$\boxed{2}$

(x, y)

(x - y)

A = 12 4 7 5 2 10

Sort Arr 2 4 | 5 7 10 12

max = 4

min = 5

Observation

1) The min difference between any 2 elements will have to be between any 2 consecutive elements

2 4 | 5 7 10 12

diff = ~~2~~ 1

(Break)

Approach

1) Sort the array

2) Iterate and find pair with min diff

10 33 | 37 45 97

$\Delta = 4$

✓ $T.C: O(n\log n + n) \Rightarrow O(n\log n)$

$S.C: O(1)$

Question:

Array of length 'n' with 0's and 1's

→ Return max length subarray with equal no. of 0's and 1's.

Ex

1	0	1	0	1	1	0
0	1	2	3	4	5	6

Brute Force:

consider all subarray

$O(n^2) \times n = O(n^3)$

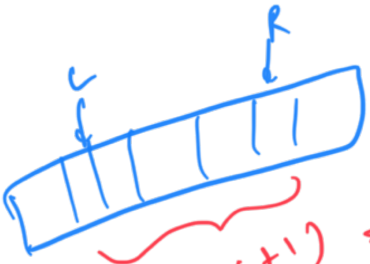
$T.C: O(n^3)$

S.C : O(1)

Approach 2: prefix Sum

$\boxed{\text{length}} = \boxed{x}$ $\quad$ $\text{sum} = \left(\frac{x}{2}\right)$

$O\left(\underset{\text{subarray}}{n^2} \times O(1)\right) =$ $\boxed{\begin{array}{l} T.C: O(n^2) \\ S.C: O(n) \end{array}}$



$L = \frac{(R-L+1)}{2} =$ sum

$\boxed{\left(\frac{R-L+1}{2}\right) == (pre[R] - pre[L-1])}$

(L, R) $O(1)$

$pre[R] - pre[L-1]$

Efficient:

	0	1	2	3	4	5	6	7	8
A =	1	0	1	0	1	1	0	0	1

Replace 0's with -1's

A = 1 -1 1 -1 1 1 -1 -1 1

Largest Subarray with sum = 0

Equal -1's & Equal 1's

Hashing Lecture

$\boxed{\begin{array}{l} T.C: O(n) \\ S.C: O(n) \end{array}}$

0 1 2 3 4 5 6 7

A = 1 0 1 0 1 1 0 0

pre : 1 1 2 2 3 4 4 4

No of 0's = 4 x

1's = 4 y

2 * min( )

4

A =

min(x, y) * 2

A =

T.C : $O(n^2)$

1 0 1 1 1 0 1 1

1 0 1 1 1 1 0 1 1

x = 6

y = 2

1 ^ 0 = 1

0 ^ 0 = 0

1 ^ 1 = 0

x ^ y = x + y

3 4 5 6 7

→ sort in ↑

Ex

sort(A) = 10 12 17 20 25 30 31 35 50 55 60

[2, 9]

[A_l ... A_r] is the min length array to make the whole ~~to be sorted~~

array sorted

$A = [A_0 \ldots A_{l-1}] [A_l \ldots A_r] [A_{r+1} \ldots A_{n-1}]$

sorted — sorted ? — sorted

$\max([A_0 \ldots A_{l-1}]) \le \min([A_l \ldots A_r])$

$\max([A_l \ldots A_r]) \le \min([A_{r+1} \ldots A_{n-1}])$

1)

2)

→ O(n)

→ O(n)

L = 2, L = 3, R = 7, R = 9

0	1	2	3	4	5	6	7	8	9	10
10	12	20	30	25	17	55	37	35	50	60

[3, 7]

20

max = 55

min = 17

[l, r]

Ans: [2, 9]

→ Iterate from first element and find first element ≥ min

T. C: O(n)

1) Find the first Indices

2) Do 1 more traversal to find the correct L & R.

(L, R)

find Min & Max O(n)

(0 to l-1)

O(n)

O(n)

T. C: O(n)

S. C: O(1)

O(n) + O(n) + O(n) + O(n) + O(n)

T. C: O(n)

Maximum Unsorted Array

```
for(int i = 0; i < n; i++)
      B[i] = A[i];
sort(B);
for(i = 0; i < n;i++)
      if(A[i] != B[i])
            break;
for(j = n-1; j>=0; j--)
      if(A[j] != B[j])
            break;

return (i, j)
```

Construct array : Brute Force

```
min_so_far = INT_MAX
for(a = 1; a <= x; a++){
      for(d = 1; d<= y-x; d++){
            // Generate AP of length k
            // a a+d a+2d......
            for(int i = 0; i < k; i++)
                  temp[i] = a + i * d;

            last_ele = temp[k-1];
            if(last_ele < minm_so_far && isX && isY){
                  minm_so_far = last_ele;
                  // Update the answer array with new AP
                  for(int i = 0; i < k; i++)
                        ans[i] = temp[i];
            }
      }
}
```

Construct Array : Efficient Approach

```
vector<int> Solution::solve(const int A, const int B, const int C) {
    vector<int> ans(A, INT_MAX);

    vector<int> factors = getFactors(C - B);
    for(int i = 0; i < factors.size(); i++){
        int d = factors[i];
        int num_terms = (C - B)/d + 1;
        if(num_terms > A)
            continue;

        // This computes the no. of terms we can have before B with common differe
        int allowed_front_terms = (B - 1)/d;
        // Compute the first term of the series
        int a = B - min(allowed_front_terms, A - num_terms) * d;

        // Generate the Series
        int temp[A];
        for(int i = 0; i < A; i++)
            temp[i] = a + i*d;

        if(temp[A-1] < ans[A-1] || (temp[A-1] == ans[A-1] && temp[0] < ans[0])){
            for(int i = 0; i < A; i++)
                ans[i] = temp[i];
        }
    }

    return ans;
}
```

Choose N puzzles

```
sort(arr);
for(int i = 0; i <= M - N; i++)
    ans = min(ans, arr[i + N-1] - arr[i]);
}
return ans;
```