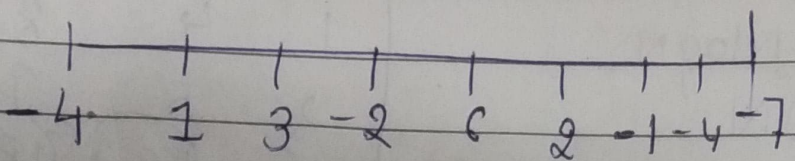
Arrays
Tricky - 0

Page No. _____
Date

(1) Maxm Contigious Sub-Array (Kadane's Algo)



cs = 0 -4 1 4 2 8 10 9 5 -2 0
ms = 0 1 4 8 10

```
int cs = 0;
int ms = 0;
```

```
int low, high;   (start) (end)
int l;
```

```
for i = 0 to n
{   cs = cs + a[i];
    if (cs > ms)
    {   ms = cs;
        high = i
        low = l
    }
    if (cs < 0)
    {   cs = 0;
        l = i + 1;
    }
}
```

Page No.
Date

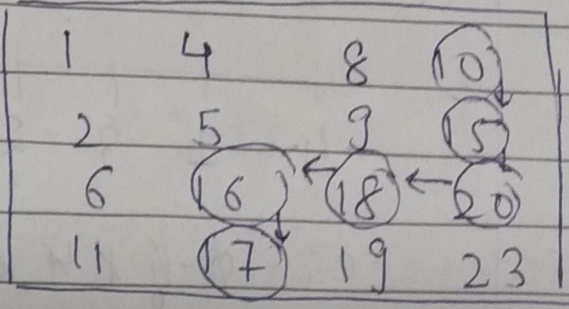
(2) Staircase Search

- row & col are sorted

(1) N^2

(2) NlogN

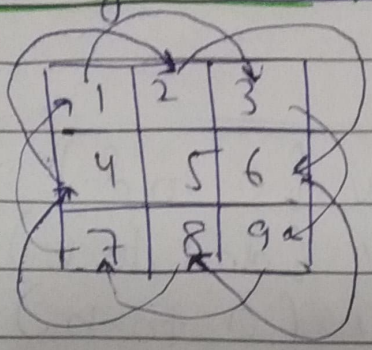
(3) N




```
int search (int mat[4][4], int val)
  {   if (sizee==0)
          return -1;
      if ( val < mat[0][0] || val > mat [size
                                         [siz
          return -1;

int i=0 , j= size-1;
  while ( i<n && j>=0)
    {   if ( mat[i][j] == val)
            return found
        if ( mat[i][j] > val)
            j--;
        else
            i++;
    }
  return -1;
}
```

Page No.
Date

(3) Image Rotation



90°

7	4	1
8	5	2
9	6	3

① Transpose

1 4 7
2 5 8
3 6 9

② Reverse Row

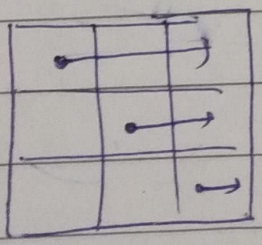
7 4 1
8 5 2
9 6 3

Reverse Row

```
for i = 0 to R
    for (j = 0, k = C-1 ; j < k ; j++, k--)
        swap(arr[i][j], arr[i][k]);
```

Transpose

```
for i = 0 to R
for j = i to C
    swap(arr[i][j], arr[j][i])
```



Page No.
Date

(4) Rotate an array d times

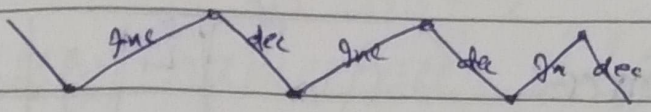
(1) rotate ( )
{ rev (0 to d)
rev (d+1 to n)
rev (0 to n)
}

(2) using push / back / insert and erase.

Sort (O(n log n) then swaps

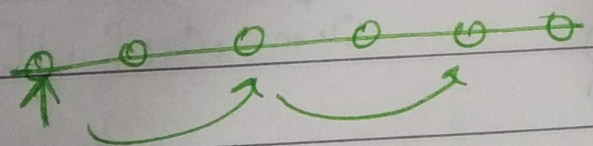
Page No. _____
Date | | | |

(5) Wave Sort



O(n) → Go to every 2nd element & make it peak.
→ check for left → check for right.

```
for i=0 ; to n ; it = 2
    {
// left check   if ( i>0 && arr[i-1] > arr[i])
                    swap(arr[i], arr[i-1]);

// right check  if ( i<=n-2 && arr[i+1] > arr[i])
                    swap(arr[i+1], arr[i]);
    }
```

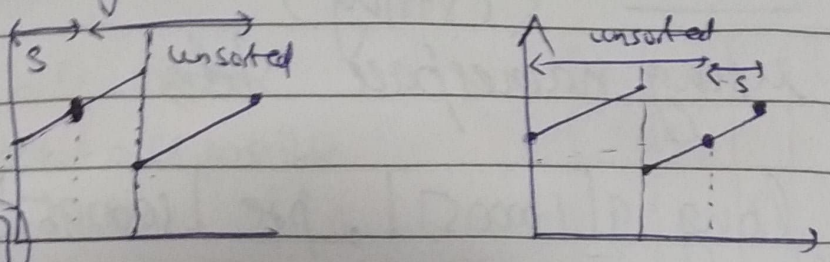


{ 4, 5, 1, 2, 3 }

Page No.
Date

(6) Binary Search in Sorted Rotated Array

Recursion



a[st] <= a[mid]

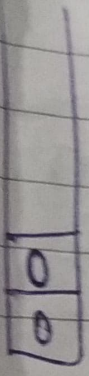
```
int search (int arr[], int l, int h, int val)
{   if ( l > h) return -1;

    int mid = (l+h)/2;
    if (arr[mid] == key) return mid;

if (arr[l] <= arr[mid])
  {   if ( val <= arr[mid] &&
           val >= arr[l]
    return search (arr, l, mid-1, val);
      else
        return search (arr, mid+1, h, val);
  }
else {
      if ( val >= arr[mid] && val <= arr[h]
    return search (arr, mid+1, h, val);
    else
    return search (arr, l, mid-1, val);
    }

}
```

Tricky - 1

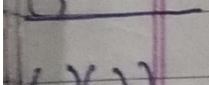
① Longest Valid Parantheses

(() → 2
)()()) → 4

```
stack<int> st;
int curr = 0;
int ans = 0;
for (i=0; i<s.length(); i++)
{ if (s[i] == '(')
    { st.push(curr);
      curr = 0;
    }
else
{ if (st.empty())
    curr = 0;
  else
    { curr += st.top() + 2;
      st.pop();
      ans = max(ans, curr);
    }
}
}
```



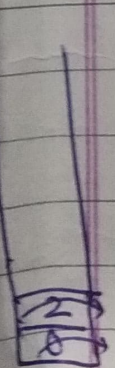
curr = 0 0 2



( ) ( )

// updating curr & ans when ')'

// storing curr if '(' and curr from 0 again.



ans = 0 0 2 4

Page No.
Date : / /

(2) Min length of contigious sub-array of which sum $\geq s$

2, 3, 1, 2, 4, 3
s = 7
→ 2

```
int sum = 0; start = 0; end = 0;
int ans = INT_MAX;

while ( end < arr.size() )
  {
    while (sum<s  && end<arr.siz)
      { sum += arr [end++]; }

    while(sum>=s  && start<arr.size)
      { ans = min (ans, end - start);
        sum -= arr [start++];
      }
  }

if (ans == INT_MAX) return 0;
return ans;
```

Page No.
Date / /

③ Max length of sub array that sums to k

10, 5, 2, 7, 1, 9
k = 15
→ 4

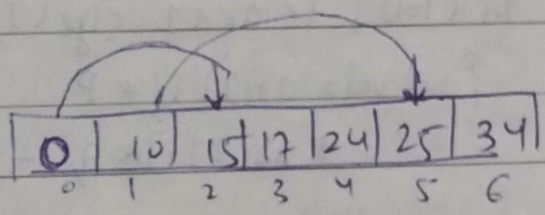
PrefixSum	0	10	15	17	24	25	34

(n^2)

O(n) →

map 0 → 0, 10 → 1, 15 → 2 .....

```
for (i=0; i< prefixSum.size(); i++)
  {
    if (m.find(k+PrefixSum[i]) != m.end())
        ans = max(ans, m[k+prefixSum[i]] - i)
  }
```



k + prefixSum[i] → 15 + 0 = 15
15 + 10 = 25
15 + 15 = 30

Max Absolute Value

Page No.
Date : / /

(4) arr1 arr2

return max value of

$|arr1[i] - arr1[j]| + |arr2[i] - arr2[j]| + |i$

1. arr1[i] + arr2[i] + i
2. arr[i] + arr2[i] + i
3. -arr[i] - arr2[i] + i
4. -arr[i] + arr2[i] + i

```
int max_Dist = 0,
for (int P: {-1, 1})
 { for (int q: {-1, 1})
    { int maxVal = INT_MIN;
      int minVal = INT_MAX;
      for (i=0; i<arr1.size(); i++)
       { int val = arr1[i] * P
                 + arr2[i] * q + i;
         maxVal = -
         minVal = -
       }
      maxDist = max (maxDst, maxVal - minV
    }
 }

return maxDist.
```

Page No.
Date : / /

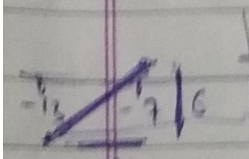
(5) Sum of All Sub Matrix Pre

for i=0 to n-1
  for j=0 to m-1
    sum += mat[i][j] * ((i+1)*(j+1)*(n-i)*(m-j))

---

(6) [(0,0) (1,1) (1,2)] → 2
[(7,1), (-13,-5)] → 6

Move in 8 dirn



```
for ( i=0 to n-1)
  { A = abs( - ) //x
    B = abs( - ) //y
    sum += max(A,B);
  }
```

---

(7) if 4 dirn

sum += A + B

Page No.
Date : / /

(8) a → [3, 30, 34, 5, 9]

→ "9 5 34 3 30"

```
sort ( . . . , mycmp), ans = "";
for (i=0 to b
   ans += b[i];

bool mycmp ( string x, string y)
   { xy = x+y;
     yx = y+x;
     return xy > yx;
   }
```

---

(9) Merge Intervals

[1,3] [6,9] [2,5] ← ① push

② sort with first val

③ [1,3] [2,5] [6,9]

max

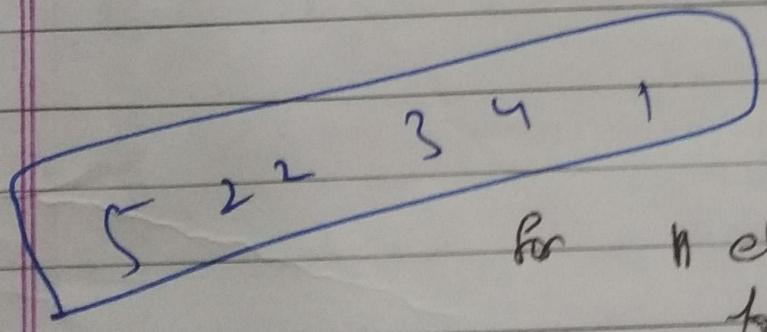
[1,5] [6,9]

Page No.
Date : / /

(10) Find Duplicate In Array

O(n) → Time
O(1) → space

```
slow = A[0]
fast = A[A[0]]

while (slow != fast)
   { slow = A[slow]
     fast = A[A[fast]]
   }
fast = 0;
while (slow != fast)
   { slow = A[slow]
     fast = A[fast];
   }
if (slow == 0) return -1;
return slow;
```



for n elements
from 1 to n-1

Page No.
Date : / /

(11) Rain Water harvesting

```
left[0] = height[0]
for i = 1 to A
    left[i] = max(left[i-1], A[i-1])

right[A.size() - 1] = A[A.size() - 1]
for (i = A.size() - 2 ; i >= 0 ; i--)
    right[i] = max(right[i+1], A[i+1]);

int ans = 0
for (i = 0 to A.size())
{ d = (min(left[i], right[i]) - A
    if (d < 0) d = 0;
    ans += d;
}
return ans;
```

(12) histogram Area

A.push_back(0)
or
while(!st.empty())
{
—
}

```
int maxarea = 0;
stack <int> st.
 i=0;
 while (i < A.size ())
  { if (st.empty() || A[i] >= A[st.top()]
        st.push( i++);
     else
      { int top = st.top();
        st.pop();
 maxarea = max (maxarea, A[top] *
                (st.empty()? i : i-s.top()
                                    -1 ,
      }
  }
```

i-s.top()-1

deque<int> dq;

Page No.
Date : / /

(13) Max Sliding Window

[1, 3, -1, 3, 5, 3, 6, 7] K=3
→ [3, 3, 5, 5, 6, 7]

```
for (i=0 to A)
  { if( !dq.empty() && dq.front == i-k)
        dq.pop_front();

  while( !dq.empty() && A[i] >= A[dq.back]
        dq.pop_back();

      dq.push_back(i);

  if( i >= k-1)
      v.push_back(A[dq.front]);

  }

return v;
```

Page No.
Date : / /

Sub

(14) Longest Fibonacci Sequence

1, 2, 3, 4, 5, 6, 7, 8
1, 2, 3, 5, 8 → (5)

```
int solve(vector<int> A)
{ unordered_set<int> s(A.begin(), A.end());
  int ans = 0;

  for(int i=0; i<A.size(); i++)
    for(int j=i+1; j<A.size(); j++)
      { int a = A[i], b[i]=A[j], l=2;
        while(s.count(a+b))
          { l++;
            b=a+b; a=b-a;
          }
        ans = max(ans, l);
      }

  return res>2? res : 0;
}
```

Page No.
Date : / /

(15) Repeating & Missing Nos 1 3 3 2 4

1 3 3 2 4
x 2 3 4 5 = 3×5 = 15 = (1111)

for i & set_bit_no
for arr[i] & set_bit_no

2 unique elements

1 3 3 4 (2 6) 4 = 2×6

```
int set_bit_no = xor & ~(xor-1)

for (int i=0; i<arr[i] ; i++)        x=0, y=0
{
  if (arr[i] & set_bit_no)
      x = x ^ arr[i];
  else
      y = y ^ arr[i];
}
```

~(xor-1)

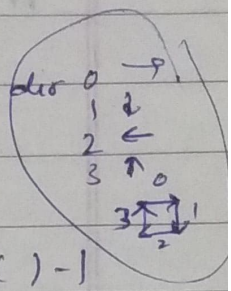
x or

Page No.
Date : / /

rev

(16) Spiral Matrix

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \Rightarrow [1,2,3,4,5,6,7,8,9]$$

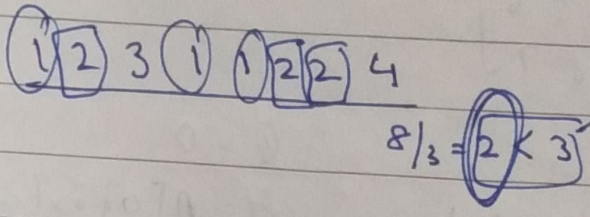



```
t=0;                    l=0
b=A.size()-1            r=A[0].size()-1

                dir=0;
while( t<=b && l<=r)
  { if (dir==0)
    {for(int i=l; i<=r ; i++)
       {ans.push_back(A[t][i]);}
     t++; dir=1;
    }

  else if (dir==1)
     for(int i=t to b
          ans.push_back(A[i][r]);
     r--; dir=2;

  else if (dir==2)
     for(i=r to l)
                [b][i]
     b--; dir=3

  else if (dir==3)
     for(i=b to t)
                [i][l]
     l++; dir=0;
  }
}
```

17

Page No.
Date : / /

Q find all element which appear more than N/3 times




```
for (i=1  to n)
 {if(arr[i]==el1)
     count1++;
else if(arr[i]==el2)
     count2++;
else
   if(count1=0)
      el1=arr[i];
      count1++.
else if (count2==0)
      el2=arr[i];
      count2++;
else
     count1--;
     count2--;
 }
count1=0; count2=0;
for (i=0 to arr.size)
 {if(arr[i]==el1)
     count1++;
else if(arr[i]=el2)
     count2++;
```

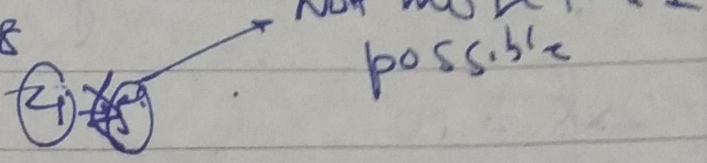
```
int count1=1;
int count2=0;
el1=arr[0];
el2=0;
```

el1, el2 → max times occur them 2

Page No.
Date : / /

N/2 times

1 1 1 5 6 7 8




```
el = arr[0]; cout = 1;

for (i=1 to n)
{ if (arr[i] == el)
      cout ++;

  else
      cout --;

  if (cout == 0)
      el = arr[i];
      cout ++;
  }
  return el;
}
```

100 → 124

Page No.
Date : / /

(18) $n!$ → count trailing 0's

$2 \times 5 = 10$

less

```
while (n>0)
{  n = n/5;
   count += n;
}
```

(19) hcf × lcm = a × b

```
int hcf (int a, int b)
{
  return b==0 ? a : hcf (b, a%b)
}
```