1. Stock buy and sell:

CODE:

```
import java.util.*;
class Stock{
   public static int maxprofit(int[] prices){
      int mini=prices[0];
      int r=0;
      for(int i=1;i<prices.length;i++){
         mini=Math.min(mini,prices[i]);
         r=Math.max(r,prices[i]-mini);
      }
      return r;
   }
   public static void main(String[] args){
      int[] prices={7,10,1,3,6,9,2};
      System.out.println(maxprofit(prices));
   }
}
```

OUTPUT:

```
[Running] cd "c:\Data\Knowledge\DSA\Programs\" && javac Stock.java && java Stock
8
```

Time Complexity: O(n)

2. Coin Change:

CODE:

```
import java.util.*;
class Coinchange{
   public static int count(int[] coin, int s, int n, int[][] dp){
      if(s==0) return dp[n][s]=1;
      if(n==0 || s<0) return 0;
      if(dp[n][s]!=-1) return dp[n][s];
      return dp[n][s]=count(coin,s-coin[n-1],n,dp)+count(coin,s,n-1,dp);
   }
   public static void main(String[] args){
      int t=1;
      while(t!=0){
         int n,s;
         n=3;
         s=5;
         int[] coin={1,2,3};
         int[][] dp=new int[n+1][s+1];
         for(int[] r:dp) Arrays.fill(r,-1);
         int res=count(coin,s,n,dp);
         System.out.println(res);
         t--;
      }
   }
}
```

OUTPUT:

```
[Running] cd "c:\Data\Knowledge\DSA\Programs\" && javac Coinchange.java && java Coinchange
5
```

Time Complexity: O(n*s)

3. First and last occurrences:

CODE:

```
import java.io.*;
class first_last{
   public static int first(int a[], int l, int h, int x, int n){
      if(h>=l){
         int m=l+(h-l)/2;
         if((m==0 || x>a[m-1]) && a[m]==x) return m;
         else if(x>a[m]) return first(a,(m+1), h,x,n);
         else return first(a,l,(m-1),x,n);
      }
      return -1;
   }
   public static int last(int a[], int l, int h, int x, int n){
      if(h>=l){
         int m=l+(h-l)/2;
         if((m==n-1 || x<a[m+1]) && a[m]==x) return m;
         else if(x<a[m]) return last(a,l,(m-1),x,n);
         else return last(a,(m+1),h,x,n);
      }
      return -1;
   }
   public static void main(String[] args){
      int a[] = {1,2,2,2,2,3,4,7,8,8};
      int n=a.length;
      int x=8;
      System.out.println("First occurrence: "+first(a,0,n-1,x,n));
      System.out.println("Last occurrence: "+last(a,0,n-1,x,n));
   }
}
```

OUTPUT:

```
[Running] cd "c:\Data\Knowledge\DSA\Programs\" && javac first_last.java && java first_last
First occurrence: 8
Last occurrence: 9
```

Time Complexity: O(log n)

4. Find transition point:

CODE:

```
import java.io.*;
class Transition_pt{
   public static int Transition(int a[], int n){
      int l=0,r=n-1;
      while(l<=r){
         int m=(l+r)/2;
```

```
        if(a[m]==0) l=m+1;
        else if(a[m]==1){
            if(m==0 || (m>0 && a[m-1]==0)) return m;
            r=m-1;
        }
    }
    return -1;
  }
  public static void main(String[] args){
    int a[]={0,0,0,0,1,1};
    int pt=Transition(a,a.length);
    System.out.println(pt>=0 ? "Transition point is: " + pt : "There is no Transition point");
  }
}
```

OUTPUT:

```
[Running] cd "c:\Data\Knowledge\DSA\Programs\" && javac Transition_pt.java && java Transition_pt
Transition point is: 4
```

Time Complexity: O(log n)

5. Find repeating element:

CODE:

```
import java.util.*;
class Repeat{
  public static void firstrepeat(int[] a){
    int mini=-1;
    HashSet<Integer> s=new HashSet<>();
    for(int i=a.length-1;i>=0;i--){
      if(s.contains(a[i])) mini=i;
      else s.add(a[i]);
    }
    if(mini!=-1) System.out.println("The first repeating element is: "+a[mini]);
    else System.out.println("There are no repeating elements");
  }
  public static void main(String[] args){
    int a[]= {10,5,3,4,3,5,6};
    firstrepeat(a);
  }
}
```

OUTPUT:

```
[Running] cd "c:\Data\Knowledge\DSA\Programs\" && javac Repeat.java && java Repeat
The first repeating element is: 5
```

Time Complexity: O(n)

6. Remove duplicates sorted array:

CODE:

```
import java.util.*;
class Remove_duplicate{
  public static int remdupl(int[] a){
    int n=a.length;
```

```
        if(n<=1) return n;
        int ind=1;
        for(int i=1;i<n;i++){
            if(a[i]!=a[i-1]){
                a[ind++]=a[i];
            }
        }
        return ind;
    }
    public static void main(String[] args){
        int[] a={1,2,2,3,4,4,4,5,5};
        int newsize=remdupl(a);
        for(int i=0;i<newsize;i++){
            System.out.print(a[i]+" ");
        }
    }
}
```

OUTPUT:

```
[Running] cd "c:\Data\Knowledge\DSA\Programs\Day_5\" && javac Remove_duplicate.java && java
Remove_duplicate
1 2 3 4 5
```

Time Complexity: O(n)

7. Maximum index:

CODE:

```
import java.util.*;
public class Max_ind{
    int maxi(int x,int y){
        return x>y ? x : y;
    }
    int mini(int x, int y){
        return x<y ? x : y;
    }
    int maxidxdiff(int a[], int n){
        int maxdiff;
        int i, j;
        int rmax[]=new int[n];
        int lmin[]=new int[n];
        lmin[0]=a[0];
        for(i=1;i<n;++i) lmin[i]=mini(a[i],lmin[i-1]);
        rmax[n-1]=a[n-1];
        for(j=n-2;j>=0;--j) rmax[j]=maxi(a[j],rmax[j+1]);
        i=0;
        j=0;
        maxdiff=-1;
        while(j<n && i<n){
            if(lmin[i]<=rmax[j]){
                maxdiff=maxi(maxdiff,j-i);
                j+=1;
```

```
        }else i+=1;
      }
      return maxdiff;
   }
   public static void main(String[] args) {
      Max_ind x=new Max_ind();
      int a[]={9,2,3,4,5,6,7,8,18,0};
      int n=a.length;
      int maxdiff=x.maxidxdiff(a,n);
      System.out.println(maxdiff);
   }
}
```

OUTPUT:

```
[Running] cd "c:\Data\Knowledge\DSA\Programs\Day_5\" && javac Max_ind.java && java Max_ind
8
```

Time Complexity: O(n)

8. Wave array:

CODE:

```
import java.util.*;
public class Wave_array{
   void swap(int a[], int x, int y){
      int t=a[x];
      a[x]=a[y];
      a[y]=t;
   }
   void wave(int[] a, int n){
      for(int i=0;i<n;i+=2){
         if(i>0 && a[i-1]>a[i])
            swap(a,i,i-1);
            if(i<n-1 && a[i+1]>a[i])
            swap(a,i,i+1);
      }
   }
   public static void main(String[] args){
      Wave_array w=new Wave_array();
      int a[]={10,90,49,2,1,5,23};
      int n=a.length;
      w.wave(a,n);
      for(int i:a) System.out.print(i+" ");
   }
}
```

OUTPUT:

```
[Running] cd "c:\Data\Knowledge\DSA\Programs\Day_5\" && javac Wave_array.java && java Wave_array
90 10 49 1 5 2 23
```

Time Complexity: O(n)