Strong password

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    static int minimumNumber(int n, String pwd) {
        if(n<=3) return 6-n;
        boolean num = false, lower = false, upper = false, special = false;
        for(char c : pwd.toCharArray()){
            if(isNum(c)) num = true;
            else if(isLower(c)) lower = true;
            else if(isUpper(c)) upper = true;
            else special = true;
        }
        boolean length = (n>=6);
        int count = 0;
        if(!num) count++;
        if(!lower) count++;
        if(!upper) count++;
        if(!special) count++;
        return (count+n < 6) ? 6-n : count;
    }

    static boolean isNum(char c){
```

```
        return (c>='0' && c<='9');
    }

    static boolean isLower(char c){
        return (c>='a' && c<='z');
    }

    static boolean isUpper(char c){
        return (c>='A' && c<='Z');
    }

    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        String password = in.next();
        int answer = minimumNumber(n, password);
        System.out.println(answer);
        in.close();
    }
}
```

Java comparator

```
import java.util.*;


// Write your Checker class here
class Checker implements Comparator {
    public int compare(Object o1,  Object o2) {
        Player p1 = (Player) o1;
        Player p2 = (Player) o2;
        if (p2.score - p1.score == 0) return p1.name.compareTo(p2.name);
        return p2.score - p1.score;
    }
}
```

Pattern syntax

```
import java.util.Scanner;
import java.util.regex.*;

public class Solution
{
  public static void main(String[] args){
    Scanner in = new Scanner(System.in);
      int testCases = Integer.parseInt(in.nextLine());

      while(testCases>0){
        String pattern = in.nextLine();

        try {
          Pattern.compile(pattern);
          System.out.println("Valid");
        } catch(PatternSyntaxException e) {
          System.out.println("Invalid");
        }

        testCases--;
      }
  }
}
```

Height of a binary tree

```
int getHeight(struct node* root) {
   if (root == NULL) {
      return -1; // Height of an empty tree is -1 (base case)
   } else {
      int leftHeight = getHeight(root->left);
      int rightHeight = getHeight(root->right);

      // Height of the tree is the maximum height of left or right subtree + 1
      return (leftHeight > rightHeight ? leftHeight : rightHeight) + 1;
   }
}
```

Duplicate words 1

```
import java.io.*;
import java.util.*;
import java.security.*;
public class Solution {
   public static void main(String[] args) {
      /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class should be named Solution. */
      Scanner scanner = new Scanner(System.in);
      String key = scanner.next();
      try{
      MessageDigest md = MessageDigest.getInstance("SHA-256");
         md.update(key.getBytes());
            byte[] digest = md.digest();
            StringBuffer stringbuffer = new StringBuffer();
      for (byte b: digest)
```

```
      {  // needed to print it in hexadecimal format
         stringbuffer.append(String.format("%02x", b));
      }
            System.out.println(stringbuffer.toString());
            }
       catch (NoSuchAlgorithmException exception)
      {
         System.out.println(exception);
      }

   }
}
```

Duplicate words 2

```
import java.util.Scanner;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
public class DuplicateWords {
   public static void main(String[] args) {
      String regex = "\\b(\\w+)(?:\\W+\\1\\b)+";
      Pattern p = Pattern.compile(regex, Pattern.CASE_INSENSITIVE);
       Scanner in = new Scanner(System.in);
      int numSentences = Integer.parseInt(in.nextLine());
        while (numSentences-- > 0) {
         String input = in.nextLine();
                  Matcher m = p.matcher(input);
          // Check for subsequences of input that match the compiled pattern
         while (m.find()) {
             input = input.replaceAll(m.group(), m.group(1));
         }
         // Prints the modified sentence.
```

```
            System.out.println(input);
        }
                in.close();
    }
}
```

Insertion sort

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {




    public static void insertIntoSorted(int[] ar) {
        int sort = ar[ar.length - 1];
        int i;
        for (i = ar.length - 2; (i >= 0) && (ar[i] > sort); i--) {
            ar[i + 1] = ar[i];
            printArray(ar);
        }
        ar[i + 1] = sort;
        printArray(ar);

    }



/* Tail starts here */
```

```
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       int s = in.nextInt();
       int[] ar = new int[s];
        for(int i=0;i<s;i++){
           ar[i]=in.nextInt();
        }
        insertIntoSorted(ar);
   }


   private static void printArray(int[] ar) {
     for(int n: ar){
        System.out.print(n+" ");
     }
       System.out.println("");
  }


}
```

The power sum

```
import java.util.Scanner;


public class PowerSum {
            int count=0;
            int sum;
            int pow;
   public static void main(String[] args) {
                        Scanner in=new Scanner(System.in);
                        int x=in.nextInt();
```

```
                int n=in.nextInt();
        PowerSum p=new PowerSum();
        p.sum=x;
        p.pow=n;
        int N=(int)Math.pow(x, (1.0/n));
        p.getcount(p.sum,N,true);
        p.getcount(p.sum,N,false);
        System.out.println(p.count);
        in.close();
            }
    void getcount(int sum1,int n,boolean lenyani){

            if(lenyani==true){
                sum1=sum1-(int)Math.pow(n, pow);

            }
            if(sum1<0) return;
            if(sum1==0){
                count++;
                return;
            }
            if(n==1) return;
            getcount(sum1,n-1,true);
            getcount(sum1,n-1,false);
    }

}
```

Running time of algorithm

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void insertionSort(int[] A){
        int shifts = 0;
        for(int i = 1; i < A.length; i++){
            int value = A[i];
            int j = i - 1;
            while(j >= 0 && A[j] > value){
                A[j + 1] = A[j];
                j = j - 1;
            }
            A[j + 1] = value;
            shifts += i - (j+1);
        }

        //printArray(A);
        System.out.println(shifts);
    }


    static void printArray(int[] ar) {
        for(int n: ar){
            System.out.print(n+" ");
        }
    }
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
```

```
        int n = in.nextInt();
        int[] ar = new int[n];
        for(int i=0;i<n;i++){
           ar[i]=in.nextInt();
        }
        insertionSort(ar);
    }
}
```