**STRINGS - 1. Validation**

```
//file TransactionParty.java

public class TransactionParty {
	String seller;
	String buyer;
	public TransactionParty(String seller, String buyer) {
		this.seller = seller;
		this.buyer = buyer;
	}
}

//file Receipt.java

public class Receipt {
	TransactionParty transactionParty;
	String productsQR;
	public Receipt(TransactionParty transactionParty, String productsQR) {
		this.productsQR = productsQR;
		this.transactionParty = transactionParty;
	}
}

// file GenerateReceip.java

import java.util.regex.Pattern;

public class GenerateReceipt {

	public int verifyParty(Receipt r) {
		int numOfVerifiedNames = 0;
		String seller = r.transactionParty.seller;
		String buyer = r.transactionParty.buyer;
		String re = "^[a-zA-Z-']+\s{1}[a-zA-Z-']+$";
		if(Pattern.matches(re, seller))
			numOfVerifiedNames += 1;
		if(Pattern.matches(re, buyer))
			numOfVerifiedNames += 1;
		return numOfVerifiedNames;
	}

	public String calcGST(Receipt r) {
		String productQR = r.productsQR;
		String[] products = productQR.split("@");
		int total = 0;
		double GST_Rate = 0.12;

		for(String prod : products) {
			String[] rateQuant = prod.split(",");
			int rate = Integer.parseInt(rateQuant[0]);
			int quantity = Integer.parseInt(rateQuant[1]);
			total += rate*quantity;
		}
		Integer GST = (int) (total*GST_Rate);
		return GST.toString();
	}
}

//file Validation.java
public class Validation {
```

```
	public static void main(String[] args) {
		TransactionParty tparty = new TransactionParty("Daniel D'Cruz","Giselle Dawn-Wright");
		String prodQR = "250,10@100,3@50,7";
		Receipt rp = new Receipt(tparty, prodQR);
		GenerateReceipt genR = new GenerateReceipt();
		int validNames = genR.verifyParty(rp);
		System.out.println("Seller: "+rp.transactionParty.seller);
		System.out.println("Buyer: "+rp.transactionParty.buyer);
		System.out.println("Number of valid names: "+validNames);
		System.out.println("GST: "+genR.calcGST(rp));
	}
}
```

******STRINGS - 2. Employee Information******

```
//file Employee.Java

public class Employee {

	String name,ssn,dept;
	int salary;

	public Employee(String name,String ssn,String dept,int salary) {
		this.name = name;
		this.ssn = ssn;
		this.dept = dept;
		this.salary = salary;
	}
}

//file EmployeeImplementation.java

public class EmployeeImplementation {

	public Employee getEmployeeInfo(String str) {
		String name = "";
		String ssn = "";
		String dept = "";
		int salary = 0;

		String[] employeeInfo = str.split("@|-|#");
		name = employeeInfo[0];
		ssn = employeeInfo[1];
		dept = employeeInfo[2];
		salary = Integer.parseInt(employeeInfo[3]);

		Employee emp = new Employee(name, ssn, dept, salary);
		return emp;
	}

	public String getEmployeeDept(Employee e) {
		String dept = "";
		String SSN = e.ssn.substring(e.ssn.length() - 3);
		int ssnNumber = Integer.parseInt(SSN);

		if(ssnNumber >= 1 && ssnNumber <= 60 ) {
			dept = "L1";
		} else if(ssnNumber >= 61 && ssnNumber <= 120 ) {
			dept = "L2";
		} else if(ssnNumber >= 121 && ssnNumber <= 180 ) {
```

```
                        dept = "L3";
                } else {
                        dept = "L4";
                }
                return dept;
        }
}

//file EmployeeInformation.java

public class EmployeeInformation {

        public static void main(String[] args) {
                String empInfo = "Amit Rai@1PC16CS046-ALU#8";
                EmployeeImplementation empImp = new EmployeeImplementation();
                Employee emp = empImp.getEmployeeInfo(empInfo);
                String dept = empImp.getEmployeeDept(emp);

                System.out.println("Emp Name: "+emp.name);
                System.out.println("Emp SSN: "+emp.ssn);
                System.out.println("Emp Dept Name: "+emp.dept);
                System.out.println("Emp Salary: "+emp.salary+"LPA");
                System.out.println("Emp Dept: "+dept);
        }
}
```

**COLLECTIONS - 3. TV Show**

```
import java.util.ArrayList;
import java.util.Scanner;

public class Source {

        public String printIndex(ArrayList<String> list, int ind) {
                return list.get(ind);
        }

        public ArrayList<String> addAfter(ArrayList<String> a, String m, String n){
                int index = a.indexOf(m) + 1;
                a.add(index, n);
                return a;
        }

        public ArrayList<String> pickIndexAndAppend(ArrayList<String> p, int ind){
                String str = p.remove(ind);
                p.add(str);
                return p;
        }

        public static void main(String[] args) {
                Scanner sc = new Scanner(System.in);
                ArrayList<String> shows = new ArrayList<String>();
                String show1 = sc.nextLine();
                String show2 = sc.nextLine();
                String show3 = sc.nextLine();
                String show4 = sc.nextLine();
//              shows.add("Breaking Bad");
//              shows.add("Young Sheldon");
//              shows.add("Friends");
//              shows.add("Stranger Things");
                shows.add(show1);
                shows.add(show2);
```

```
                shows.add(show3);
                shows.add(show4);
                Source source =  new Source();
                System.out.println(shows);
                System.out.println("Element at index 3: " + source.printIndex(shows, 3));
                System.out.println("After adding element Sherlock after Friends : "+ source.addAfter(shows, "Friends", "Sherlock"));
                System.out.println("Pick string at index 2 and append it at the end: " + source.pickIndexAndAppend(shows, 2));
                sc.close();
        }
}
```

**COLLECTIONS - 4. Set Operations**

```
import java.util.HashSet;
import java.util.Set;

public class Source {

        public Set<Integer> subtract(Set<Integer> a, Set<Integer> b){

                HashSet<Integer> result = new HashSet<Integer>(a);
                for(int element: b) {
                        if(result.contains(element))
                                result.remove(element);
                }
                return result;
        }

        public Set<Integer> union(Set<Integer> a, Set<Integer> b){
                HashSet<Integer> result = new HashSet<Integer>(a);
                result.addAll(b);
                return result;
        }

        public Set<Integer> intersection(Set<Integer> a, Set<Integer> b){
                HashSet<Integer> result = new HashSet<Integer>(a);
                result.retainAll(b);
                return result;
        }

        public static void main(String[] args) {
                HashSet<Integer> set1 = new HashSet<Integer>();
                set1.add(5);
                set1.add(6);
                set1.add(7);
                set1.add(8);

                HashSet<Integer> set2 = new HashSet<Integer>();
                set2.add(9);
                set2.add(3);
                set2.add(7);

                Source source = new Source();
                System.out.println("Set1: "+set1);
                System.out.println("Set2: "+set2);
                System.out.println("Set Difference: "+ source.subtract(set1, set2));
                System.out.println("Set Union: " + source.union(set1, set2));
                System.out.println("Set Intersection: " + source.intersection(set1, set2));
        }

}
```

**COLLECTIONS - 5. String Position**

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class Source {

	public List<List<Integer>> printPositions(String K){
		List<List<Integer>> result = new ArrayList<>();

		for (int i = 0, j = 0; i < K.length(); i = j) {
	while (j < K.length() && K.charAt(j) == K.charAt(i)) {
	    j++;
	}
	if (j - i >= 3) {
	    List<Integer> group = Arrays.asList(i, j - 1);
	    result.add(group);
	}
		}
		return result;
	}

	public ArrayList<String> addAfter(ArrayList<String> a, String m, String n){
		ArrayList<String> result = new ArrayList<String>();
		for(String str: a) {
			result.add(str);
			if (str.equals(m)){
				result.add(n);
			}
		}
		return result;
	}

	public static void main(String[] args) {

		Source source = new Source();
		String str = "mousssssseeee";

		List<List<Integer>> res = new ArrayList<List<Integer>>();
		res = source.printPositions(str);
		System.out.println(res);

		ArrayList<String> list = new ArrayList<>();
		list.add("ad");
		list.add("cc");
		list.add("df");
		list.add("ez");

		System.out.println(source.addAfter(list, "cc", "kc"));
	}
}
```

**COLLECTIONS - 6. Longest Substring**

```
import java.util.HashSet;
import java.util.Set;

public class Source {

	public int lengthOfLongestSubstring(String s, Set<Character> set) {
```

```
		int max = 0, i = 0, j = 0;
		while(i < s.length()) {
			if(!set.contains(s.charAt(i))) {
				set.add(s.charAt(i++));
				max = Math.max(max, set.size());
			} else {
				set.remove(s.charAt(j++));
			}
		}
		return max;
	}


	public static void main(String[] args) {
		String str = "abcabcbb";
		Set<Character> c = new HashSet<Character>();
		Source source = new Source();
		System.out.println(source.lengthOfLongestSubstring(str, c));
	}
}
```

**COLLECTIONS - 7. Students Information**

```
import java.util.ArrayList;

public class Source {

	public ArrayList<String> changeOccurrence(ArrayList<String> a,String m,String n){
		for(int i = 0; i<a.size(); i++) {
			String currentStr = a.get(i);
			if(currentStr.equals(m))
				a.set(i, n);
		}
		return a;
	}

	public String listIndex(ArrayList<String> list) {
		return list.get(0);
	}

	public ArrayList<String> listAfter(ArrayList<String> a, String m, String n){
		ArrayList<String> result = new ArrayList<String>();
		for(String str: a) {
			result.add(str);
			if (str.equals(m)){
				result.add(n);
			}
		}
		return result;
	}

	public static void main(String[] args) {
		Source source = new Source();
		ArrayList<String> list = new ArrayList<>();
		list.add("A");
		list.add("B");
		list.add("S");
		list.add("D");
		list.add("B");
		System.out.println(list);
		System.out.println(source.changeOccurrence(list, "B", "F"));
		System.out.println(source.listIndex(list));
```

```
                System.out.println(source.listAfter(list, "F", "Z"));
        }
}
```

******COLLECTIONS - 8. Anagrams******

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class Source
{
   public List<Integer> findAnagrams(String s, String p)
   {
      List<Integer> rst = new ArrayList<>();
      if (s == null || s.length() == 0 || s.length() < p.length())
      {
         return rst;
      }

      int[] map_p = new int[26];
      int[] map_s = new int[26];
      // Initialize the map / window
      for (int i = 0; i < p.length(); i++)
      {
         map_p[p.charAt(i) - 'a']++;
      }
      for (int i = 0; i < p.length(); i++)
      {
         map_s[s.charAt(i) - 'a']++;
      }

      for (int i = 0; i < s.length() - p.length(); i++)
      {
         if (isMatch(map_p, map_s))
         {
            rst.add(i);
         }
         // if don't match, we move the sliding window
         // remove the preceding character and add a new succeeding character to the new window
         map_s[s.charAt(i+p.length()) - 'a']++;
         map_s[s.charAt(i) - 'a']--;
      }
      if (isMatch(map_p, map_s))
      {
         rst.add(s.length() - p.length());
      }
      return rst;
   }

   public boolean isMatch(int[] arr1, int[] arr2)
   {
      for (int i = 0; i < arr1.length; i++)
      {
         if (arr1[i] != arr2[i])
         {
            return false;
         }
      }
      return true;
   }
```

```
   public static void main(String[] args)
   {
            Scanner sc = new Scanner(System.in);
            String s =sc.nextLine();
            String p = sc.nextLine();
            Source source = new Source();
            System.out.println(source.findAnagrams(s,p));
            sc.close();    }
}
```

**COLLECTIONS - 9. Encryption**

```
import java.text.CharacterIterator;
import java.text.StringCharacterIterator;
import java.util.HashMap;
import java.util.HashSet;

public class Source {

            public int uniqueMorseRepresentations(String[] words) {
                        String temp[] = new String[]
{".-","-...","-.-.","-..",".","..-.","--.","....","..",".---","-.-",".-..","--","-.","---",".--.","--.-",".-.","...","-","..-","...-",".--","-..-","-.--","--.."};
                        HashMap<Character,String> morse = new HashMap<>();

                        int i = 0;
                        for(char c = 'a'; c<='z' && i<26; c++,i++) {
                                    morse.put(c, temp[i]);
                        }
                        String morseCode = "";
                        HashSet<String> set = new HashSet<>();

                        for(String word: words) {
                                    CharacterIterator it = new StringCharacterIterator(word);
                                    while(it.current() != CharacterIterator.DONE) {
                                                morseCode += morse.get(it.current());
                                                it.next();
                                    }
                                    set.add(morseCode);
                                    morseCode = "";
                        }
                        return set.size();
            }

            public static void main(String[] args) {
                        Source source = new Source();
                        String[] str = new String[] {"gin", "zen", "gig", "msg"};
                        System.out.println(source.uniqueMorseRepresentations(str));
            }
}
```

**EXCEPTIONS - 10. Find Age**

```
// Find Age

//file Age.java

public class Age {
            String drink;
            String vote;
            String movie;
}
```

```
//file IllegalAgeException.java

public class IllegalAgeException extends Exception {

	public IllegalAgeException(String s) {
		super(s);
	}
}

//file ExceptionCheck.java

public class ExceptionCheck {

	public String drinkingCheck(Age a, int age) {

		try {
			if(age < 21) {
				a.drink = "illegal";
				throw new IllegalAgeException("Illegal drinking age");
			} else {
				a.drink = "legal";
				return a.drink;
			}
		}catch(IllegalAgeException e) {
			return e.getMessage();
		}
	}

	public String votingCheck(Age a, int age){
		try {
			if(age < 18) {
				a.vote = "illegal";
				throw new IllegalAgeException("Illegal voting age");
			} else {
				a.vote = "legal";
				return a.vote;
			}
		}catch(IllegalAgeException e) {
			return e.getMessage();
		}
	}

	public String movieCheck(Age a, int age) {
		try {
			if(age < 14) {
				a.movie = "illegal";
				throw new IllegalAgeException("Illegal movie-watching age");
			} else {
				a.movie = "legal";
				return a.movie;
			}
		}catch(IllegalAgeException e) {
			return e.getMessage();
		}
	}

	public static void main(String[] args) {
		Age age = new Age();

		ExceptionCheck ec = new ExceptionCheck();
```

```
                System.out.println(ec.drinkingCheck(age, 15));
                System.out.println(ec.votingCheck(age, 15));
                System.out.println(ec.movieCheck(age, 15));

                System.out.println("Drink :" + age.drink);
                System.out.println("Movie :" + age.movie);
                System.out.println("Vote :" + age.vote);
        }
}
```

**EXCEPTIONS - 11. File Check**

```
package check.file;

import java.io.File;
import java.io.FileNotFoundException;
import java.util.ArrayList;

public class ExceptionCheck {
        public ArrayList<String> numberCheck(String str){
                ArrayList<String> result = new ArrayList<>();
                int number;
                for(char c: str.toCharArray()) {
                        try {
                                number = Integer.parseInt(String.valueOf(c));
                                result.add(String.valueOf(number));
                        }catch(NumberFormatException ex) {
                                result.add(ex.getMessage());
                        }
                }
                return result;
        }

        public String fileCheck(String filename) {
                try {
                        File file = new File(filename);
                        if (file.exists())
                                return "File Found";
                        throw new FileNotFoundException();
                }catch(FileNotFoundException ex) {
                        return ex.getMessage();
                }
        }


        public static void main(String[] args) {
                String str = "10ASD";
                ExceptionCheck ec = new ExceptionCheck();
                System.out.println(ec.numberCheck(str));
                String filename = "abc.txt";
                System.out.println(ec.fileCheck(filename));
        }
}
```

**EXCEPTIONS - 12. Email Operation**

```
//file Header.java

public class Header {
```

```
		String from;
		String to;

		public Header(String from, String to) {
			this.from = from;
			this.to = to;
		}
}

//file Email.java

public class Email {
		Header header;
		String body;
		String greetings;

		public Email(Header header, String body, String greetings) {
			this.header = header;
			this.body = body;
			this.greetings = greetings;
		}

		public int emailVerify(Email e) {
			int numOfVerifiedEmail = 0;
			String re = "^[a-zA-Z_]+@{1}[a-zA-Z]+\\.[a-zA-Z]{2,}$";
			if(Pattern.matches(re, e.header.from))
				numOfVerifiedEmail += 1;
			if(Pattern.matches(re, e.header.to))
				numOfVerifiedEmail += 1;

			return numOfVerifiedEmail;
		}

		public String bodyEncryption(Email e) {
			StringBuffer result = new StringBuffer();
			String text = e.body;
			int s = 3;
	for (int i=0; i<text.length(); i++)
	{
	   if(Character.isWhitespace(text.charAt(i))) {
	       result.append(text.charAt(i));
	       continue;
	   }
	       if (Character.isUpperCase(text.charAt(i)))
	   {
	      char ch = (char)(((int)text.charAt(i) + s - 65) % 26 + 65);
	      result.append(ch);
	   }
	   else
	   {
	      char ch = (char)(((int)text.charAt(i) + s - 97) % 26 + 97);
	      result.append(ch);
	   }
	}
	return result.toString();
		}

		public String greetingMessage(Email e) {
			String frm = e.header.from;
			String name = frm.substring(0, frm.indexOf("@"));
			String greetMsg = e.greetings + " " + name;
```

```
		return greetMsg;
	}


	public static void main(String[] args) {
		Header header = new Header("amit@doselect.com","_ajay@gmail.com");
		Email email = new Email(header,"Have a nice day.","Regards");

		System.out.println(email.emailVerify(email));
		System.out.println(email.bodyEncryption(email));
		System.out.println(email.greetingMessage(email));
	}
}
```

******EXCEPTION CODE (QUESTION NOT PROVIDED - SOURCE 1)*******

```
class Employee
{
	private String firstName;
	private String lastName;
	private String ssn;
	public Employee()
	{

	}
	public Employee(String firstName, String lastName, String ssn)
	{
		this.firstName = firstName;
		this.lastName = lastName;
		this.ssn = ssn;
	}
	public String getFirstName()
	{
		return firstName;
	}
	public void setFirstName(String firstName)
	{
		this.firstName = firstName;
	}
	public String getLastName()
	{
		return lastName;
	}
	public void setLastName(String lastName)
	{
		this.lastName = lastName;
	}
	public String getSsn()
	{
		return ssn;
	}
	public void setSsn(String ssn)
	{
		this.ssn = ssn;
	}
	public String validateName(String firstName, String lastName)
	{
		if(firstName==null||lastName==null)
		{
			try
			{
```

```
                                        throw new NullPointerException("Entry Missing");
                                }
                                catch(NullPointerException ex)
                                {
                                        return ex.getMessage();
                                }
                        }
                        else if(firstName.length()==0||lastName.length()==0)
                        {
                                try
                                {
                                        throw new StringIndexOutOfBoundsException("Index out of bound");
                                }
                                catch(StringIndexOutOfBoundsException ex)
                                {
                                        return ex.getMessage();
                                }
                        }
                        else if(Character.isDigit(firstName.charAt(0))||Character.isDigit(lastName.charAt(0)))
                        {
                                try
                                {
                                        throw new IllegalArgumentException("First Character is invalid");
                                }
                                catch(IllegalArgumentException ex)
                                {
                                        return ex.getMessage();
                                }
                        }
                        else
                        {
                                return "Valid String";
                        }
                }
                public String validateSsn(String ssn)
                {
                        if(Character.isDigit(ssn.charAt(0)) && Character.isDigit(ssn.charAt(ssn.length()-1)))
                        {
                                return "Valid String";
                        }
                        else
                        {
                                return "Invalid String";
                        }
                }
}
public class Source
{
        public static void main(String[] args)
        {
                Employee emp = new Employee("Adarsh","Gupta","1AAAA8");
                System.out.println(emp.validateName(emp.getFirstName(), emp.getLastName()));
                System.out.println(emp.validateSsn(emp.getSsn()));
        }
}
```

******SOURCE 2****[REPEATED]**

```
import java.util.StringTokenizer;

class Employee
{
	String empName,empID,empDept;
	int salary;
	public Employee(String empName, String empID, String empDept, int salary)
	{
		this.empName = empName;
		this.empID = empID;
		this.empDept = empDept;
		this.salary = salary;
	}
}
class EmployeeDetails
{
	public Employee getEmployeeInfo(String str)
	{
		String empName = null,empID=null,empDept=null;
		int salary=0;
		String upd = null;
		StringTokenizer st = new StringTokenizer(str,".");
		while(st.hasMoreTokens())
		{
			empName = st.nextToken();
			upd = st.nextToken();
			System.out.println(upd);
			break;
		}
		empName = empName+" "+upd.substring(0,upd.indexOf("ID"));
		upd = upd.substring(upd.indexOf("ID"), upd.length());
		empID = upd.substring(2,upd.indexOf("DT"));
		upd = upd.substring(upd.indexOf("DT"));
		empDept = upd.substring(2,upd.indexOf("CTC"));
		upd = upd.substring(upd.indexOf("CTC")+3);

		salary = Integer.parseInt(upd.substring(0,upd.indexOf("L")));
		salary = salary*100000;

		new Employee(empName,empID,empDept,salary);
	}
	public String getEmployeeTaxSlab(Employee e)
	{
		if(e.salary>=1000000)
		{
			return "High";
		}
		else if(e.salary>=800000 && e.salary<1000000)
		{
			return "Medium";
		}
		else if(e.salary>=500000 && e.salary<800000)
		{
			return "Low";
		}
		else
		{
			return "None";
		}
	}
```

```
}
public class Last
{
	public static void main(String[] args)
	{
		EmployeeDetails ed = new EmployeeDetails();
		Employee e = (Employee)ed.getEmployeeInfo("Jimmy.RyanIDIE22IT023DTITCTC8L");
		System.out.println(ed.getEmployeeTaxSlab(e));

	}
}
```