Name of the Experiment :-

An election is contested by 5 candidates. The candidates are numbered 1 to 5 and voting is done by making the candidate number on the ballot paper. Write a program to read the ballots and count the votes cast for each candidate using an array variable count. In case a number read is outside the range 1 to 5, the ballot should be considered as a "spoilt ballot" and the program should also count the number of spoilt votes. The name of candidates and number of votes of each candidate are initially assigned during loading of the class holding the aforementioned scenario :-

Introduction :-

We have to design a class to count the valid and invalid ballots. And we have to design a driver class to use the previously designed class and count the ballots.

Objectives :-

- How to use array of objects
- How to use object oriented programming to solve problems.

Analysis :-

After analysing our problem we have found following components of our problem.

- We will have to define two arrays. Once array will count the ballots according to the number written on the ballot. and other array will store the name of the candidates.

- We will have to define two methods. One to count the ballots and one to display the candidates names with their corresponding vote count.

- We will have to design a driver class which will use the ballot counting class and use its methods.

Let us consider the name of the ballot counting class as 'Ballot Counter' and the name of the driver class as Main. From above analysis the conceptual class diagram is given in figure-1.

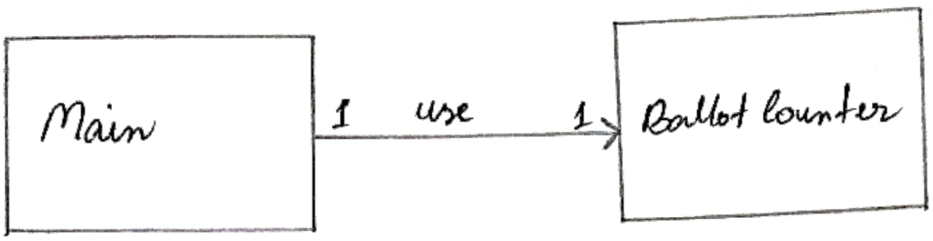

Figure-1 :- conceptual class diagram

Design :-

From the analysis above. our class design descriptions and architectural class diagram are given below:-

- class BallotCounter : Ballot counting class
  - ☐ Data Members:
    - count : an array of integers to count ballots ~~and increase the count~~
    - name : an array of strings containing candidates names.
  - ☐ Methods :
    - receiveBallot : this will receive ballots and increase the count.

- displayVotes : this will show the vote count with corresponding candidates name.

- class Main : the driver class, that will use BallotCounter class.

  - Methods :

    - main : the main function of the program from where the program will start.

The architectural class diagram is given figure-2

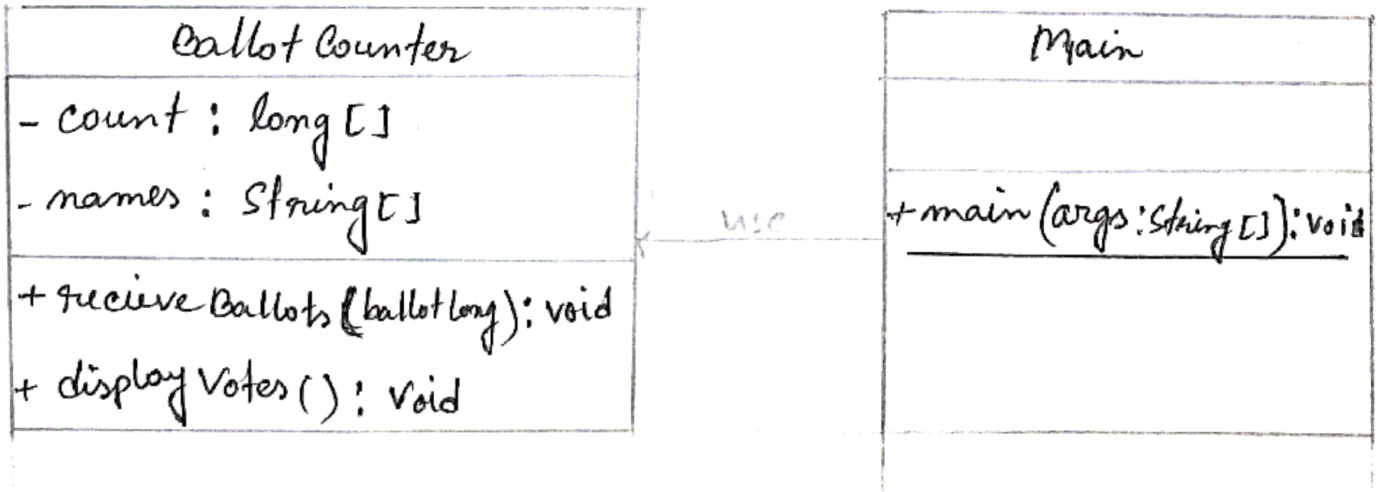

Figure-2 : Architectural class diagram.

The pseudo code of methods of the above mentioned classes are given below :-

receiveBallots (ballot) :

increase the corresponding counter of ballot;

displayVotes () :

print candidates name and their acquired votes, one by one

main (args) :

create BallotCounter object;
Count some ballots;
display vote counts.

Implementation :-

```
class BallotCounter {
private long [] count = new long [6];
private String [] names = { "candidate 1"
                          , "candidate 2"
                          , "candidate 3"
                          , "Candidate 4"
                          , "Candidate 5" (.
```

Implementation :-

```
class BallotCounter {
  private long [] count = new long [6];
static private String [] names = new String [5];

   Static {
   names [0] = "x"
   names [1] = "y"
   names [3] = "z"
   names [4] = "A"
   }
public void receive Ballot (long ballot) {
   if (ballot < 1 || ballot > 5)
      ++ count [0];
   else
      ++ count [ballot];
   }
public void displayVotes () {
for (int i = 1 ; i <= 5 ; ++i) {
System.out.println (names [i-1] + " got" + count [i] + "votes");
}
```

```
System.out.println("The spoit ballot count : "+count[0]);
}

class Main {
  public static void main(String[] args) {
  BallotCounter counter = new BallotCounter();
  Scanner sc = new Scanner(System.in);
  int totalBallots = sc.nextInt();
  for (int i=0; i<totalBallots; ++i) {
  counter.receiveBallots(sc.nextLong(),1);
   }
   counter.displayVotes();
  }
}
```

Conclusion :-

We designed a class BallotCounter to count the valid and invalid ballots. And we also designed a driver class Main to use the designed class and counted the ballots.

Name of the Experiment:-

Design a class including four methods where the first method does the following actions

a. To output the question "Who is the inventor of JAVA?"

b. To accept the answer.

c. To print out "Good" and then stop, if the answer is wrong.

d. To output the message "Try Again", the answer is wrong

e. To display the correct aswer when the answer is wrong even at the third attempt and Stop

The second method extracts a position of a charecter string and print the extracted string. The third method will read a text and count all occurrences of a particular of a particular word. Finally, the last method will read a string and re-write it in the alphabetic order. Write a test application that demonostrates the capabilities of designed class.

Introduction:-

We have to define a class with four methods as per the experiment description. And we also have to define a driver class that shows the capabilities of the designed class.

Objectives :-

- How to use object oriented approach to solve a problem
- How to use method calls.

Analysis :-

After analysing our problem we have found following components of our problem.

- We have to define a method to ask a question and check for the validity of the answer. User will get only 3 attempts.
- We have to define a method that will printout a string that has been extracted from a character string.
- We have to define another method that will take a text as an input and a particular word and will return the occurence of that word in that text.
- We have to define another method that will read a string and will print the string in alphabetic order.

- We have to define a class that will include above methods.
- We will also have to define a driver class that will use the defined class object and show its capabilities.

Let us consider the name of the designed class 'Task' and name of the driver class as Main.

From above analysis the conceptual class diagram is given in figure-1.

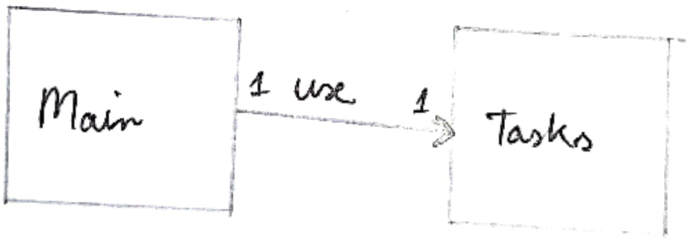

Figure-1 : Conceptual class diagram

~~Design~~ :-

From the above analysis our class design description are given below.

- class Tasks : our designed class
  - Methods :
    - method 1 : the first method as maintained in the description

Design :-

From the above analysis, our class design descriptions are given below :-

- class Tasks : our designed class

☐ Methods :

- inventor Query : the first method as mentioned in the description that is we ask about the inventor of java upto 3 times.

- extract string : this method will extract a portion of a string.

- count Occurrences : this method will count the occurrence of a word in a text.

- Write In Order : this method will print a string in alphabetic order

- class Main : the driver class, that will use the tasks class.

□ Methods :

- main : the main function of the program from where the program will start

The architectural class diagram is given in Figure-2.

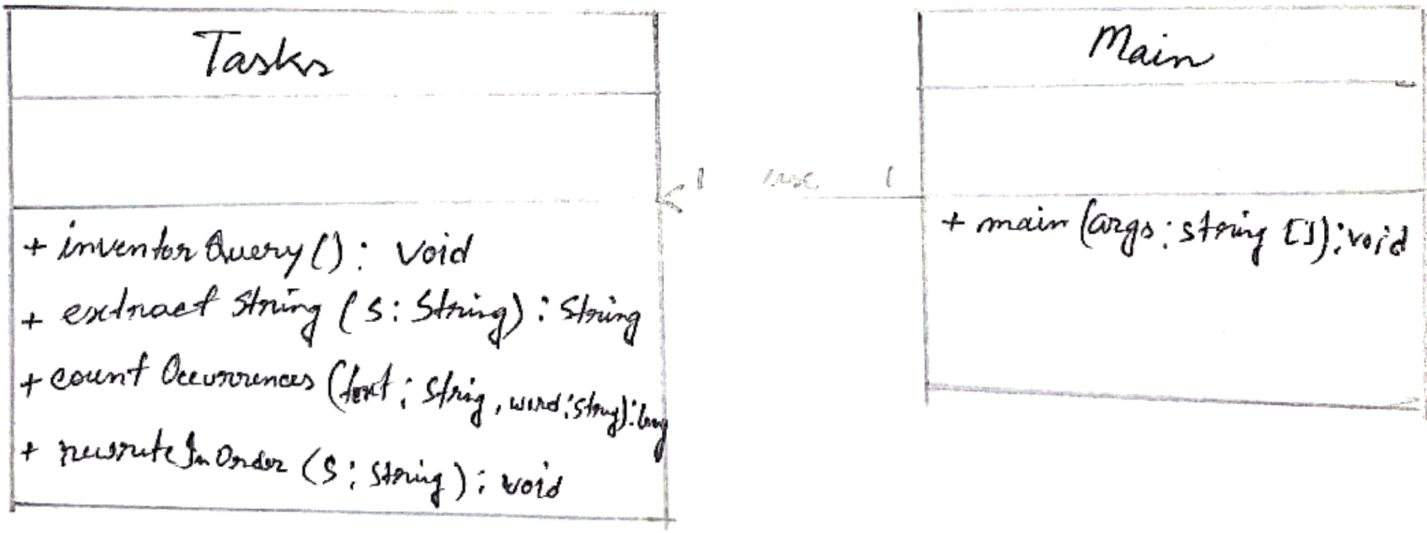

Figure-2: Architectural class Diagram.

The pseudo code of methods of the above mentioned classes are given below:-

```
inventorQuery():
print the question;
read users answer untill the answer is right or
for 3 attempts;

extract String(txt):
  extract a string and print it:

countOccurrences(text, word);
  count word. occurrences in text and return it:

rewriteInOrder(s):
print the string s in alphabetical order.

main(args):
  create Tasks object.
  use all the method on the created objects;
```

Implementation :-

```
class Tasks {
public void inventorQuery () {
String answer = " James Gosling";
System.out.println ("Who is the inventor of JAVA?");
scanner sc = new scanner (System.in);
int count = 0;
while (count != 3) {
if (answer.equals (sc.next()) {
System.out.println ("Good");
return;
}
else {
++ count;
System.out.println ("Try Again");
}
}
}
```

```
public String extract String (String text) {
  String ret = text. substring (0, v);
  System. out. println (" Extracted String : " + ret " );
  return ret;
}

public long count Occurrences (String text, String word) {
  long l = word. length (), count = 0;
  for (int i = 0; i < text. length () - l, ++i) {
    String t = text. substring < i, l+i);
    if (t. equals (word))
      ++ count;
  }
System. out. println (" Occurrences of " + word + ": " +
                                          + count);
  return count;
}
```

```
public void rewriteInOrder (String s) {
char [] a = s.toCharArray();
Arrays.sort(a);
String t = new String(a);
System.out.println("The string is alphabetic order"
                                        +t);
    s = t;
    }
}

Class Main {
  public static void main (String[] args) {
   Tasks t = new Tasks();
   t.inventionQuery();
   t.enterText String ("Java is an object oriented Language");
   Scanner se = new Scanner (System.in);
   t.countOccurrences (se.next(), se.next());
   t.rewriteInOrder (se.next());
    }
}
```

Conclusion :-

We designed a class Tasks with four method as per experiment description. And we also designed a Driver class to show the capabilities of the Tasks class.