Requirment for the Election console application

| Control Instruction | varibles,array | package and library | Predefined Methods of HashMap |
|---|---|---|---|
| A) while{}------->1<br>B) switch case{}->1<br>C) if(){}-------->3<br>D) for(){}------->1 | String str = scanner.nextLine()<br>String[] parts = str.split(" ")<br>String command = parts[0];<br><br>split(): return array of String | import java.util.HashMap;<br>import java.util.Scanner;<br>import com.google.gson.Gson;<br><br>predefined method of Gson class:<br><br>1)toJson(object of HashMap):<br><br>return string in JSON format<br><br>Referenced Libraries<br>gson-2.6.2.jar - C:\Users\tusha\Download | HashMap<String, Integer> candidates = new HashMap<String, Integer>();<br><br>1) put(k,v): For Insertion<br>2) containsKey(K):We check the key present or Not.returns true false<br>3)get(k): returns the value to which specific key is mapped.<br>4)keySet():return set view of keys. |

**Declare HashMap: initial capacity is 16 load factor 0.75**

```
HashMap<String, Integer> candidates = new HashMap<String, Integer>();
```

**Election class that contains following methods for each of the required commands.**

1) **The “entercandidate" method:**

can take a name as a parameter and save it in a HashMap with a count (vote count) initialized to 0

```
public  void entercandidate( String name)
    {
        candidates.put(name, 0);
        System.out.println("Candidate Name: " + name);
    }
```

Output:

```
 1.entercandidate 2.castvote 3.countvote 4.listvote 5.getwinner 6.exit

Enter your choice
entercandidate tushar
Candidate Name: tushar
```

```
 1.entercandidate 2.castvote 3.countvote 4.listvote 5.getwinner 6.exit

Enter your choice
entercandidate pavan
Candidate Name: pavan
```

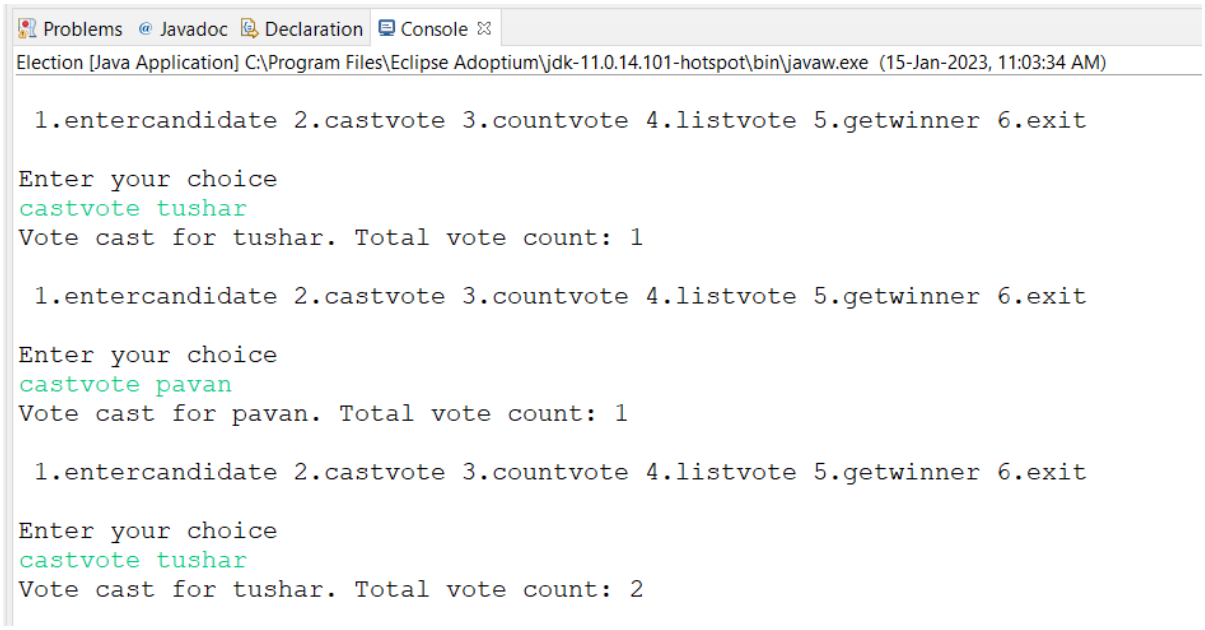
2) **The "castvote" method:**

The command castvote shall take a name as a parameter and increment the vote count and return it.

It should enter vote only for a valid candidate.

```
public int castvote(String name)
{
        if (!candidates.containsKey(name))
        {
            System.out.println("Sorry for Inconvenience, Please enter a valid candidate name.");
            return 0;
        }

        int voteCount = candidates.get(name);
        candidates.put(name, voteCount + 1);
        return voteCount + 1;
}
```

Output:



3) **The " countvote " method:**

The command countvote shall take a name as a parameter and should return the latest vote count.
Validate candidate name.

```
public int countvote(String name)
{
    if (!candidates.containsKey(name))
    {
        System.out.println("Sorry for Inconvenience, Please enter a valid candidate name.");
        return 0;
    }
    return candidates.get(name);
}
```

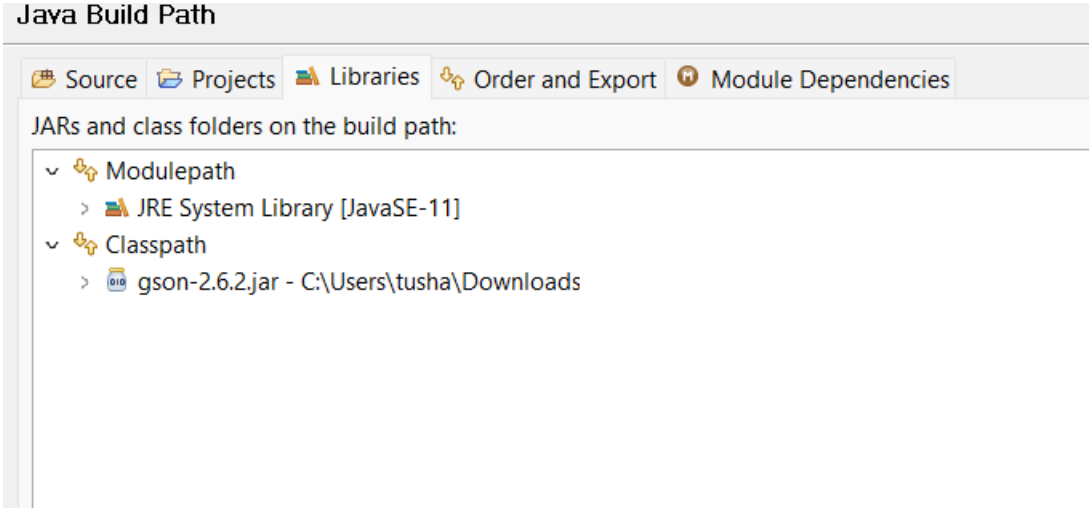
Output:

```
Enter your choice
countvote pavan
Vote count for pavan: 1

 1.entercandidate 2.castvote 3.countvote 4.listvote 5.getwinner 6.exit

Enter your choice
countvote tushar
Vote count for tushar: 2
```

4) **The " listvote " method:**

The command listvote should return all names and votecounts.
The return value is in JSON.

Add library:




```
public String listvote()
{
    Gson gson = new Gson();
    return gson.toJson(candidates);
}
```

Output: In JSON Format

```
 1.entercandidate 2.castvote 3.countvote 4.listvote 5.getwinner 6.exit

Enter your choice
listvote
{"pavan":1,"tushar":2}
```

5) **The " getwinner " method:**

The command getwinner should return the name of the candidate who got largest number of votes.

```
public void getwinner()
{
    String winner = "";
    int maxVotes = 0;
    for (String name : candidates.keySet())
    {
        int voteCount = candidates.get(name);//Returns the value to which the specified key is mapped
        if (voteCount > maxVotes)
        {
            maxVotes = voteCount;
            winner = name;
        }
    }
    System.out.println("Winner: " + winner + " with " + maxVotes + " votes.");
}
```

Output:

```
 1.entercandidate 2.castvote 3.countvote 4.listvote 5.getwinner 6.exit

Enter your choice
getwinner
Winner: tushar with 2 votes.
```

```
Enter your choice
exit
```

**Main Method:**

```
public static void main(String[] args)
    {
        @SuppressWarnings("resource")
          Scanner scanner = new Scanner(System.in);
        Election election = new Election();

        boolean flag=true;
        while (flag)
        {
          System.out.println(" ");
          System.out.print(" 1.entercandidate");
          System.out.print(" 2.castvote");
          System.out.print(" 3.countvote");
          System.out.print(" 4.listvote");
          System.out.print(" 5.getwinner");
          System.out.print(" 6.exit");
          System.out.println(" ");
          System.out.println(" ");
          System.out.println("Enter your choice");

           String str = scanner.nextLine();
            String[] parts = str.split(" ");
            String command = parts[0];


              switch(command)
              {
              case "entercandidate":

                 election.entercandidate(parts[1]);
                 break;


              case "castvote":
                   int voteCount= election.castvote(parts[1]);
                   System.out.println("Vote cast for " +
parts[1] + ". Total vote count: " + voteCount);
                    break;

              case "countvote":
                int totalVote= election.countvote(parts[1]);
                 System.out.println("Vote count for " +
parts[1] + ": " + totalVote);
                 break;

              case "listvote":
                 String str1=election.listvote();
                 System.out.println(str1);
```

```
                    break;

                case "getwinner":
                    election.getwinner();
                    break;

                case "exit":
                    flag=false;
                    break;

                default:
                     System.out.println("Invalid Choice");
                }
            }
        }
}
```