Objective:

- This lab should help in gripping the concept of object manipulation involved while object transition in case of aggregation/composition and array of objects.

Challenge: ***Voting Machine***

You have dealt this task already but using C structs.
Let's redesign the task in light of Object Orientation.

```
class Candidate
{
    String name;              //name of candidate
    String electionSymbol;    //election symbol of candidate
    unsigned int votes = 0;   //number of votes casted to candidate

public:
    Candidate(String n="", String es=""):name(n), electionSymbol(es)
    {}
    void castVote();          //increment by 1 in the vote count in *this object
    unsigned int getVotes(); //getter for votes
    void setName(String);
    String getName();
    void setElectionSymbol(String);
    String getElectionSymbol();
};

class VotingMachine
{
    String constituencyName;
           // like NA-105 or PP-404
    Date electionDate;
           // stores election date. Same struct as we used previously.
           // Candidates can't be added in machine on the electionDate.
    Time startTime;
           // stores election start time.
    Time endTime;
           // stores election end time.
    Candidate * candidateList = nullptr;
           // points to an array of candidate objects contesting in elections
    unsigned int numOfCandidates = 0;
           // number of candidate objects stored in machine
    unsigned int capacity = 0;
          // capacity of machine to store candidates in machine. i.e., it stores the size of array
             pointed by candidateList.

    Candidate * result = nullptr;
          //points to the array returned by electionResult function. It will free the user to take care
             of any memory deallocation of heap allocated array. Before returning the array which
             contains the result of top 3 candidates, its address will be stored in it, which will be
             deleted in Voting Machine destructor. If gets time in the course, we shall see more elegent
             way of dealing with such issues: named as Smart Pointers :)

    void reSize();
           // resizes the array pointed by candidateList by doubling the capacity.
};
```

Public Operations for the `VotingMachine class`

1. `VotingMachine (String constName, Date d, Time sTime, Time eTime)`

```
It initializes the voting machine with given data.
    constituencyName will be initialized with constName.
    electionDate of voting machine with d.
    startTime of voting machine with sTime.
    endTime of voting machine with eTime.
You may assume that sTime will always be less than eTime
```

2. **bool addCandidate (String candName, String elecSymb);**
   Add the candidate object with given candidate name and election symbol in the candidateList. Make sure that a candidate with duplicate election symbol must not be added. It also resize the candidate array if needed.
   It returns true if candidate is added successfully otherwise false.
   Candidates can't be added in machine on the electionDate.

3. **bool removeCandidate (String elecSymb);**
   Removes the candidate object with given election symbol from the candidateList.

4. **bool castVote(String elecSymbol);**
   It searches the candidate in Voting Machine with given election symbol and increments vote count by calling castVote function on the particular candidate object.
   Note: This function will not cast vote if the system date is not equal to the date stored in voting machine. Same goes for the time, the time of cast vote must be within the range of voting machine start time and end time.
   It returns true if vote is casted successfully otherwise false.

5. **int candidateReport(String elecSymbol);**
   It returns the number of votes casted so far to the candidate whose election symbol is received. Return -1 if elecSymbol is not found.

6. **Candidate * electionResult ();**
   It returns a heap allocated array of candidates having 3 objects with winner at index 0, runner-up at index 1 and 3rd position holder at index 2.

7. **~VotingMachine ();**
   Free the resources.

8. Some Getter/setters
   ```
   void setElectionDate(Date);
   Date getElectionDate();
   void setElectionStartTime(Time);
   Time getElectionStartTime();
   void setElectionEndTime(Time);
   Time getElectionEndTime();
   void setConstituencyName(String);
   String getConstituencyName();
   unsigned int getNumOfCandidates();
   ```

Note:

- class **Date**, and class **Time** are the ones that we used recently. You are free to add any **related** functionality in them rather you are encouraged to add any time and date related functionality in Time class and Date class respectively like comparing two dates or comparing two times.
- Use String class with latest interface along with operators =, +, +=, and []. + operator in place of function 'concatenate', += operator in place of concateEqual and [] operator in place of function 'at'.

Sample Run

```
int main()
{

    VotingMachine vMachine("PP-404", Date{7,12,2021},
Time{0,0,0}, Time{18,0,0});


    vMachine.addCandidate("Aslam", "Racket");
    vMachine.addCandidate("Naeem", "Kulhara");
    vMachine.addCandidate("Ayesha", "abc");
    vMachine.addCandidate("Rabia", "zzz");
    vMachine.addCandidate("Abdul Wahab", "Hockey");
    vMachine.addCandidate("Manan", "TV");

    vMachine.castVote("zzz");
```

Console Output

```
Vote count so far for Hockey: 4

*******Election Results*******
1st Position: Hockey : 6
2nd Position: zzz : 4
3rd Position: Racket : 1
```

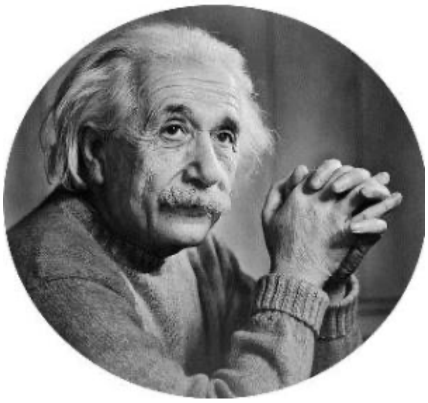
```
    vMachine.castVote("Kulhara");
    vMachine.castVote("zzz");
    vMachine.castVote("zzz");
    vMachine.castVote("Hockey");
    vMachine.castVote("Hockey");
    vMachine.castVote("zzz");
    vMachine.castVote("Hockey");
    vMachine.castVote("Hockey");

    cout<<"Vote count so far for Hockey:
"<<vMachine.candidateReport("Hockey");

    vMachine.castVote("Hockey");
    vMachine.castVote("Hockey");
    vMachine.castVote("TV");
    vMachine.castVote("abc");
    vMachine.castVote("Racket");


    Candidate * list = vMachine.electionResult();
    cout<<"\n\n*******Election Results*******\n";
    cout<<"1st Position: ";
list[0].getElectionSymbol().display(); cout<<" :
"<<list[0].getVotes()<<'\n';
    cout<<"2nd Position: ";
list[1].getElectionSymbol().display(); cout<<" :
"<<list[1].getVotes()<<'\n';
    cout<<"3rd Position: ";
list[2].getElectionSymbol().display(); cout<<" :
"<<list[2].getVotes()<<'\n';
    cout<<"\n\n";
    return 0;
}
```

Note: You must paste both .h and .cpp files in MS Word and then in pdf.



It's not that I'm so smart; it's just that I stay with problems longer.

- Albert Einstein -