



- 145 Removals + 102 Additions

12.cpp

```

1 #include <iostream>
2 #include <iomanip>
3 #include <string>
4 #include <sstream>
5
6 using namespace std;
7
8 const int maxCandidate=10;// of candidate max
9 int candidate=0;// of candidate we count
10 string nameList[maxCandidate]; //candidate name list
11 int
12 voteList[maxCandidate]={0,0,0,0,0,0,0,0,0,0}; //vote list
13 int tmpVoteList[maxCandidate]={0,0,0,0,0,0,0,0,0,0}; //temp
14 vote list b4 check its valid or not
15 int total=0; //total vote
16 int valid=0; //valid vote
17 int spoilt=0; //spoilt vote
18 int voteAllow=0; //vote allowed each person
19 int countUp=0; //temp vote count up
20 int argFlag=0; //check if theres a argument or not
21 int voteNow=0; //a
22 boolean tells we done with names lets begin with count vote
23 s
24
25 /***** validVote *****/
26 Purpose: This function will be called after read a valid
27 d vote
28 Returns: This function returns nothing
29 *****/
30 void validVote()
31 {
32     for(int j=0; j<candidate; j++)
33     {
34         voteList[j]+=tmpVoteList[j];
35         //cout << tmpVoteList[j];
36         tmpVoteList[j]=0;
37     }
38     //cout << endl;
39     countUp=0;
40     valid++;
41     total++;
42 }
43
44 /***** spoiltVote *****/
45 Purpose: This function will be called after read a spoilt
46 lt vote
47 Returns: This function returns nothing
48 *****/
49 void spoiltVote()
50 {
51     for(int j=0; j<10; j++)
52     {
53         tmpVoteList[j]=0;
54         countUp=0;
55         spoilt++;
56     }
57 }

```

```

1 #include <iostream>
2 #include <iomanip>
3 #include <string>
4 #include <sstream>
5
6 using namespace std;
7
8 const int maxChoice = 10; // of numchoice max
9 int numchoice = 0; // of numchoice we count
10 string names [maxChoice]; //numchoice name list
11 int
12 numVotes [maxChoice] = {0,0,0,0,0,0,0,0,0,0}; //vote list
13 int tempVotes [maxChoice] = {0,0,0,0,0,0,0,0,0,0}; //temp
14 vote list b4 check its validVotes or not
15 int totalVotes = 0; //totalVotes vote
16 int validVotes = 0; //validVotes vote
17 int spoiledVote = 0; //spoiledVote vote
18 int allowedVotes = 0; //vote allowed each person
19 int incCount = 0; //temp vote count up
20 int isFlag = 0; //check if theres a argument or not
21 int startVote = 0; //a
22 boolean tells we done with names lets begin with count vote
23 s
24
25 // This fucntion reads valid votes and counts them
26
27 void readValidVotes() {
28     for(int i = 0; i < numchoice; i++) {
29         numVotes[i]+=tempVotes[i];
30         tempVotes[i]=0;
31     }
32 }
33
34 incCount=0;
35 validVotes++;
36 totalVotes + +;
37
38 }
39
40 //This function reads votes and counts spoiled votes
41
42 void spoiledVoteCount() {
43     for(int i=0; i<10; i++) {
44         tempVotes[i]=0;
45     }
46     incCount=0;
47     spoiledVote++;
48     totalVotes++;
49 }

```

```

48     total++;
49 }
50 /***** checkVote *****/
51     Purpose: This function checks if the vote is valid or not
52     Returns: This function returns nothing but will change
53     global variables which counts votes
54     *****/
55 void checkVote(string ticket)
56 {
57     int voteVoter=0;
58     int countUp=0;
59     stringstream maStringStream(ticket);
60     int n;
61     while(maStringStream >> n)
62     {
63         voteVoter++;
64         tmpVoteList[voteVoter-1]+=n;
65         countUp+=n;
66     }
67     if((countUp>voteAllow) || (voteVoter != candidate))
68         spoiledVote();
69     else
70         validVote();
71 }
72 /***** addCandidate *****/
73     Purpose: This function consume a string which is candidate's name and we store it in the name list
74     Returns: This function returns nothing but will change
75     global variable which is the name list
76     *****/
77 void addCandidate(string name)
78 {
79     if(candidate<10)
80     {
81         nameList[candidate]=name;
82         candidate++;
83         if(argFlag==0)
84             voteAllow++;
85     }
86 }
87 /***** nameOrVote *****/
88     Purpose: This function consume a string and check if it is a candidate name or vote
89     Returns: This function returns nothing but will call vote/name function
90     *****/
91 void nameOrVote(string line)
92 {
93     stringstream maStringStream(line);
94     int n;
95     if(voteNow==1)
96         checkVote(line);
97     else
98     {
99         if(maStringStream >> n)
100         {
101             checkVote(line);
102             voteNow=1;
103         }
104     }
105 }
106 else

```

```

39 }
40
41 //This function checks if votes are valid
42 void validVoteCheck(string t) {
43
44     int voteVoter=0;
45     int incCount=0;
46     stringstream stream(t);
47     int n;
48     while (stream >> n) {
49
50         voteVoter++;
51         tempVotes[voteVoter-1]+=n;
52         incCount+=n;
53     }
54     if ((incCount>allowedVotes) || (voteVoter != numchoice))
55         spoiledVoteCount();
56     else
57         readValidVotes();
58 }
59 // this function stores names into the name list.
60 void addchoice(string name) {
61
62     if(numchoice<10) {
63         names[numchoice]=name;
64         numchoice++;
65         if(isFlag==0) allowedVotes++;
66     }
67 }
68 //function checks if string is name or vote
69 void checkNameVote(string s) {
70     stringstream stream(s);
71
72     int n;
73     if(startVote==1)
74         validVoteCheck(s);
75     else {
76         if(stream >> n) {
77             validVoteCheck(s);
78             startVote=1;
79         }
80     }
81 }
82 else

```

```

103     addCandidate(line);
104 }
105 }
106 /***** readVotes *****/
107 Purpose: This function read inputs and
108 call nameOrVote to let it decide whatever its a name or vote
109 Returns: This function returns nothing
110 *****/
111 void readVotes()
112 {
113     int i;
114     int voteVoter=0;
115     //cin >> i;
116     string s;
117     getline(cin,s);
118     while(!cin.eof())
119     {
120         nameOrVote(s);
121         // cout << s << endl;
122         getline(cin,s);
123     }
124 /***** printResults *****/
125 Purpose: This function print all
126 the information we get from cin and count
127 Returns: This function returns nothing
128 *****/
129 void printResults()
130 {
131     cout << "Number of voters: " << total << endl;
132     cout << "Number of valid ballots: " << valid << endl;
133     cout << "Number of spoilt ballots: " << spoilt
134     << endl << endl;
135     cout << left << setw(15) << "Candidate"
136     << right << setw(3) << "Score" << endl << endl;
137     for(int i=0; i<candidate; i++)
138     {
139         cout << left << setw(15) << nameList[i]
140         << right << setw(3) << voteList[i] << endl;
141     }
142 }
143 /***** main *****/
144 Purpose: Main
145 Returns: state
146 *****/
147 int main(int argc, char *argv[])
148 {
149     if(argc == 2)
150     {
151         stringstream maArg(argv[1]);
152         maArg >> voteAllow;
153         argFlag=1;
154     }
155     readVotes();
156     printResults();
157     return 1;

```

```

78     addchoice(s);
79 }
80 }
81 //this function reads inputs and
82 calls checkNameVote to decide if it's a name or a vote
83 void readVotes() {
84
85     int voteVoter=0;
86
87     string s;
88     getline(cin,s);
89     while(!cin.eof()) {
90         checkNameVote(s);
91
92         getline(cin,s);
93     }
94 //this function prints all info
95 void printResults() {
96
97     cout << "# of votes = " << totalVotes << endl;
98     cout << "# of valid votes: " << validVotes << endl;
99     cout << "# of spoiled votes: " << spoiledVote
100     << endl << endl;
101     cout << left << setw(15) << "numchoice"
102     << right << setw(3) << "Score" << endl << endl;
103     for(int i=0; i<numchoice; i++) {
104         cout << left << setw(15) << names[i]
105         << right << setw(3) << numVotes[i] << endl;
106     }
107 }
108 int main(int argc, char *argv[]) {
109     int result = 1;
110     if(argc == 2) {
111
112         stringstream maArg(argv[1]);
113         maArg >> allowedVotes;
114         isFlag = 1;
115     }
116     readVotes();
117     printResults();
118     return result;

```

155 }

112 }