Felipe Lima - 109290055

ElectoralMap	**calls**
ElectoralMap();	
static *ElectoralMap*& GetInstance()	ElectoralMap();
ElectoralMap(*ElectoralMap* const&);	
void operator=(*ElectoralMap* const&);	
void InsertDistrict(*District* &*d*);	
District get_district(int id)	
int GetMapSize()	
operator<<	

Fields/Attributes:	std::map<int, District> districts_;
	std::map<int, *District*> :: iterator it;
	int map_size_;

District	**calls**

District()	*ElectoralMap*::GetInstance()
	GetMapSize()
	AreaGenerator()

int AreaGenerator();	
int get_area()	
int get_constituents(Party &p)	
void MoveConstituents(Party &p1, Party &p2)	
operator<<	

Fields/Attributes:	std::*string* district_name_;
	int district_id;
	int area_;
	std::map<Party, int> constituents_;

TextUI	**calls**
ElectionType()	
RegisterCandidates()	Election::RegisterCandidates()
CandidatesCampaining()	Campaign()
PrintCandidates()	

PrintResults()	Election::CalculateVotes()
	RepresentativeElection::CalculateVotes()

Fields/Attributes:	std::string election_type_ static bool end_election;

Party	**calls**
enum class	no calls

Candidate	**calls**
struct	no calls

Fields/Attributes:	std::string name;
	Party party_affiliation;
	int id;

Election	**calls**
RegisterCandidates()	
void Campaign()	District::MoveConstituents(Party &p1, Party &p2)
CalculateVotes()	District:: get_constituents(Party &p)

RepresentativeElection	**calls**
CalculateVotes()	District:: get_constituents(Party &p)

int main.cpp

```
int main(){
        TextUI ui;
        while (end_election == 0){
                ui.ElectionType();
                ui.RegisterCandidates();
                ui.CandidatesCampaining();
                ui.PrintCandidates();
                ui.PrintResults();
        }
}
```

// main goes in a loop until user selects to exit program.
// only calls Ui because all the other calls are handled by the methods in TextUI