**Hacettepe University**

**Computer Science and Engineering Department**


| | | |
|---|---|---|
| **Name and Surname** | : | İrem Dereli |
| **Identity Number** | : | 21727135 |
| **Course** | : | BBM 203 |
| **Experiment** | : | Assignment 3 |
| **Subject** | : | Linked Lists |
| **Programming Language** | : | C++ |
| **Compiled With** | : | C++11 |
| **Due Date** | : | 16.12.2018 |
| **e-mail** | : | iremm.dereli@windowslive.com |

## Problem Definition

In this assignment, the main problem was creating one singly linked list and one doubly linked list. There are input files to read, one of them includes some footballer's name, team name and the details of given footballer's match. Purpose is put these datas into linked list according to sequential of footballer's name without repeating any footballer again and again. After completing creation of linked lists, there are 8 calculations to finish the assignment.

## Solution

Firstly, initialized two structures for footballers and details of matches. These structures include some properties for footballers and matches. Using these structures, initialized a singly linked list for footballers. After that initialized a doubly linked list for match details. There is a global head for footballers linked list to hold first item of list. When completed linked list initialization, started to read input files named "input" that includes footballers and match details. Adding these matches and footballers to linked lists was done with some controls. Like, if a footballer is already added to footballer's linked list, it should not be added again. Only his/her match details should be added.

When completing these adding processing, there are some calculations to program must be do. These calculations are done by functions as stated below.

When every function is called, outputs are written in output file. Finally, all files will be closed, and program ends.

## Functions Implemented

- **matchDetails* insertMatchDetails(char awayTeam[], int minuteGoal, int matchID):** Function gets away team name, minute of goal and match id as parameter. Creates a new matchDetails struct that named temp to put these parameters. Returns this temporary.

- **void insertFootballer (char name[], char teamName[], char awayTeam[], int minuteGoal, int matchID):** This function also gets footballer's name, footballer's team name, away team, minute of goal of this match and match id. Function put these parameters into a temporary element of Footballer struct. Also creates temporary2 for holding and checking for head (without do any change on head item). This function calls when the footballer is not in footballer's linked list, I mean, there is no match of this footballer's played before. So, details of match must be head element of this footballer. And next element of head must be NULL at first
  If head element (footballers linked list's first element) is NULL, it means that this footballer is first item, so head must be this element. And function finishes.
  If not, function checks the alphabetic order of elements and put the new element the right place.

- **void read(char content[]):** Function gets only a line of a input file as parameter. Splits the line by comma and put these new items into variables.
  Starts to check temporary's name is equal to new name or not; if is equal, if footballer's head is NULL, puts the new match details into head. If not, checks the last item of footballer's doubly linked list and puts the details of match at last of list. If there is no match up with footballer's name, creates new footballer into list of footballers.

- **int maxElement(int array[], int count):** Gets an array of footballer's goals inside as a parameter and number of footballers. Equalizes max variable into first element of the array, and checks if there are greater number is existing or not. Returns the maximum number.

- **void topGoal(int count, ofstream &outfile):** Creates an array to hold total number of goals of footballers. Array size is given as parameter (count). Initializes array with checking linked lists. After finishing these controls, writes the name of the footballers that his/her goal number is equal the maximum number into output file that given as parameter. Maximum number of goals is calculated by maxElement method.

- **void goal(ofstream &outfile):** Creates two new variables named that countFirst and countSecond for counting the goal numbers of the first half of match and second half of match of all given matches. Counting calculation is done by checking the linked lists of matches. After initializing countFirst and countSecond, checks the which one is greater or two of them is equal. Prints into output file.

- **void hattrick(int count, ofstream &outfile):** Creates a vector for hold the which footballer did hat trick. Vector is used to not print a footballer again and again.
  Function checks all matches by match id, number of matches are given by parameter. If a footballer goaled in a one match 3 or more, checks if there is footballer in vector, does not print again that footballer. If there is not, prints the footballer into output file.

- **void printTeams(int count, ofstream &outfile):** Function creates a vector for hold the teams. Checks all footballers teams and if the team is not in that vector, prints the name of team into output file.

- **void printFootballers(of stream &outfile):** Prints the all names of footballers from singly linked list into output file.

- **void printMatches(Footballer *ff, ofstream &outfile):** Gets the linked list of given footballer as a parameter, and prints all matches of given footballer into output file.

- **void printMatchID(Footballer *ff, ofstream &outfile):** Prints the given footballers matches sequential of match id into output file.

- **void printMatchIDReversed(Footballer *ff, ofstream &outfile):** Prints the given footballers matches reverse sequential of match id into output file.

- **<u>void readOperations(char content[], int count, ofstream &outfile):</u>** Gets the names of given in operations file. If count is equal to 0, calls the printMatches function, if equal to 1, calls the printMatchID, else calls printMatchIdReversed function.

## Functions not Implemented

In this assignment, I did not implement function for opening input files and output file. Reading lines is done in main function. Also calculating the number of matches is done in main function. I called all other functions in main function. And closed files in main.