Hacettepe University Computer Science Department
BBM203 Assignment 3

Name and Surname	: Ismet OKATAR
Identity Number	: 21727542
Course	: BBM203 Software Laboratory II
Experiment	: Assignment 3
Subject	: Linked Lists
Date Due	: 14.12.2018
E-mail	: b21727542@cs.hacettepe.edu.tr

3. Software Design Notes
3.1 Description of the program

This is a simple search engine which`s data structure is singled and doubled linked lists. This program creates an linked list due to the given input.txt file. with that program you can print all futbollers, all team names , the futboller who hat scored a hat-trick. And you can also print a futboller`s match data`s which`s are in ascending or descending order.

3.1.1 Problem

There must be two linked lists first single linked list and the second doubly linked list. They must be filled with the given data.

And then we have to print;
1. The period in which the most goals are scored in the league
2. Find the top goal scorer and print his name on the screen.
3. To print the names of footballers who scored hat-trick.
4. Print the team list in the league
5. Print the list of footballers
6. Matches and goals of given footballer
7. Sort by match ID in ascending order of given footballer
8. Sort by match ID in descending order of given footballer

3.1.2 Solution

We have to create two structs first for upper linked list which is singled linked list and second for inner linked list which is doubled linked list.
And then we have to write access codes to that linked lists. Like;
Add,
Delete,
Print,
SortedAdd;
Get
Then we have to read the data line by line from the input file and split it and add it to the linked list.
Once data loading is done we can step to the operations part.

1.operation;
Look for each data and get the goal time. if second half is greater print 1 else print 0
2.operation;
Look for each data and get the top scorer. if there is more than one print everyone.
3.operation;
Look for each data and get the futbollers who has 3 or more goals in one game and print his name.
4.operation;
Look for each data and get the all of the team names and print them.
5.operation;
Look for each data and get the all of the futboller names and print them.
6.operation;
Look for each data and get the given futbollers matchs and goals
7.operation;
Look for each data and get the given futbollers matchs its also will be in ascending order.
8.operation;
Look for each data and get the given futbollers matchs .It must be in descending order. Thats why print it by using prev pointers.

3.2 Main Data Structures

First the upper linked list.

```
typedef struct node{ // struct of first linked list
   char   futbolName[100]; // futboller name
   char   teamName[100]; // team name
   struct Node * head; // a pointer to the inner linked lists
head
   node * next;  // pointer to next linked list
}* nodePtr;
```

Second the inner linked list.

```
struct Node { // struct of second linked list
   char awayteam[100]; // away team name
   int  mofgoal;       // minute of goal
   int  macId;         // match id
   struct Node* prev, *next;  // pointer to next and previous
linked list
};
```

3.3 Algorithm

Oncelikle fonkisyonlarimiza bir bakalim.

```
void Addnode(char futbolName[100],char teamName[100]):
Input: Data
result: The linked global link adds a new node.
void Printnode()
No entry
result: Print our Linked list.
nodePtr getnode(int y)
```

```
Input: Linear position in Linked list
result: returns the pointer of the element there.
int searchnode(nodePtr head, char * aranan)
input: the head pointer and the searched name are entered.
Result: if there is no element in the location it returns -1.
struct Node* getNode(char awayteam[100],int mofgoal,int macId)
Input: data
Result: creates the desired element and returns the pointer.
void sortedInsert(struct Node** head_ref, struct Node* newNode)
Introduction: element
result: insert given element as sorted
int main(int argc, char *argv[])

open argv[1] file ;
for there_is_more_lines:
     read a line and add it to lines[] array;
sort the lines[] array
while line in lines[]:
     split the line for comma;
     and add it to {
          char name[100];
          char team[100];
          char awayteam[100];
          int mofgoal;
          int macId;
     }
// Reading and splitting is done
if there is no node in name of name[] in linked list
     add it;
get the added linked list and sortedInsert the data for every
line.
// Linked list creation has done
//1. Operation
for all upper nodes
     for all inner nodes
```

```
     get the values of minute of goal;
if second half more
     print 1
if first half more
     print 0
//2. Operation

for all upper nodes
     for all inner nodes
          get the total number of nodes for each uppernode;
     add them to an array for comparing the goal numbers.
for in goalArray
     compare pair by pair and find the most goaled futboller;
for in players
     print all futbollers name that have the same number of
goal with the most goaler.

//3. Operation
for all upper nodes
     for all inner nodes
          get the goal number of all players and print  when it
is more than or equal to 3;

//4. Operation
for all upper nodes
     if that team name is not printed yet
          print it to screen
//5. Operation
for all upper nodes
     if that player name is not printed yet
          print it to screen
//6. Operation
read a line from argv[2]
split it by comma;
for in splitted_line
```

```
     for all upper nodes
          for all inner nodes
               print the data of it
//7. Operation
read a line from argv[2]
split it by comma;
// The linked list is already in ascending order.
for in splitted_line
     for all upper nodes
          for all inner nodes
               print the data of it
//8. Operation
read a line from argv[2]
split it by comma;
for in splitted_line
     for all upper nodes
          for all inner nodes
               get the head pointer to the last node
          for all inner nodes (by using ->prev pointer)
               print the data of it
```