

CSE102 – Computer Programming (Spring 2018)

Homework #7

Handed out: 01:00 pm Saturday April 20, 2018.

Due: 11:30 pm Monday April 30, 2018.

Hand-in Policy: Via Moodle. No late submissions will be accepted.

Collaboration Policy: No collaboration is permitted.

Grading: This homework will be graded on the scale of 100.

Part 1 (40 points): In this part, you may do this question in the previous homework. You may use some parts of the previous homework 5 for this question. But be careful about step 7!

Description: In this homework, you will implement Turkish Draughts board game. The rules of the game are listed below.

- 1) There is a 8x8 board.
- 2) Each player starts with 16 men (white and black) placed on all the squares of the second and third rows (see Figure 1). All the squares are used.
- 3) Men's moves: They move one square horizontally or vertically forwards, never backwards (see Figure 2).
- 4) Kings moves: They move and jump vertically and horizontally any number of squares (see Figure 2).
- 5) A man captures by jumping to a vacant square in the moveable direction (rule 3) beyond a piece of the opponent.
- 6) A king captures by jumping to a vacant square beyond an adverse piece, any distance away.
- 7) Maximum capture compulsory: If there are two or more different chances to make capture, it is compulsory to make the move that captures the maximum amount of the opponent's pieces. **You must use recursion algorithm to find Max. capture. Any other solutions are not accepted.**
- 8) Taking away the captured pieces: Captured pieces are taken away from the board.
- 9) If a player has no legal move he loses the game. This may come about either by being eliminated or being blocked completely – no moves left.

You must use the functions listed below. You may add other functions if you think there is need for it.

typedef enum {white_man, black_man, white_king, black_king, empty} piece;

You must use this enumeration to define the pieces on the board.

typedef enum {white = 10, black = 20} player;

You must use this enumeration to define the players.

void init_board(piece board[][8]);

This function initializes the board as mentioned in the second rule.

int move(piece board[][8], int from_x, int from_y, int to_x, int to_y, player p);

This function checks if the given player p can move a piece belonging to it from location (from_x, from_y) to location (to_x, to_y). If the players move is not allowed the function returns:

- -1: The player p is trying to move a piece that does not belong to him.
- -2: The move is not allowed.
- n>=0: The move is allowed and it is executed with n of opponents pieces captured.

int check_end_of_game(piece board[][8]);

This function checks if the game has been completed after the move. It returns:

- -1: Game continues.
- white: White wins the game.
- black: Black wins the game.
- **void display_board(piece board[][8]);**
- This function displays the board in current state. The following symbols should be used for printing:
- ‘-’: Empty squares.
- ‘b’: The regular black pieces.
- ‘B’: The black kings.
- ‘w’: The regular white pieces.
- ‘W’: The white kings.

Sample outputs are shown below:



void sample_game_1();

This function should use the above functions to play end-to-end a game. You may want to find a

famous game and replay it in this function.

void sample_game_2();

This function should use the above functions to play the game given in the following gif animation.

http://damaakademisi.com/wp-content/uploads/2009/02/10puan_1.gif

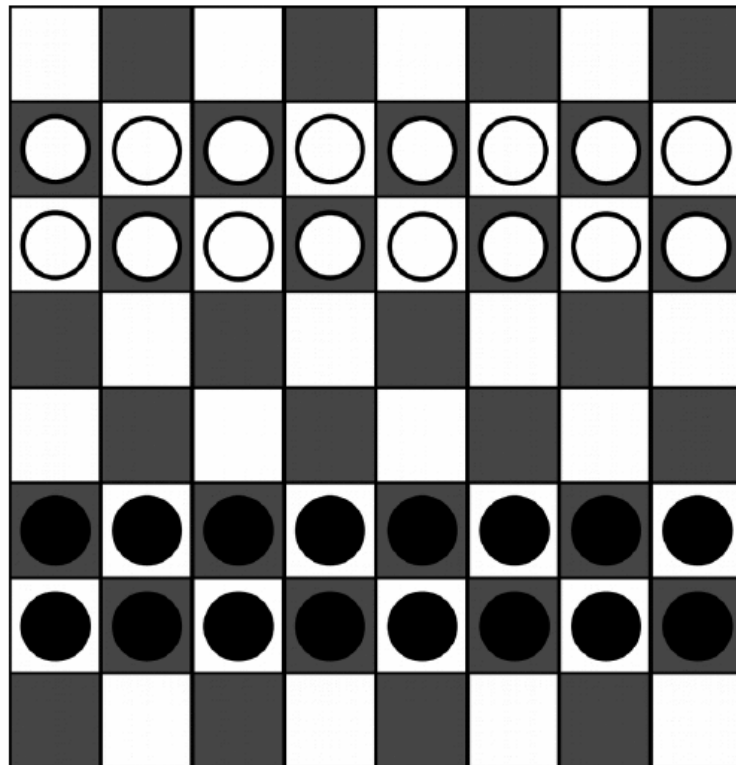


Figure 1 Starting position.

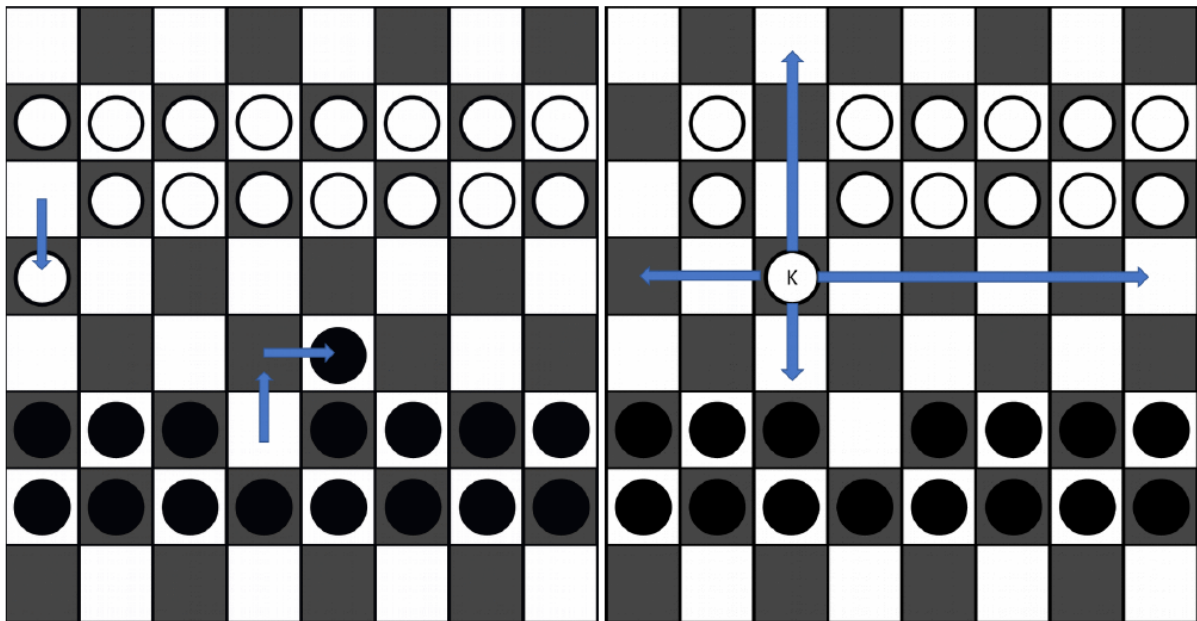


Figure 2 Men's (left) and king's (right) moves.

Part 2 (60 points):

In this part

- You create *input_date.txt* input file using C programming function. This *input_date.txt* file will consist of dates in dd/MM/yyyy format, that include **all days** between a start and an end date entered by user. For example, if user enter 30/05/2015 as a start date and 30/05/2018 as an end date your function should generate all dates (day by day) between these two dates.
- Write a C programming function which read each line from "input_date.txt" file, convert date format (dd/MM/yyyy) to (dddd, MMMM dd, yyyy) format and write this new formatted data into *new_date.txt* line by line.

For example, if *input_date.txt* includes 30/05/1978 (dd/MM/yyyy), your program converts 21/04/2018 (dd/MM/yyyy) to Saturday, April 21, 2018 (dddd, MMMM dd, yyyy) and write this new formatted data into *new_date.txt*.

For any questions: Mrs Altan Akin, tualtan@gtu.edu.tr

Good Luck!

I

Your submission (studentnumber_hw08.zip) should include the following files **and NOTHING MORE** :

input_date.txt, *new_date.txt*, screenshots of your running programs step by step with explanations as a **HW7.pdf** file and

HW07_<student_name>_<studentSurname>_<student number>_part1.c

HW09_<student_name>_<studentSurname>_<student number>_part2.c