Department of Computer Science **FAST-NUCES** **Lahore Campus**

CS401 Artificial Intelligence

Programming Assignment No 1 (Sections C, D, E) Spring 2019
Assigned on 11/2/2019 Deadline 25/2/2019 before 5:00 p.m.
Submission: Online on SLATE Weight 3%

Instructions:

Following rules will be enforced for this assignment

- You might work in a group of at most Two students having same gender (Across-section group are allowed)
- You need to decide a name of your group (Like Champions, Brutal, Brute Force etc. and submit the group name on slate before Thursday 14th of February Morning 8:00 am)
- Your group will submit a detailed report explaining your strategy (Evaluation Function) of playing the game.
- Late submission will be allowed but the following deductions will be applicable
  - 20 % deduction on submissions within 24 hours of the deadline
  - 50 % deduction for submissions that are more than 24 hours late but are within 48 hours of the deadline.
  - 100% deduction if more than 48 hour late.

Please remember that PLAGIARISM is INTOLERABLE and anyone found involved in it will get -3 marks (i.e. 100% penalty) in this assignment.

There have been several cases in the past where grades of some students were severely affected because of such penalty

Department of Computer Science **FAST-NUCES** **Lahore Campus**

CS401 Artificial Intelligence

Programming Assignment No 1 (Sections C, D, E) Spring 2019
Assigned on 11/2/2019 Deadline 25/2/2019 before 5:00 p.m.
Submission: Online on SLATE Weight 3%

Game Playing

Game of chess has been a classic AI problem and these days there are several excellent open source implementations of automatic chess players, like GNU chess, are available on line. In this assignment we are going to make yet another FAST-chess player that will play chess using some well-established strategies along with **MINIMAX** to compute moves for a chess player.

To complete this assignment you have been provided with an abstract base class called **chessPlayer** and a demo player class named **groupName**. Along with these two class, another class, called **gameState,** is also provided that holds state of the game at any given time. The state of the chess at any given time consists of information about player turn (i.e. an enumeration called **Color** with a value of 1 means **WHITE** player will move and 0 means **BLACK** player will move) and an 8 x 8 chess board with chess pieces encoded as shown below.

4	2	3	5	6	3	2	4
1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
-1	-1	-1	-1	-1	-1	-1	-1
-4	-2	-3	-5	-6	-3	-2	-4

The **negative numbers** represent **pieces** of **Black player** and **positive numbers** represent **pieces** of **White player**

For your convenience, all major functions for playing chess are already implemented and the chess state also includes a list of valid moves available at any given time.

Department of Computer Science **FAST-NUCES** **Lahore Campus**

CS401 Artificial Intelligence

Programming Assignment No 1 (Sections C, D, E) Spring 2019
Assigned on 11/2/2019 Deadline 25/2/2019 before 5:00 p.m.
Submission: Online on SLATE Weight 3%

Your primary job in this assignment is to create an automatic chess player by overwriting a virtual function **decideMove** in the abstract base class **chessPlayer.**

To give you an idea, a sample **autoPlayer** is also provided to along with a deriver program. My auto-player selects a move at random from a set of available moves.

MAJOR TASK: DECIDE A SINGLE MOVE USING MINIMAX (DFS) ALGORITHM

Your implementation must use alpha-beta pruning and a **max-depth** parameter to specify the maximum depth of the tree to be used for making a single move. Indeed this would require an implementation of the evaluation function as well. [3 Marks].

Grading Guidelines

Marks will be awarded for

- a detailed review of available Evaluation Functions (You must submit a report explaining various evaluation functions already available) **[10 Points]**
- a detailed description of your own evaluation function **[10 Points]**
- correct implementation of your auto player **[20 Points]**
- ranking of your player w.r.t the remaining students **[10 Points]**

Please remember that you are not allowed to modify the existing classes and must provide only your own auto-player class that must be a child of the abstract **chessPlayer** class already provided to you.