**DOMINEERING – 2. faza projekta**

Nakon faktorisanja koda, u fajlu **UserInterface.py** posedujemo argumente koji služe da kada u terminalu pokrenemo komandu python main.py sa dva brojčana argumenta, crta se tabla tih dimenzija, a ukoliko se doda i treći argument bira se da li igra prvi AI ili čovek. Maksimalna veličina table postavljena je na 20x20.

length = len(sys.argv)

if length >= 3:

    if int(sys.argv[1]) > 20 or int(sys.argv[2]) > 20:

        print("Maksimalne dimenzije su 20x20!")

        print("Tabla postavljena na podrazumevanu vrednost 8x8")

    else:

        ROWS = int(sys.argv[1])

        COLS = int(sys.argv[2])

elif length >= 4:

        AI\_TURN = int(sys.argv[3])

def show\_bg(self, surface, field):

Ova funkcija iscrtava pozadinu, tj šahovsku tablu na kojoj se postavljaju domine, kao i same domine.

def get\_alphacol(col):

Funkcija get\_alphacol iscrtava numeraciju polja na šahovskoj tabli. Ukoliko se postavi polje veće od 20x20, tablja se postavlja na podrazumevanu vrednost 8x8.

def HoverPlayerOne(self, rows:int, cols:int, field, surface):

def HoverPlayerTwo(self, rows:int, cols:int, field, surface):

Ove funkcije omogućavaju igraču da vidi kako bi izgledao njegov potez kada se odigra.

def InvalidMoveAlert():

Iscrtava prozor sa porukom da je potez nevalidan.

def PlayerWonAlert(playerNum):

Iscrtava prozor sa porukom koji igrač je pobedio.

U fajlu **GameEngine.py** imamo funkcije iza koje stoji logika igre i poteza

def RaiseCounter():

Funkcija koja računa broj poteza, služi za ispisivanje brojeva na dominama i računanje mogućih poteza igrača.

def CreateMatrix(rows: int, cols: int):

Kreiranje matrice poteza

def getNextPlayer(playerOnMove):

Služi da vidimo koji igrač je na potezu

def isMoveValid(row, col, Mat, playerOnMove):

Provera da li su potezi oba igrača validni

def getAvailableMovesNumber(mat, playerOnMove):

Računanje mogućih preostalih poteza za oba igrača i da li je trenutni igrač pobedio

def getAvailableMovesMatrices(mat, playerOnMove):

Vraća matricu svih mogućih poteza za igrača koji je na potezu

def getNewMoveMatrix(row, col, mat, playerOnMove):

Funkcija za vraćanje sledećeg poteza igrača

def placeDomino(row, col, mat, playerOnMove):

Stavljanje domina aktivnog igrača ukoliko je potez validan

def PrintField(Field):

Štampanje celog polja u konzoli radi lakšeg pregleda (nije neophodno za GUI)

U fajlu **AI.py** imamo funkciju koja će služiti za pokretanje igre kada igra kompjuter u 3. fazi projekta

def getNextMove(matrix, isMoveValid):

U fajlu **Main.py** imamo funkcije koje služi za pokretanje cele igre

def resetGame(Field):

Funkcija koja služi za ponovno pokretanje igre klikom na taster R

def mainLoop(self):

Glavna petlja koja crta tablu i igraču, i pamti prethodna stanja kao i koji igrač igra. Na klik miša omogućava postavljanje pločice ili resetovanje igre na pritisak tastera R

**Tim PathFinders**

Nenad Đorđević, 16080

Ivan Bogosavljević, 17561

Nikola Rašić, 17907