Лабораториска вежба 2 – Пнви

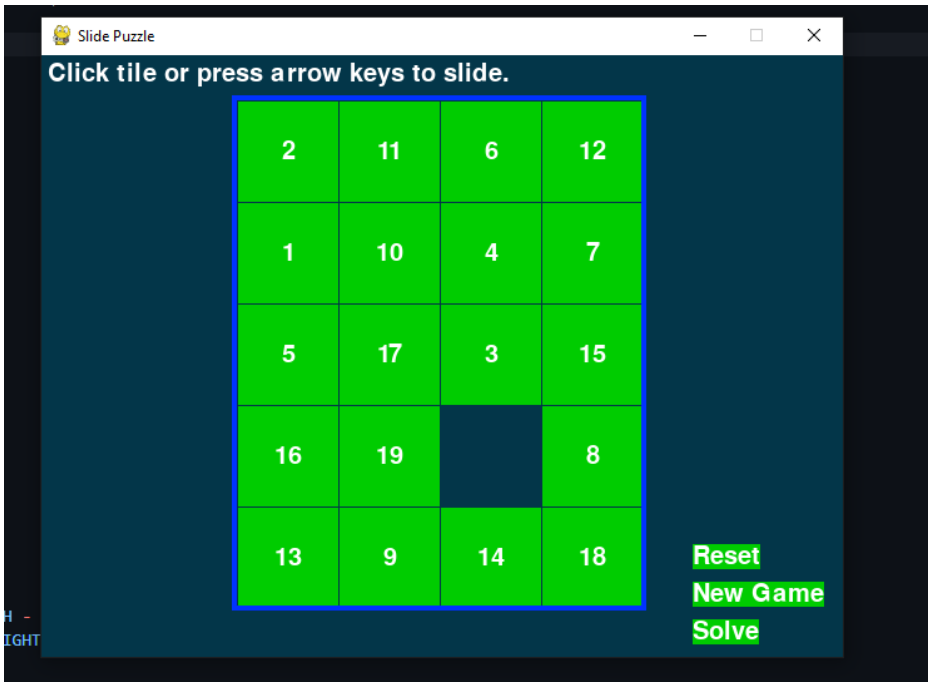
Фросина Цветковска 191216

1. Бројот на полиња во редиците и колоните да биде различен (на пример 8x6).

Со цел да се промени бројот на полињата во редиците и колоните ја менуваме вредноста на BOARDWIDTH и BOARDHEIGHT (линија 11 и 12). Доколку се зголеми бројот потребно би било и да се зголеми и вредноста на WINDOWHEIGHT и WINDOWWIDTH, но во овој случај со 4x5 тоа не е потребно

```
8
9   # Create the constants (go ahead and experiment with different values)
10  #барање 1 промена на бројот на полиња
11  BOARDWIDTH = 4  # number of columns in the board
12  BOARDHEIGHT = 5 # number of rows in the board
13  TILESIZE = 80
14  WINDOWWIDTH = 640
15  WINDOWHEIGHT = 480
```

Изгледот на board по самите промени:



2. Додадете ново копче "HELP". Ако играчот кликне на копчето, на екранот треба да му се прикаже порака со можна насока за придвижување во следниот чекор. Покрај тоа, сите соседни полиња на празното треба да се обоени во црвена боја.

Најпрво додаваме две нови глобални променливи HELP_SURF и HELP_RECT на линија 50 кои ќе ги користиме за да го креираме самото копче и да ги зачуваме промените во нив.

Исто така ќе ја додадеме и црвената боја која што ги прикажува можните обиди кои ги имаме(линија 25)

```
RIGHT = 'right'

def main():
    global FPSCLOCK, DISPLAYSURF, BASICFONT, RESET_SURF, RESET_RECT, NEW_SURF, NEW_RECT, SOLVE_SURF, SOLVE_RECT, HELP_SURF, HELP_RECT #барање 2

    pygame.init()
```

```
#                   R    G    B
BLACK =         (  0,   0,   0)
WHITE =         (255, 255, 255)
BRIGHTBLUE =    (  0,  50, 255)
DARKTURQUOISE = (  3,  54,  73)
GREEN =         (  0, 204,   0)
RED =           (255,   0,   0) #барање 2

BGCOLOR = DARKTURQUOISE
```

Потоа на линија 59 ја додаваме местоположбата на копчето заедно со одредени параметри како боја и големината на самиот текст.

```

    # Store the option buttons and their rectangles in OPTIONS.
    HELP_SURF, HELP_RECT = makeText('Help', TEXTCOLOR, TILECOLOR, WINDOWWIDTH - 120, WINDOWHEIGHT - 120) #барање 2
    RESET_SURF, RESET_RECT = makeText('Reset',    TEXTCOLOR, TILECOLOR, WINDOWWIDTH - 120, WINDOWHEIGHT - 90)
    NEW_SURF,   NEW_RECT   = makeText('New Game', TEXTCOLOR, TILECOLOR, WINDOWWIDTH - 120, WINDOWHEIGHT - 60)
    SOLVE_SURF, SOLVE_RECT = makeText('Solve',    TEXTCOLOR, TILECOLOR, WINDOWWIDTH - 120, WINDOWHEIGHT - 30)

    mainBoard, solutionSeq = generateNewPuzzle(80)
```

Во функцијата DrawBoard() на линија 258 додаваме DISPLAYSURF.blit(HELP_SURF, HELP_RECT) за да го исцртаме копчето на екран.

```
def drawBoard(board, message):
    DISPLAYSURF.fill(BGCOLOR)
    if message:
        textSurf, textRect = makeText(message, MESSAGECOLOR, BGCOLOR, 5, 5)
        DISPLAYSURF.blit(textSurf, textRect)

    for tilex in range(len(board)):
        for tiley in range(len(board[0])):
            if board[tilex][tiley]:
                drawTile(tilex, tiley, board[tilex][tiley])

    left, top = getLeftTopOfTile(0, 0)
    width = BOARDWIDTH * TILESIZE
    height = BOARDHEIGHT * TILESIZE
    pygame.draw.rect(DISPLAYSURF, BORDERCOLOR, (left - 5, top - 5, width + 11, height + 11), 4)

    DISPLAYSURF.blit(HELP_SURF, HELP_RECT) #барање 2
    DISPLAYSURF.blit(RESET_SURF, RESET_RECT)
    DISPLAYSURF.blit(NEW_SURF, NEW_RECT)
    DISPLAYSURF.blit(SOLVE_SURF, SOLVE_RECT)

```

Со цел да ги креираме самите функционалности на копчето почнуваме со додавање на elif дел во main делот (линии 93-97)

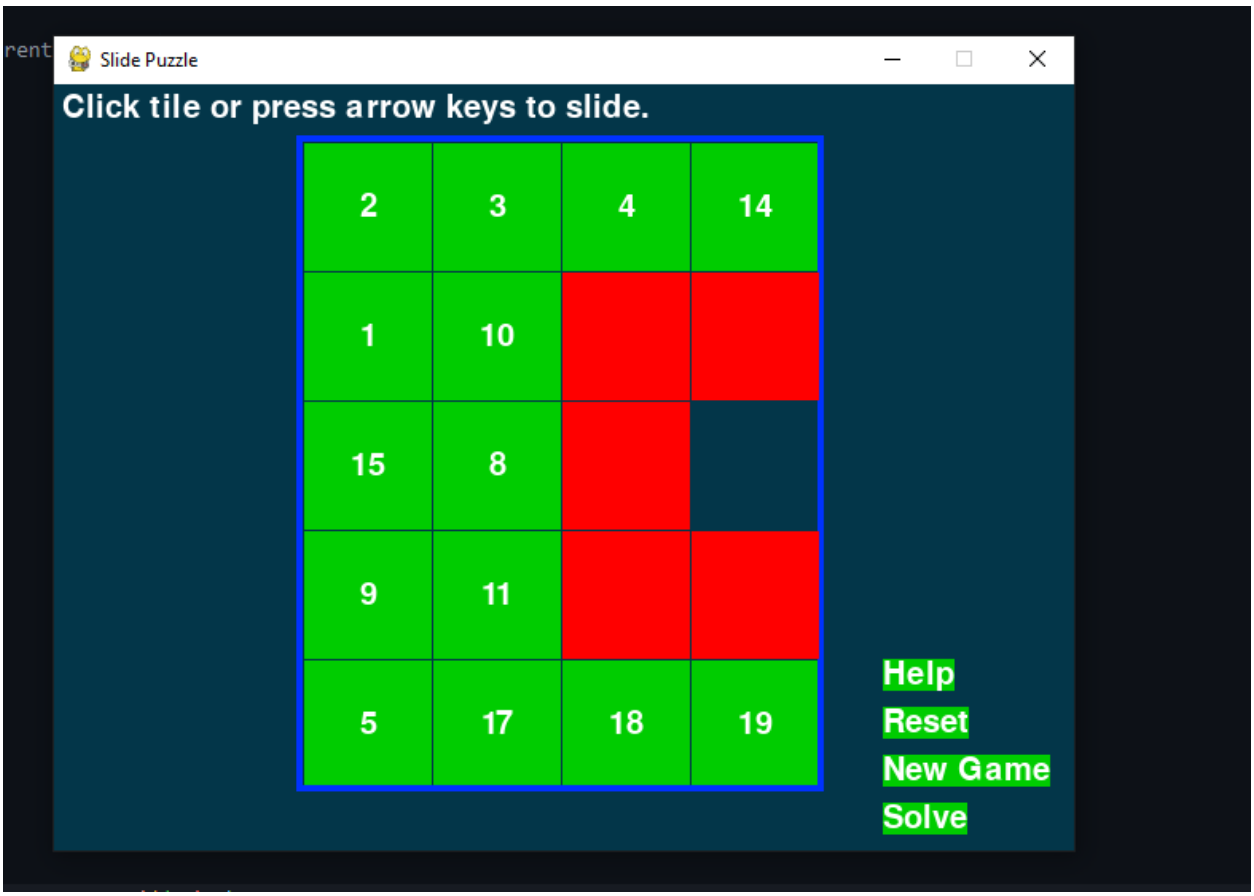
```
75
76          checkForQuit()
77          for event in pygame.event.get(): # event handling loop
78              if event.type == MOUSEBUTTONUP:
79                  spotx, spoty = getSpotClicked(mainBoard, event.pos[0], event.pos[1])
80
81                  if (spotx, spoty) == (None, None):
82                      # check if the user clicked on an option button
83                      if RESET_RECT.collidepoint(event.pos):
84                          resetAnimation(mainBoard, allMoves) # clicked on Reset button
85                          allMoves = []
86                      elif NEW_RECT.collidepoint(event.pos):
87                          mainBoard, solutionSeq = generateNewPuzzle(80) # clicked on New Game button
88                          allMoves = []
89                      elif SOLVE_RECT.collidepoint(event.pos):
90                          resetAnimation(mainBoard, solutionSeq + allMoves) # clicked on Solve button
91                          allMoves = []
92                      #барање 2
93                      elif HELP_RECT.collidepoint(event.pos):
94                          print(solutionSeq)
95                          print(allMoves)
96                          helpAnimation(mainBoard, msg)
97                          allMoves = []
98                  else:
99                      # check if the clicked tile was next to the blank spot
```

На крај креираме нова функција helpAnimation која што ја зема празната позиција, проверува дали истата е празна и доколку е проверува кои се можните влечења кои може играчот да ги постигне. Потоа се изцртуваат во црвена боја а во самиот терминал ја добиваме листата од moves (линии 345-366)

```
, 'left', 'up', 'up', 'right', 'up', 'left', 'down', 'down', 'left', 'down', 'right', 'down', 'right', 'right', 'up', 'left', 'up', 'left', 'up', 'up', 'right', 'down', 'left', 'left', 'down', 'right', 'up', 'left', 'up', 'right', 'right', 'rig
ht', 'down', 'left', 'down', 'left', 'down', 'right', 'right', 'up', 'left', 'down', 'right', 'down', 'left', 'left', 'left', 'up', 'up']
[]
['down', 'right', 'right', 'right', 'down', 'down', 'down', 'left', 'left', 'up', 'left', 'up', 'up', 'right', 'right', 'right', 'up', 'left', 'left', 'down', 'left', 'down', 'right', 'up', 'up', 'right', 'right', 'down', 'down', 'left', 'down'
, 'left', 'up', 'up', 'right', 'up', 'left', 'down', 'down', 'left', 'down', 'right', 'down', 'right', 'right', 'up', 'left', 'up', 'left', 'up', 'up', 'right', 'down', 'left', 'left', 'down', 'right', 'up', 'left', 'up', 'right', 'right', 'rig
ht', 'down', 'left', 'down', 'left', 'down', 'right', 'right', 'up', 'left', 'down', 'right', 'down', 'left', 'left', 'left', 'up', 'up']
[]
```

```python
#барање 2, креирање на help функција
def helpAnimation(board, message):
    posx, posy = getBlankPosition(board)

    if(posx, posy) == (None, None):
        return

    highlight = pygame.Surface((TILESIZE, TILESIZE), pygame.SRCALPHA)
    pygame.draw.rect(highlight, RED, (0,0,TILESIZE, TILESIZE), 0)

    for i in range (-1, 2):
        for j in range (-1, 2):
            if 0 <= posx + i < BOARDWIDTH and 0 <= posy + j < BOARDHEIGHT and (i!=0 or j!=0):
                left, top = getLeftTopOfTile(posx + i, posy + j)
                DISPLAYSURF.blit(highlight, (left, top))

    pygame.display.update()
    pygame.time.wait(100)

    textSurf, textRect = makeText(message, MESSAGECOLOR, BGCOLOR, 4, 5)
    DISPLAYSURF.blit(textSurf, textRect)

    drawBoard(board, message)

if __name__ == '__main__':
    main()
```

Крајниот резултат:



3. Сменете ја програмата со воведување на променливи blankxpos и blankypos за чување на позицијата на празната коцка. Функцијата којашто ја проверува позицијата на празната коцка треба да биде сменета соодветно, а и сите други функции коишто би имале влијание на ова.

Ги додаваме променливите како глобални со цел да извршиме промена на останатите места и да ја чуваме промената на едно место.

```
43    RIGHT = 'right'
44
45    #барање 3
46    blankxpos = BOARDWIDTH - 1
47    blankypos = BOARDHEIGHT - 1
48
49    def main():
```

Првата промена ја извршуваме на getBlankPosition

```
for x in range(BOARDWIDTH):
    for y in range(BOARDHEIGHT):
        if board[x][y] == BLANK:
            return (x, y)
```

Извршена промена:

```
160
161   #барање 3 промена на blankxpos и blankypos
162   def getBlankPosition(board):
163       # Return the x and y of board coordinates of the blank space.
164       return (blankxpos, blankypos)
165
```

Потоа ги менуваме blankx и blanky во makeMove() функцијата

```
166
167 def makeMove(board, move):
168     # This function does not check if the move is valid.
169     blankx, blanky = getBlankPosition(board)
170
171     if move == UP:
172         board[blankx][blanky], board[blankx][blanky + 1] = board[blankx][blanky + 1], board[blankx][blanky]
173     elif move == DOWN:
174         board[blankx][blanky], board[blankx][blanky - 1] = board[blankx][blanky - 1], board[blankx][blanky]
175     elif move == LEFT:
176         board[blankx][blanky], board[blankx + 1][blanky] = board[blankx + 1][blanky], board[blankx][blanky]
177     elif move == RIGHT:
178         board[blankx][blanky], board[blankx - 1][blanky] = board[blankx - 1][blanky], board[blankx][blanky]
179
```

Извршена промена:

```
#барање 3 промена на blankxpos и blankypos
def makeMove(board, move):
    # This function does not check if the move is valid.
    global blankxpos, blankypos

    if move == UP:
        board[blankxpos][blankypos], board[blankxpos][blankypos + 1] = board[blankxpos][blankypos + 1], board[blankxpos][blankypos]
    elif move == DOWN:
        board[blankxpos][blankypos], board[blankxpos][blankypos - 1] = board[blankxpos][blankypos - 1], board[blankxpos][blankypos]
    elif move == LEFT:
        board[blankxpos][blankypos], board[blankxpos + 1][blankypos] = board[blankxpos + 1][blankypos], board[blankxpos][blankypos]
    elif move == RIGHT:
        board[blankxpos][blankypos], board[blankxpos - 1][blankypos] = board[blankxpos - 1][blankypos], board[blankxpos][blankypos]
```

Следна промена ќе биде во функцијата isValidMove()

```
def isValidMove(board, move):
    blankx, blanky = getBlankPosition(board)
    return (move == UP and blanky != len(board[0]) - 1) or \
           (move == DOWN and blanky != 0) or \
           (move == LEFT and blankx != len(board) - 1) or \
           (move == RIGHT and blankx != 0)
```

Извршена промена:

```
#барање 3 промена на blankxpos и blankypos
def isValidMove(move):
    return (move == UP and blankypos != BOARDHEIGHT - 1) or \
           (move == DOWN and blankypos != 0) or \
           (move == LEFT and blankxpos != BOARDWIDTH - 1) or \
           (move == RIGHT and blankxpos != 0)
```

Крајната промена ќе се изврши во slideAnimation функцијата.

```
262
263  #барање 3 промена на blankxpos и blankypos
264  def slideAnimation(board, direction, message, animationSpeed):
265      # Note: This function does not check if the move is valid.
266      global blankxpos, blankypos
267
268      if direction == UP:
269          movex = blankxpos
270          movey = blankypos + 1
271          blankypos +=1
272      elif direction == DOWN:
273          movex = blankxpos
274          movey = blankypos - 1
275          blankypos -=1
276      elif direction == LEFT:
277          movex = blankxpos + 1
278          movey = blankypos
279          blankxpos +=1
280      elif direction == RIGHT:
281          movex = blankxpos - 1
282          movey = blankypos
283          blankxpos -=1
284
285      # prepare the base surface
286      drawBoard(board, message)
287      baseSurf = DISPLAYSURF.copy()
288      # draw a blank space over the moving tile on the baseSurf Surface.
289      moveLeft, moveTop = getLeftTopOfTile(movex, movey)
```

Извршена промена:

```
263
264  def slideAnimation(board, direction, message, animationSpeed):
265      # Note: This function does not check if the move is valid.
266
267      blankx, blanky = getBlankPosition(board)
268      if direction == UP:
269          movex = blankx
270          movey = blanky + 1
271      elif direction == DOWN:
272          movex = blankx
273          movey = blanky - 1
274      elif direction == LEFT:
275          movex = blankx + 1
276          movey = blanky
277      elif direction == RIGHT:
278          movex = blankx - 1
279          movey = blanky
280
281      # prepare the base surface
282      drawBoard(board, message)
283      baseSurf = DISPLAYSURF.copy()
284      # draw a blank space over the moving tile on the baseSurf Surface.
285      moveLeft, moveTop = getLeftTopOfTile(movex, movey)
286      pygame.draw.rect(baseSurf, BGCOLOR, (moveLeft, moveTop, TILESIZE, TILESIZE))
287
288      for i in range(0, TILESIZE, animationSpeed):
289          # animate the tile sliding over
290          checkForQuit()
291          DISPLAYSURF.blit(baseSurf, (0, 0))
292          if direction == UP:
293              drawTile(movex, movey, board[movex][movey], 0, -i)
294          if direction == DOWN:
295              drawTile(movex, movey, board[movex][movey], 0, i)
296          if direction == LEFT:
297              drawTile(movex, movey, board[movex][movey], -i, 0)
298          if direction == RIGHT:
299              drawTile(movex, movey, board[movex][movey], i, 0)
```

Прикачете го конечниот код со означени и коментирани промени што сте ги направиле. Дополнително, во еден документ ставете ги направените промени, прикачете слики што

ќе ги илустрираат тие промени заедно со една до неколку реченици објаснување на што се однесува сликата и промената.