ВВЕДЕНИЕ

Цель прохождения практики: изучить библиотеку “Pygame” для использования в будущих проектах, а также проявить способности в разработке игры, использующей графический интерфейс и обработку действий с компьютерных устройств ввода – мышь. А ещё - укрепление знаний в самом языке “Python”, его синтаксисе, конструкциях.

Для прохождения практики я выбрал игру “Крестики – нолики”. Это игра для двух игроков, в которой игроки ставят свои фигуры на поле 3х3 и тот, кто поставил 3 своих фигуры в ряд или диагональ первым – выиграл.

Задачи, решение которых направлено на достижение поставленной цели:

1. Обозначить правила игры.
2. Реализовать пользовательский интерфейс, с которым будут взаимодействовать игроки.
3. Написать грамотный код.
4. ОСНОВНАЯ ЧАСТЬ
   1. Словесное описание.

Игровое поле – квадрат, разделенный на 9 равных квадратов (по 3 в строчку и 3 в столбик). В один из незанятых квадратов, игрок, который на данный момент ходит должен поставить свою фигуру (1 игрок – крестик, 2 игрок – нолик). Первый игрок, поставивших 3 свои фигуры в строчку, столбик или диагональ – выигрывает игру.

Логика игры:

1. Игра начинается с пустого поля
2. Первый ход – крестик, игрок имеет право выбрать любой квадрат
3. Последующие ходы (четный ход – нолик, нечетный ход – крестик) имеет право выбрать любой, незанятый другой фигурой квадрат

Как игра может закончиться:

1. На поле есть 3 одинаковые фигуры в строку, столбик или диагональ, в этом случае выигрывает тот, чьи фигуры так поставлены.
2. На поле закончились свободные места и нет 3 фигур подряд, в этом случае – ничья.
   1. Математическое описание.

Игровое поле – список, состоящий из 9 “ячеек”.

Начало игры:

1. Создается окно игры.
2. Создается и обнуляется список значений ячеек, а также другие переменные.
3. Программа чертит игровое поле.

Игрок кликнул на игровое поле:

1. Проверяется положение курсора. Если он не на клетке – ничего не происходит, иначе – пункт 2.
2. Программа проверяет элемент списка соответствующей клетки. Если она занята – ничего не происходит, иначе – пункт 3.
3. Программа присваивает значение игрока в ячейку списка соответствующей клетки (1 для “крестика”, -1 для “нолика”).
4. На нужной клетке на экране появляется фигура игрока.

Затем программа проверяет на наличие ситуации, когда игра должна, закончиться:

1. Если сумма строки, столбца или диагонали списка значений равна 3 или -3 – нужно вывести результаты игры (если 3 – победа первого игрока, или же “крестиков”, а если -3 – победа второго игрока, или же “ноликов”)
2. Значение счетчика ходов достигло 9, это означает, что все клетки уже заняты. В этом случае программа выводит ничью

Бывают случаи, когда выигрывают последним ходом, именно поэтому программа сначала проверяет первое условие, тогда ошибок не возникает.

1.3 Программная реализация:

Создание окна и переменных

#window resolution

screen\_width=800 #must be EXACTLY 200 more than height

screen\_height=600 #must divide by 3 and be at least 300

#window setup

screen = pygame.display.set\_mode((screen\_width,screen\_height))

pygame.display.set\_caption('TicTacToe')

#variables

line\_width = 6

markers = []

clicked = False

pos = []

player=1

winner=0

game\_over=False

tie=False

turns=0

run=True

sc\_w=screen\_width

sc\_h=screen\_height

sc\_s=screen\_height

#colors

green=(0,255,0)

red=(255,0,0)

blue=(0,0,255)

bg = (255,255,240)

grid = (50,50,50)

#font

font=pygame.font.SysFont(None,40)

Создание списка значений:

for x in range(3):

row=[0]\*3

markers.append(row)

Функция отрисовки поля:

def draw\_grid():

screen.fill(bg)

for x in range(0,4):

pygame.draw.line(screen, grid, (0, x\*sc\_h//3), (sc\_w-200, x\*sc\_h//3), line\_width)

pygame.draw.line(screen, grid, (x\*(sc\_w-200)//3, 0), (x\*(sc\_w-200)//3, sc\_h), line\_width)

Функция, рисующая фигуры:

def draw\_markers():

x\_pos=0

for x in markers:

y\_pos=0

for y in x:

if y==1:

pygame.draw.line(screen,blue,(sc\_s//3\*x\_pos+sc\_s//20, sc\_s//3\*y\_pos+sc\_s//20),(sc\_s//3\*(x\_pos+1)-sc\_s//20,sc\_s//3\*(y\_pos+1)-sc\_s//20),line\_width)

pygame.draw.line(screen,blue,(sc\_s//3\*x\_pos+sc\_s//20, sc\_s//3\*(y\_pos+1)-sc\_s//20),(sc\_s//3\*(x\_pos+1)-sc\_s//20,sc\_s//3\*y\_pos+sc\_s//20),line\_width)

if y==-1:

pygame.draw.circle(screen,red,(sc\_s//3\*(x\_pos+1/2),sc\_s//3\*(y\_pos+1/2)),sc\_s//8,line\_width)

y\_pos+=1

x\_pos+=1

Функция проверки состояния игры:

def check\_winner():

global winner

global game\_over

global tie

y\_pos=0

for x in range(3):

#check rows

if sum(markers[x])==3:

winner=1

game\_over=True

if sum(markers[x])==-3:

winner=2

game\_over=True

#check columns

if markers[0][x] + markers[1][x] + markers[2][x] == 3:

winner=1

game\_over=True

if markers[0][x] + markers[1][x] + markers[2][x] == -3:

winner=2

game\_over=True

#check diagonals

if markers[0][0] + markers[1][1] + markers[2][2] == 3:

winner=1

game\_over=True

if markers[0][2] + markers[1][1] + markers[2][0] == 3:

winner=1

game\_over=True

if markers[0][0] + markers[1][1] + markers[2][2] == -3:

winner=2

game\_over=True

if markers[0][2] + markers[1][1] + markers[2][0] == -3:

winner=2

game\_over=True

#check tie situation

elif turns==9 and winner==0:

tie=True

game\_over=True

Функция вывода результата игры:

#drawing results of the game function

def draw\_winner(winner,tie):

if tie==False:

win\_text= 'Player ' + str(winner) + ' wins!'

win\_img= font.render(win\_text,True,blue)

pygame.draw.rect(screen,green,(sc\_w-196,0,196,50))

screen.blit(win\_img, (sc\_w-196,12))

elif tie==True:

tie\_text='Tie'

tie\_img=font.render(tie\_text,True,blue)

pygame.draw.rect(screen,green,(sc\_w-196,0,196,50))

screen.blit(tie\_img, (sc\_w-126,12))

again\_text='Play again?'

again\_img=font.render(again\_text,True,blue)

pygame.draw.rect(screen,green,again\_rect)

screen.blit(again\_img,(sc\_w-176,72))

Основной цикл программы:

while run:

draw\_grid()

draw\_markers()

#checking the events

for event in pygame.event.get():

#quitting game

if event.type==pygame.QUIT:

run=False

#if game isn't over

if game\_over==False:

Проверка нажатия на мышь и её позицию:

#checking IF the mouse is clicked

if event.type==pygame.MOUSEBUTTONDOWN and clicked==False:

clicked=True

#checking WHERE the mouse is clicked

if event.type==pygame.MOUSEBUTTONUP and clicked==True:

clicked=False

pos=pygame.mouse.get\_pos()

cell\_x=pos[0]//((sc\_w-200)//3)

cell\_y=pos[1]//(sc\_h//3)

Запись действий игроков в список и переход на следующий ход:

if pos[0]<sc\_w-200 and pos[1]<sc\_h and markers[cell\_x][cell\_y]==0:

markers[cell\_x][cell\_y]=player

player\*=-1

turns+=1

check\_winner()

Возвращение игры в начальное состояние при нажатии “Играть снова”:

if game\_over==True:

draw\_winner(winner,tie)

#if so returning game state to beginning

if event.type==pygame.MOUSEBUTTONDOWN and clicked==False:

clicked=True

if event.type==pygame.MOUSEBUTTONUP and clicked==True:

clicked=False

pos=pygame.mouse.get\_pos()

if again\_rect.collidepoint(pos):

markers = []

pos = []

player=1

winner=0

game\_over=False

tie=False

turns=0

for x in range(3):

row=[0]\*3

markers.append(row)

ЗАКЛЮЧЕНИЕ

Поставленные цели были достигнуты. Написана игра с графикой, получены основные знания по библиотеке “Pygame”. Применены навыки работы с графическим интерфейсом и устройствами ввода. Также получены навыки работы с координатами на экране компьютера и в окне игры, и в написании функций.

В итоге разработана готовая игра, которая работает правильно, быстро, имеет простой интерфейс и правила. В неё можно поиграть вдвоём в любое время, в любом месте.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Документация языка “Python” – URL: <https://docs.python.org/3/index.html>
2. Документация библиотеки “Pygame” – URL: <https://devdocs.io/pygame/>
3. Статья “Крестики-нолики” – Wikipedia – URL: <https://en.wikipedia.org/wiki/Tic-tac-toe>
4. Работа с двумерными списками Python – URL: https://timeweb.com/ru/community/articles/dvumernye-massivy-python-i-metody-raboty-s-nimi