Итоговый проект по дисциплине «Основы алгоритмизации и программирования» Игра “Тетрис”

Выполнил: Баранюк Никита ИС-24

Преподаватель: Манакова Ольга Петровна

Цели и задачи проекта

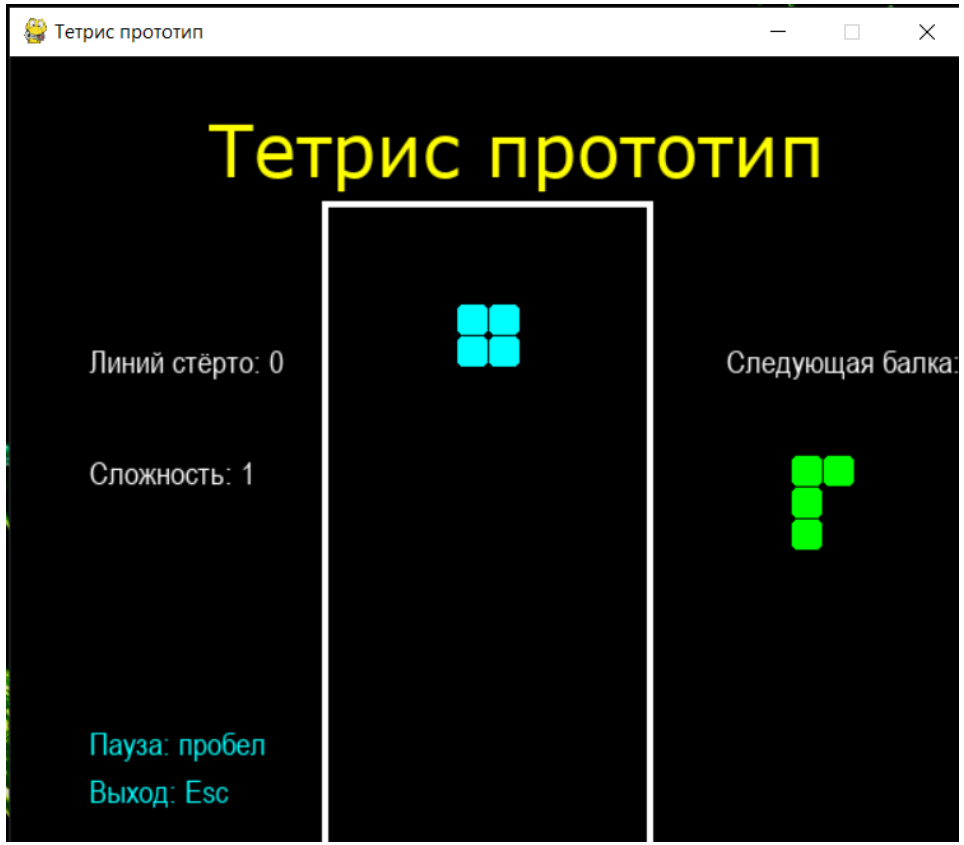
- Создать рабочую версию игры «Тетрис» на языке Python
- Написать программный код с разумной логикой, протестировать его и решить возможные ошибки
- Создать систему сложности , повышающеюся по мере прохождения игры

Библиотеки и модули

- pygame - основная библиотека для создания графической составляющей (необходимо установить при помощи командной строки)
- random - необходим для случайности вылета фигур
- sys - обеспечивает доступ к некоторым переменным и функциям, взаимодействующим с интерпретатором python
- time - дает возможность задать скорость падения фигуры вниз

Дизайн программы

- Дизайн приложения крайне прост и интуитивно понятен, ведь в своём проекте я ставил больший упор на геймплейную составляющую.



Игровой процесс

- По началу возможны небольшие проблемы с управлением(т.к. при некоторые фигуры не поворачиваются у граней «стакана» и возможны некоторые проблемы при перемещении фигур), однако со временем к этому можно привыкнуть.
- Также, когда игрок стирает 10 линий, сложность повышается на 1, при этом скорость падения увеличивается, благодаря чему игра становится все сложнее и сложнее по мере прохождения

Управление

- Стрелки вправо и влево – перемещение фигур
- Стрелка вверх – вращение фигуры
- Стрелка вниз – ускорение падения фигуры
- Space – пауза
- escape - выйти из игры

Написание кода

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
import pygame as pg
import random, time, sys
from pygame.locals import *

fps = 25
window_w, window_h = 600, 500
block, cup_h, cup_w = 20, 20, 10
```

Здесь мы импортируем необходимые библиотеки, а также задаем параметры для окна программы и настраиваем fps.

```
fig_w, fig_h = 5, 5
empty = 'o'

figures = {'S': [['ooooo',
                  'ooooo',
                  'ooxxo',
                  'oxxoo',
                  'ooooo'],
                 ['ooooo',
                  'ooxoo',
                  'ooxxo',
                  'oooxo',
                  'ooooo']],
```

Тут мы задаем шаблон для фигур и их перевернутых фигур. Так как у фигур размер 4x4, то мы делаем размер шаблона 5x5.
Так мы настраиваем положения и для остальных фигур (Z, S, O, J, L, T).

Написание кода

```
def main():
    global fps_clock, display_surf, basic_font, big_font
    pg.init()
    fps_clock = pg.time.Clock()
    display_surf = pg.display.set_mode((window_w, window_h))
    basic_font = pg.font.SysFont('arial', 20)
    big_font = pg.font.SysFont('verdana', 45)
    pg.display.set_caption('Тетрис прототип')
    showText('Тетрис прототип')
    while True:  # начинаем игру
        runTetris()
        pauseScreen()
        showText('Потрачено')
```

Функция main отвечает за создание нескольких дополнительных глобальных констант, инициализирует модуль Pygame, рисует стартовое окно игры, вызывает запуск Тетриса runTetris() и отображает сообщение о проигрыше.

```
def runTetris():
    cup = emptycup()
    last_move_down = time.time()
    last_side_move = time.time()
    last_fall = time.time()
    going_down = False
    going_left = False
    going_right = False
    points = 0
    level, fall_speed = calcSpeed(points)
    fallingFig = getNewFig()
    nextFig = getNewFig()
```

В этой функции реализуется сам код игры, а именно:
Создается стакан, где будут находится балки
Значения меняются на True, когда нажимается соответственные клавиши
Ведется подсчет очков и сложности
Отображается падающая фигура (балка)
Отображение следующей фигуры

Написание кода

```
for event in pg.event.get():
    if event.type == KEYUP:
        if event.key == K_SPACE:
            pauseScreen()
            showText('Пауза')
            last_fall = time.time()
            last_move_down = time.time()
            last_side_move = time.time()
        elif event.key == K_LEFT:
            going_left = False
        elif event.key == K_RIGHT:
            going_right = False
        elif event.key == K_DOWN:
            going_down = False

elif event.type == KEYDOWN:
    # перемещение фигуры вправо и влево
    if event.key == K_LEFT and checkPos(cup, fallingFig, adjX=-1):
        fallingFig['x'] -= 1
        going_left = True
        going_right = False
        last_side_move = time.time()

    elif event.key == K_RIGHT and checkPos(cup, fallingFig, adjX=1):
        fallingFig['x'] += 1
        going_right = True
        going_left = False
        last_side_move = time.time()
```

Здесь мы уже расписываем основное управление и назначения клавиш, такие как:
Остановка игры на паузу
Перемещение фигур влево и вправо на соответствующие стрелочки
Ускорения падения при задержании стрелки вниз
Вращение фигур при нажатии стрелки вниз

Написание кода

```
# поворачиваем фигуру, если есть место
elif event.key == K_UP:
    fallingFig['rotation'] = (fallingFig['rotation'] + 1) % len(figures[fallingFig['shape']])
    if not checkPos(cup, fallingFig):
        fallingFig['rotation'] = (fallingFig['rotation'] - 1) % len(figures[fallingFig['shape']])

# ускоряем падение фигуры
elif event.key == K_DOWN:
    going_down = True
    if checkPos(cup, fallingFig, adjY=1):
        fallingFig['y'] += 1
    last_move_down = time.time()

# мгновенный сброс вниз
elif event.key == K_RETURN:
    going_down = False
    going_left = False
    going_right = False
    for i in range(1, cup_h):
        if not checkPos(cup, fallingFig, adjY=i):
            break
    fallingFig['y'] += i - 1
```

Продолжение предыдущего слайда

Написание кода

```
# управление падением фигуры при удержании клавиш
if (going_left or going_right) and time.time() - last_side_move > side_freq:
    if going_left and checkPos(cup, fallingFig, adjX=-1):
        fallingFig['x'] -= 1
    elif going_right and checkPos(cup, fallingFig, adjX=1):
        fallingFig['x'] += 1
    last_side_move = time.time()

if going_down and time.time() - last_move_down > down_freq and checkPos(cup, fallingFig, adjY=1):
    fallingFig['y'] += 1
    last_move_down = time.time()

if time.time() - last_fall > fall_speed:  # свободное падение фигуры
    if not checkPos(cup, fallingFig, adjY=1):  # проверка "приземления" фигуры
        addToCup(cup, fallingFig)  # фигура приземлилась, добавляем ее в содержимое стакана
        points += clearCompleted(cup)
        level, fall_speed = calcSpeed(points)
        fallingFig = None
    else:  # фигура пока не приземлилась, продолжаем движение вниз
        fallingFig['y'] += 1
        last_fall = time.time()
```

В данной функции мы задаем управление падения фигуры и настраиваем ускорение падения фигуры, если вмешивается пользователь

Написание кода

```
    # рисуем окно игры со всеми надписями
    display_surf.fill(bg_color)
    drawTitle()
    gamecup(cup)
    drawInfo(points, level)
    drawnextFig(nextFig)
    if fallingFig != None:
        drawFig(fallingFig)
    pg.display.update()
    fps_clock.tick(fps)
```

```
def txtObjects(text, font, color):
    surf = font.render(text, True, color)
    return surf, surf.get_rect()


def stopGame():
    pg.quit()
    sys.exit()


def checkKeys():
    quitGame()

    for event in pg.event.get([KEYDOWN, KEYUP]):
        if event.type == KEYDOWN:
            continue
        return event.key
    return None
```

Эта часть кода завершает функцию runTetris(). Она отвечает за отрисовку окна, игрового поля, показ фигур и всех надписей

Вспомогательные функции:
Функция txtObjects() принимает текст, шрифт и цвет, и с помощью метода render() возвращает готовые объекты Surface (поверхность) и Rect (прямоугольник).
Эти объекты в дальнейшем обрабатываются методом blitв функции showText(), выводящей информационные надписи и название игры.
Выход из игры обеспечивает функция stopGame(), в которой используется sys.exit() из импортированного в начале кода модуля sys.

Написание кода

```
def drawInfo(points, level):
    pointsSurf = basic_font.render(f'Линий стёрто: {points}', True, txt_color)
    pointsRect = pointsSurf.get_rect()
    pointsRect.topleft = (window_w - 550, 180)
    display_surf.blit(pointsSurf, pointsRect)

    levelSurf = basic_font.render(f'Сложность: {level}', True, txt_color)
    levelRect = levelSurf.get_rect()
    levelRect.topleft = (window_w - 550, 250)
    display_surf.blit(levelSurf, levelRect)

    pausebSurf = basic_font.render('Пауза: пробел', True, info_color)
    pausebRect = pausebSurf.get_rect()
    pausebRect.topleft = (window_w - 550, 420)
    display_surf.blit(pausebSurf, pausebRect)

    escbSurf = basic_font.render('Выход: Esc', True, info_color)
    escbRect = escbSurf.get_rect()
    escbRect.topleft = (window_w - 550, 450)
    display_surf.blit(escbSurf, escbRect)
```

Эта функция отвечает за отрисовку всех надписей и их положения в игровом окне

Написание кода

```
def drawFig(fig, pixelx=None, pixely=None):
    figToDraw = figures[fig['shape']][fig['rotation']]
    if pixelx == None and pixely == None:
        pixelx, pixely = convertCoords(fig['x'], fig['y'])

    # отрисовка элементов фигур
    for x in range(fig_w):
        for y in range(fig_h):
            if figToDraw[y][x] != empty:
                drawBlock(None, None, fig['color'], pixelx + (x * block), pixely + (y * block))


def drawnextFig(fig):  # превью следующей фигуры
    nextSurf = basic_font.render('Следующая балка:', True, txt_color)
    nextRect = nextSurf.get_rect()
    nextRect.topleft = (window_w - 150, 180)
    display_surf.blit(nextSurf, nextRect)
    drawFig(fig, pixelx=window_w - 150, pixely=230)
```

Обрисовывает и красит фигуры, а также обрисовывает следующую фигуру

Значение проекта

- Тетрис не нуждается в представление, большинство если не все люди хоть раз играли в него. Поэтому я и взял его в качестве своей работы, я хотел отдать честь этой легендарной игре.
- Так же игра предназначена для того, сто-бы скоротать свое время и немного отвлечься, так даже я в процессе работы над ним иногда просто залипал и пытался пройти как можно дальше

Тестирование

- Я дал некоторым одногруппникам свой код, дабы они протестировали игру и в случае чего указали возможные баги и ошибки
- В процессе тестирования они не обнаружили каких-либо серьезных ошибок и им понравился мой проект (прямой комментарий: «игра имба вцелом»)

Список литературы

- https://pythonworld.ru/moduli/modul-time.html
- https://pythonru.com/uroki/biblioteka-pygame-chast-1-vvedenie
- https://proglib.io/p/python-oop
- А также лекции ООП на основах алгоритмизации и программирования