МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«АЛТАЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Колледж Алтайского государственного университета
Отделение экономики и информационных технологий

Разработка игры «Морской бой»

Выполнила студентка 2 курса группы К.105с9-3 Радчен
Ксения Дмитриевна

Научный руководитель: Притула Ж.В.

Объект - язык программирования Python.

Предмет - игра «Морской бой».

Цель - реализации игры Морской бой на языке Python.

Задачи:

1. Провести анализ предметной области;
2. Изучить доступные средства программирования;
3. Разработать структуру программы.
4. Написать код игры Морской бой

Выбор языка программирования для разработки программы

Python – это скриптовый язык программирования. Он универсален, поэтому подходит для решения разнообразных задач и для многих платформ.
Синтаксис Питона максимально облегчен, что позволяет выучить его за сравнительно короткое время. Ядро имеет очень удобную структуру, а широкий перечень встроенных библиотек позволяет применять внушительный набор полезных функций и возможностей. ЯП может использоваться для написания прикладных приложений, а также разработки WEB-сервисов.

Библиотека Tkinter для реализации интерфейса

Tkinter– это **кроссплатформенная библиотека для разработки графического интерфейса на языке Python** (начиная с Python 3.0 переименована в tkinter). Tkinter расшифровывается как Tk interface, и является интерфейсом к Tcl/Tk. Tkinter входит в стандартный дистрибутив Python.

Игра “Морской бой”

Это – увлекательная развивающая игра, в которой участвуют два игрока. Цель – уничтожить корабли соперника быстрее, чем он потопит ваши. Каждому игроку понадобится листок бумаги, желательно в клеточку, ручка, можно пользоваться карандашом. Игрок рисует на листке два квадрата, размер каждого из них 10×10, т.е по 10 клеток по вертикали и горизонтали. .

- Но в неё можно играть в любом возрасте, в этом и состоит её актуальность, здесь нет никаких ограничений, а также эта игра отлично развивает логику и стратегическое мышление.

Импорт необходимых библиотек

```
from tkinter import *
from tkinter import messagebox
import time
import random
```

Создание игрового окна, полей и кораблей

```
tk = Tk()
app_running = True

size_canvas_x = 500
size_canvas_y = 500
s_x = s_y = 8  # размер игрового поля - не меньше 8 и не больше 18
#s_y = 8
step_x = size_canvas_x // s_x  # шаг по горизонтали
step_y = size_canvas_y // s_y  # шаг по вертикали
size_canvas_x = step_x * s_x
size_canvas_y = step_y * s_y


menu_y = 40
ships = s_x // 2  # определяем максимальное кол-во кораблей
ship_len1 = s_x // 5  # длина первого типа корабля
ship_len2 = s_x // 3  # длина второго типа корабля
ship_len3 = s_x // 2  # длина третьего типа корабля
enemy_ships1 = [[0 for i in range(s_x + 1)] for i in range(s_y + 1)]
enemy_ships2 = [[0 for i in range(s_x + 1)] for i in range(s_y + 1)]
list_ids = []  # список объектов canvas
```

```
def on_closing():
    global app_running
    if messagebox.askokcancel("Выход из игры", "Хотите выйти из игры?"):
        app_running = False
        tk.destroy()


tk.protocol("WM_DELETE_WINDOW", on_closing)
tk.title("Игра Морской Бой")
tk.resizable(0, 0)
tk.wm_attributes("-topmost", 1)
canvas = Canvas(tk, width=size_canvas_x + menu_x + size_canvas_x, height=size_canvas_y + menu_y, bd=0,
                highlightthickness=0)
canvas.create_rectangle(0, 0, size_canvas_x, size_canvas_y, fill="#827a71")
canvas.create_rectangle(size_canvas_x + menu_x, 0, size_canvas_x + menu_x + size_canvas_x, size_canvas_y,
                        fill="#e8dac9")
canvas.pack()
tk.update()
```

```
def draw_table(offset_x=0):
    for i in range(0, s_x + 1):
        canvas.create_line(offset_x + step_x * i, 0, offset_x + step_x * i, size_canvas_y)
    for i in range(0, s_y + 1):
        canvas.create_line(offset_x, step_y * i, offset_x + size_canvas_x, step_y * i)


draw_table()
draw_table(size_canvas_x + menu_x)

t0 = Label(tk, text="Игрок №1", font=("Helvetica", 16))
t0.place(x=size_canvas_x // 2 - t0.winfo_reqwidth() // 2, y=size_canvas_y + 3)
t1 = Label(tk, text="Игрок №2"+add_to_label, font=("Helvetica", 16))
t1.place(x=size_canvas_x + menu_x + size_canvas_x // 2 - t1.winfo_reqwidth() // 2, y=size_canvas_y + 3)

t0.configure(bg="#b5444c")
t0.configure(bg="#f0f0f0")

t3 = Label(tk, text="@@@@@@@", font=("Helvetica", 16))
t3.place(x=size_canvas_x + menu_x//2 - t3.winfo_reqwidth() // 2, y= size_canvas_y)
```

Координаты на игровом поле

```
def add_to_all(event):
    global points1, points2, hod_igrovomu_polu_1
    _type = 0  # ЛКМ
    if event.num == 3:
        _type = 1  # ПКМ
    # print(_type)
    mouse_x = canvas.winfo_pointerx() - canvas.winfo_rootx()
    mouse_y = canvas.winfo_pointery() - canvas.winfo_rooty()
    # print(mouse_x, mouse_y)
    ip_x = mouse_x // step_x
    ip_y = mouse_y // step_y
    #  print(ip_x, ip_y, "_type:", _type)
```

```
canvas.bind_all("<Button-1>", add_to_all)  # ЛКМ
canvas.bind_all("<Button-3>", add_to_all)  # ПКМ
```

Генерация кораблей для начала игры

```python
def generate_ships_list():
    global ships_list
    ships_list = []
    # генерируем список случайных длин кораблей
    for i in range(0, ships):
        ships_list.append(random.choice([ship_len1, ship_len2, ship_len3]))
    # print(ships_list)


def generate_enemy_ships():
    global ships_list
    enemy_ships = []

    # подсчет суммарной длины кораблей
    sum_1_all_ships = sum(ships_list)
    sum_1_enemy = 0
```

```python
while sum_1_enemy != sum_1_all_ships:
    # обнуляем массив кораблей врага
    enemy_ships = [[0 for i in range(s_x + 1)] for i in
                   range(s_y + 1)]  # +1 для доп. линии справа и снизу, для успешных проверок генерации противника

    for i in range(0, ships):
        len = ships_list[i]
        horizont_vertikal = random.randrange(1, 3)  # 1- горизонтальное 2 - вертикальное

        primerno_x = random.randrange(0, s_x)
        if primerno_x + len > s_x:
            primerno_x = primerno_x - len

        primerno_y = random.randrange(0, s_y)
        if primerno_y + len > s_y:
            primerno_y = primerno_y - len

        # print(horizont_vertikal, primerno_x,primerno_y)
        if horizont_vertikal == 1:
            if primerno_x + len <= s_x:
                for j in range(0, len):
                    try:
                        check_near_ships = 0
                        check_near_ships = enemy_ships[primerno_y][primerno_x - 1] + \
                                           enemy_ships[primerno_y][primerno_x + j] + \
                                           enemy_ships[primerno_y][primerno_x + j + 1] + \
```

Создание кнопок

```
b0 = Button(tk, text="Показать корабли \n Игрока №1", command=button_show_enemy1)
b0.place(x=size_canvas_x + menu_x // 2 - b0.winfo_reqwidth() // 2, y=10)

b1 = Button(tk, text="Показать корабли \n Игрока №2", command=button_show_enemy2)
b1.place(x=size_canvas_x + menu_x // 2 - b1.winfo_reqwidth() // 2, y=60)

b2 = Button(tk, text="Начать заново!", command=button_begin_again)
b2.place(x=size_canvas_x + menu_x // 2 - b2.winfo_reqwidth() // 2, y=110)
```

Задаём цвета кораблям

```
def button_show_enemy1():
    for i in range(0, s_x):
        for j in range(0, s_y):
            if enemy_ships1[j][i] > 0:
                color = "red"
                if points1[j][i] != -1:
                    color = "#1d4d00"
                _id = canvas.create_rectangle(i * step_x, j * step_y, i * step_x + step_x, j * step_y + step_y,
                                              fill=color)
                list_ids.append(_id)


def button_show_enemy2():
    for i in range(0, s_x):
        for j in range(0, s_y):
            if enemy_ships2[j][i] > 0:
                color = "red"
                if points2[j][i] != -1:
                    color = "#1d4d00"
                _id = canvas.create_rectangle(size_canvas_x + menu_x + i * step_x, j * step_y,
                                              size_canvas_x + menu_x + i * step_x + step_x, j * step_y + step_y,
                                              fill=color)
                list_ids.append(_id)
```

Определяем победителя

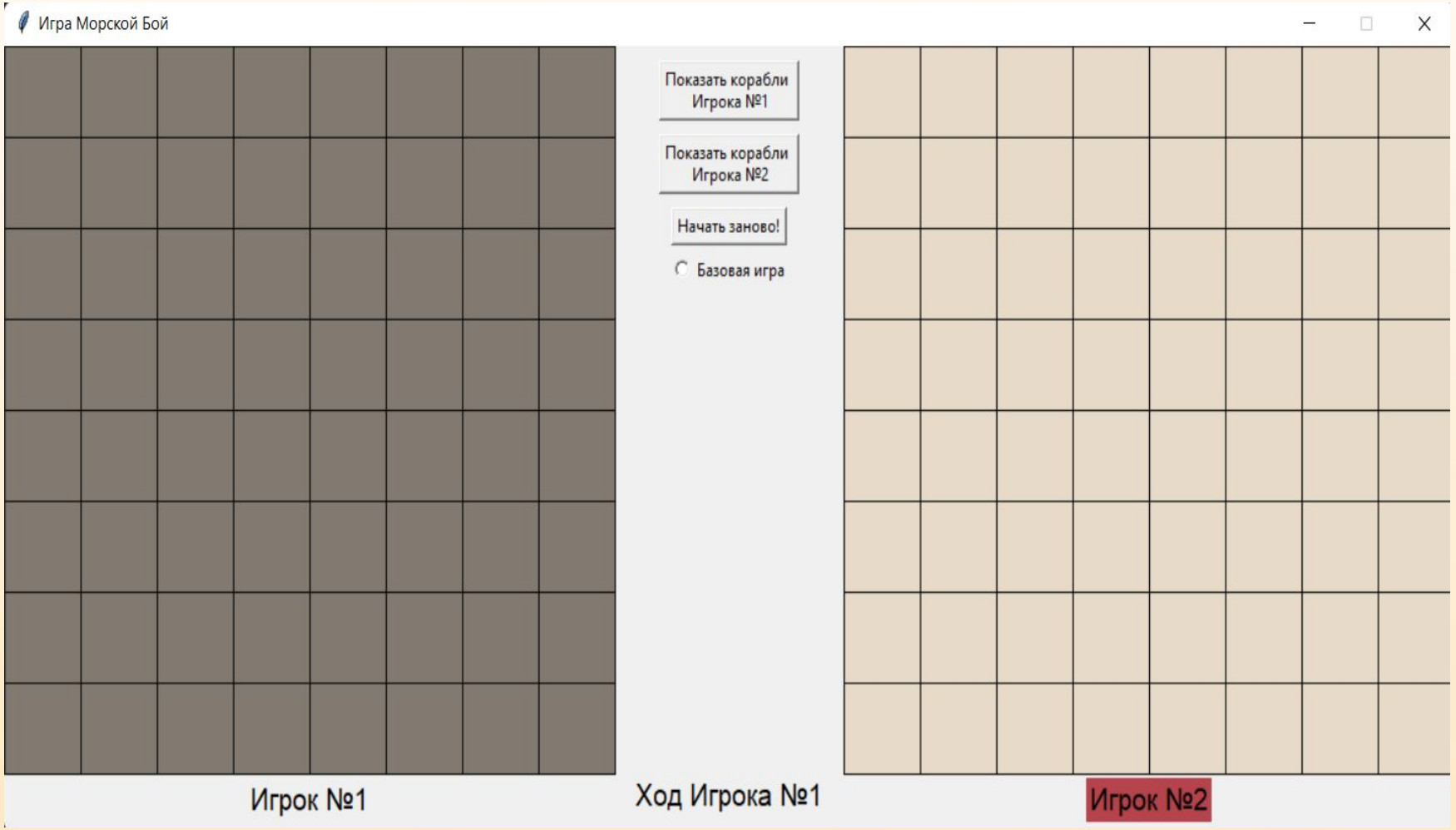
```
if ip_x < s_x and ip_y < s_y and hod_igrovomu_polu_1:
    if points1[ip_y][ip_x] == -1:
        points1[ip_y][ip_x] = _type
        hod_igrovomu_polu_1 = False
        draw_point(ip_x, ip_y)
        # if check_winner(ip_x, ip_y):
        if check_winner2():
            hod_igrovomu_polu_1 = True
            winner = "Победа Игрока №2"
            winner_add = "(Все корабли противника Игрока №1 подбиты)!!"
            print(winner, winner_add)
            points1 = [[10 for i in range(s_x)] for i in range(s_y)]
            points2 = [[10 for i in range(s_x)] for i in range(s_y)]
            id1 = canvas.create_rectangle(step_x*3, size_canvas_y // 2, size_canvas_x + menu_x + size_canvas_x-step_x*3, size_canvas_y // 2+step_y+step_y // 2 + 
            list_ids.append(id1)
            id2 = canvas.create_rectangle(step_x * 3+step_x//2, size_canvas_y // 2 +step_y//2,
                                          size_canvas_x + menu_x + size_canvas_x - step_x * 3 - step_x//2,
                                          size_canvas_y // 2+step_y+step_y // 2 + 50 + 25 + step_y // 2 - step_y//2, fill="#364483")
            list_ids.append(id2)
            id3 = canvas.create_text(size_canvas_x+menu_x//2, size_canvas_y // 2+step_y+step_y // 2, text=winner, font=("Arial", 50), justify=CENTER)
            id4 = canvas.create_text(size_canvas_x+menu_x//2, size_canvas_y // 2+step_y+step_y // 2 + 50, text=winner_add, font=("Arial", 25), justify=CENTER)
            list_ids.append(id3)
            list_ids.append(id4)
```

Игра с компьютером

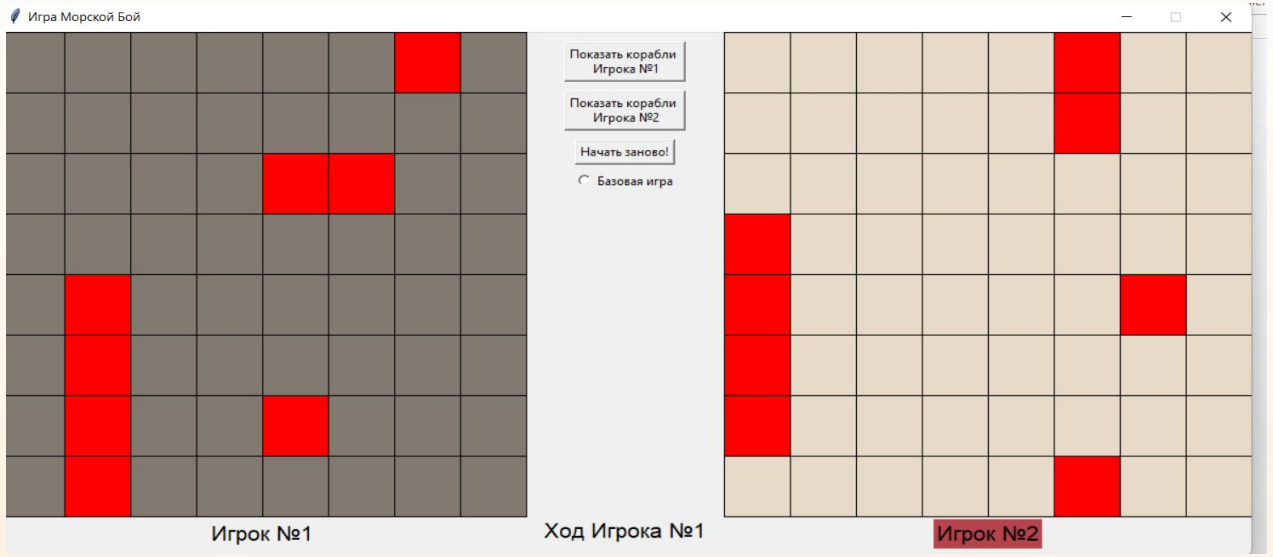
```
def mark_igrok(igrok_mark_1):
    if igrok_mark_1:
        t0.configure(bg="#b5444c")
        t0.configure(text="Игрок №1"+add_to_label2)
        t0.place(x=size_canvas_x // 2 - t0.winfo_reqwidth() // 2, y=size_canvas_y + 3)
        t1.configure(text="Игрок №2" + add_to_label)
        t1.place(x=size_canvas_x + menu_x + size_canvas_x // 2 - t1.winfo_reqwidth() // 2, y=size_canvas_y + 3)
        t1.configure(bg="#f0f0f0")
        t3.configure(text="Ход Игрока №2"+add_to_label)
        t3.place(x=size_canvas_x + menu_x // 2 - t3.winfo_reqwidth() // 2, y=size_canvas_y)
    else:
        t1.configure(bg="#b5444c")
        t0.configure(bg="#f0f0f0")
        t0.configure(text="Игрок №1")
        t0.place(x=size_canvas_x // 2 - t0.winfo_reqwidth() // 2, y=size_canvas_y + 3)
        t1.configure(text="Игрок №2" + add_to_label)
        t1.place(x=size_canvas_x + menu_x + size_canvas_x // 2 - t1.winfo_reqwidth() // 2, y=size_canvas_y + 3)
        t3.configure(text="Ход Игрока №1")
        t3.place(x=size_canvas_x + menu_x // 2 - t3.winfo_reqwidth() // 2, y=size_canvas_y)
mark_igrok(hod_igrovomu_polu_1)
```

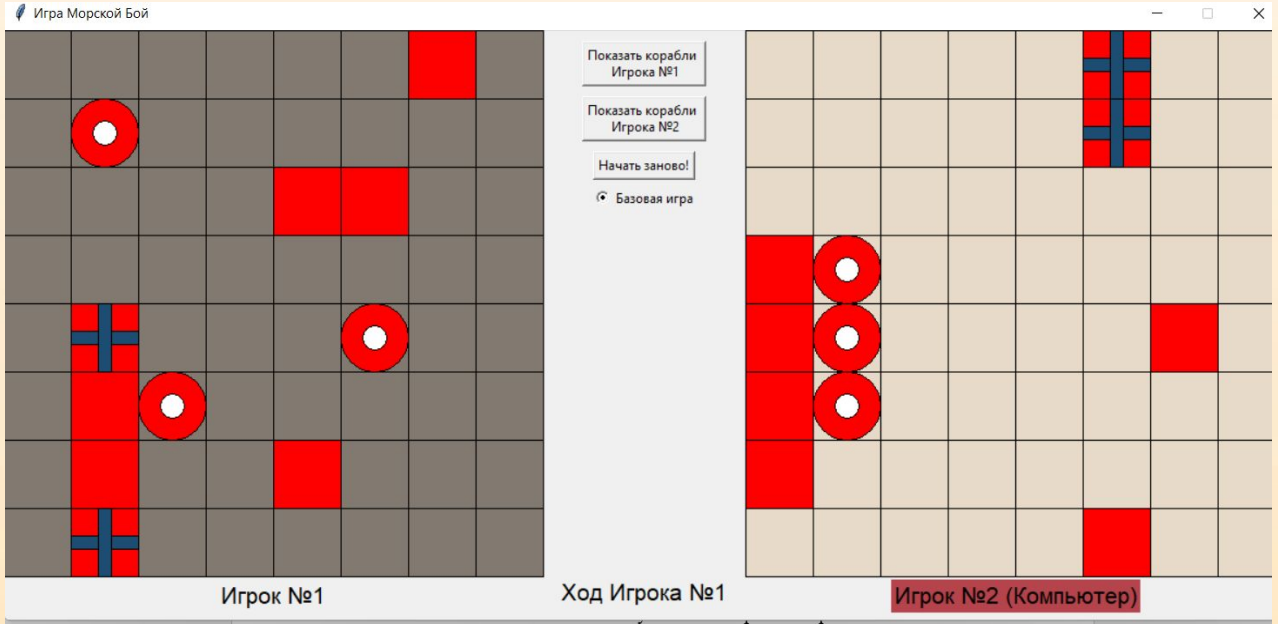
```
def hod_computer():
    global points1, points2, hod_igrovomu_polu_1
    tk.update()
    time.sleep(1)
    hod_igrovomu_polu_1 = False
    ip_x = random.randint(0, s_x-1)
    ip_y = random.randint(0, s_y-1)
    #print(ip_x, ip_y)
    while not points1[ip_y][ip_x] == -1:
        ip_x = random.randint(0, s_x-1)
        ip_y = random.randint(0, s_y-1)
    points1[ip_y][ip_x] = 7
    draw_point(ip_x, ip_y)
```

Интерфейс игры



Расстановка кораблей

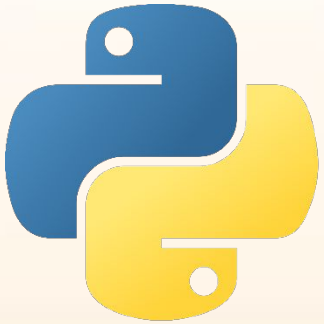




Конец игры


Игра Морской Бой
Показать корабли Игрока №1
Показать корабли Игрока №2
Начать заново!
Базовая игра
Победа Игрока №1
(Все корабли противника Игрока №2 подбиты)!!
Игрок №1
Ход Игрока №1
Игрок №2 (Компьютер)

Заключение



В результате курсовой работы было решено множество задач.

Во-первых, были изучены основные теоретические понятия языка программирования Python, включая его историю, сведения о синтаксисе и версиях.

Во-вторых, была изучена история создания игры “Морской бой”, её правила и ключевые моменты игры.

Затем, была реализована сама игра “Морской бой” на языке программирования Python, его циклов и функций, с использованием сторонних библиотек и модулей.

Наконец, созданная программа была протестирована на предмет ошибок.

Спасибо за внимание