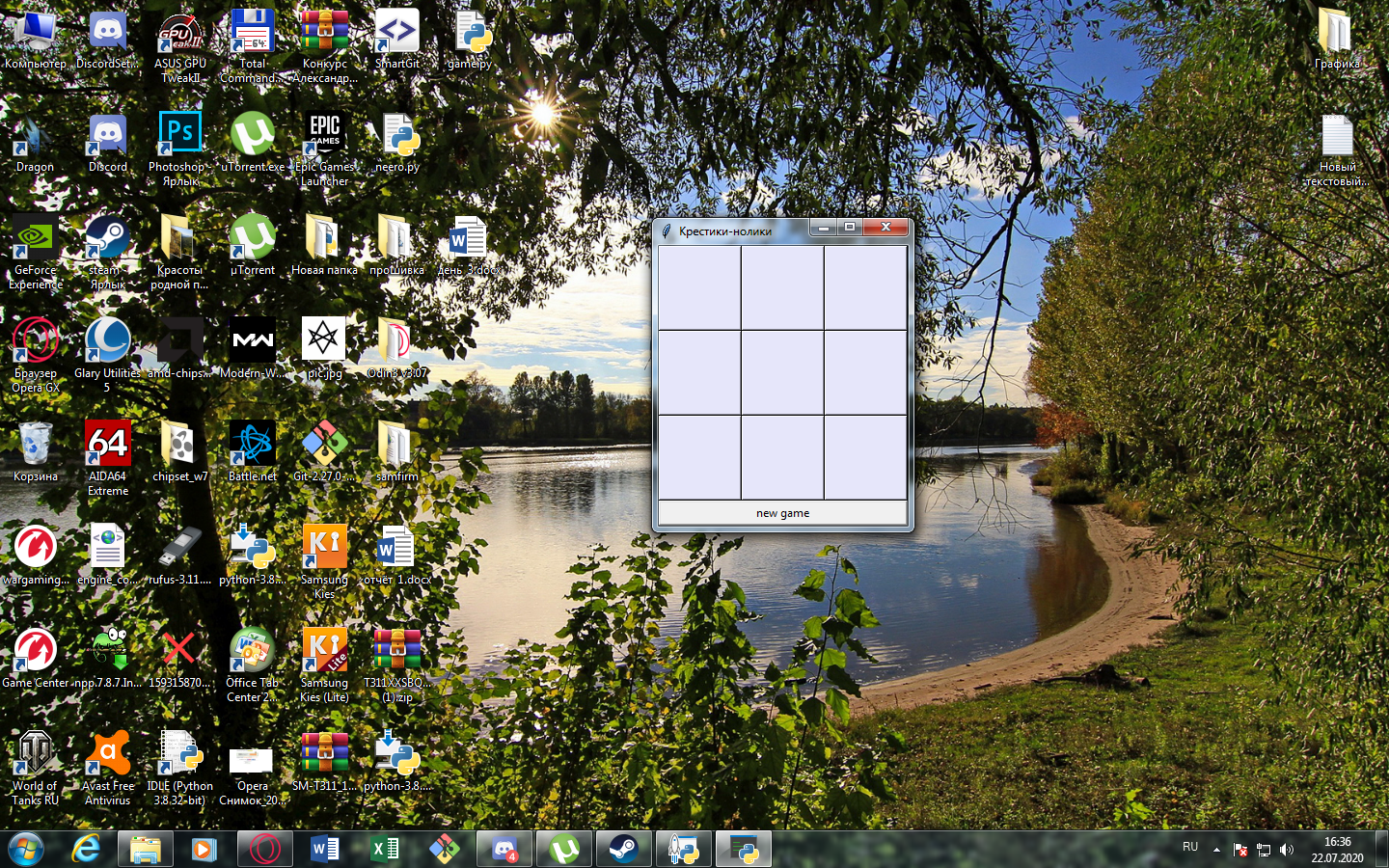
**О программе**

В этой игре можно сыграть с компьютером. Первый ход за игроком. Но победить искусственный интеллект в этом противостоянии будет не так уж и просто. Компьютер не делает «зевков» и если у него есть шанс победить, он непременно им воспользуется.



Внизу расположена кнопка для начала новой игры. По её нажатию игровое поле очистится и можно будет начать игру заново.

Игрок ставит крестики, а компьютер нолики. Как всегда, побеждает тот, кто первый составит линию из своих символов: по горизонтали, вертикали или диагонали.

**Библиотеки и объявление переменных**

Для отображения графики будем использовать стандартную библиотеку Tkinter, которая устанавливается вместе с Python. Так же нам потребуется библиотека random для того, чтобы получать [случайные](https://all-python.ru/osnovy/sluchajnoe-chislo.html) числа, благодаря которым ходы компьютера будут неожиданными.

Это также стандартная библиотека Python. Так что ничего дополнительно устанавливать не надо. Просто подключаем их с помощью Import. Создаем окно Root, устанавливаем ему заголовок и **объявляем необходимые переменные**:

* game\_run – в эту переменную будем записывать False при завершении игры, чтобы запретить делать ходы когда уже выявлен победитель.
* field – это будет двумерный список, в котором будут храниться кнопки игрового поля. Ходом будет изменение надписи на кнопке на символ «X» или «O».
* cross\_count в этой переменной мы будем отслеживать количество крестиков на поле. Чтобы по выставлению пятого крестика, в случае если никто не выиграл фиксировать ничью.

from tkinter import \*

import random

root = Tk()

root.title('Criss-cross')

game\_run = True

field = []

cross\_count = 0

**Обработка нажатия кнопок**

Функция new\_game будет вызываться при нажатии кнопки начала новой игры. На поле убираются все крестики и нолики. Цвет кнопок делаем бледно-лиловым. Устанавливаем глобальные переменные game\_run и cross\_count в начальные значения. Это [глобальные переменные](https://all-python.ru/osnovy/oblast-vidimosti.html) к которым пытаемся обратиться из функции. Поэтому перед попыткой изменить их значение, в Python надо использовать ключевое слово global.

def new\_game():

for row in range(3):

for col in range(3):

field[row][col]['text'] = ' '

field[row][col]['background'] = 'lavender'

global game\_run

game\_run = True

global cross\_count

cross\_count = 0

Функция click будет вызываться после нажатия на поле, то есть при попытки поставить крестик. Если игра еще не завершена, то крестик ставится. После этого увеличиваем счетчик количества выставленных крестиков.

Потом проверяем с помощью функции check\_win, не победили ли мы этим ходом. Если еще не выявлен победитель и есть еще ходы, то выполняет ход компьютер функцией computer\_move, и также после хода идет проверка выигрыша.

def click(row, col):

if game\_run and field[row][col]['text'] == ' ':

field[row][col]['text'] = 'X'

global cross\_count

cross\_count += 1

check\_win('X')

if game\_run and cross\_count < 5:

computer\_move()

check\_win('O')

**Проверка победы**

Функция check\_win осуществляет проверку выигрыша. Она перебирает все возможные комбинации полей, образующих линию и вызывает с ними функцию check\_line. Переменная smb – это символ «X» или «O», то есть крестики или нолики. Если задан «O», то проверяется: не победил ли компьютер.

Если зафиксирован выигрыш, то меняем цвет фона кнопок, составляющих линию на розовый. А также записываем в game\_run значение False.

def check\_win(smb):

for n in range(3):

check\_line(field[n][0], field[n][1], field[n][2], smb)

check\_line(field[0][n], field[1][n], field[2][n], smb)

check\_line(field[0][0], field[1][1], field[2][2], smb)

check\_line(field[2][0], field[1][1], field[0][2], smb)

def check\_line(a1,a2,a3,smb):

if a1['text'] == smb and a2['text'] == smb and a3['text'] == smb:

a1['background'] = a2['background'] = a3['background'] = 'pink'

global game\_run

game\_run = False

Проверяем все возможные варианты, так как теоретически можно одним ходом составить сразу 2 линии.

**Действия компьютера**

Ход компьютера рассчитывается в функции computer\_move. **Алгоритм его действий следующий**:

1. Проверка возможности победы. Если компьютеру представился шанс победы – он не должен его упустить. Сразу же делает победу.
2. Проверка возможной победы противника за один ход. Если игрок выставил два крестика в ряд, компьютер пытается разрушить планы игрока.
3. Случайный ход. Так как победить нет возможности и нет угрозы проигрыша, то выбирается случайное свободное поле. В бесконечном цикле wile перебираются случайные числа, пока они не выпадут на не занятое поле.

def can\_win(a1,a2,a3,smb):

res = False

if a1['text'] == smb and a2['text'] == smb and a3['text'] == ' ':

a3['text'] = 'O'

res = True

if a1['text'] == smb and a2['text'] == ' ' and a3['text'] == smb:

a2['text'] = 'O'

res = True

if a1['text'] == ' ' and a2['text'] == smb and a3['text'] == smb:

a1['text'] = 'O'

res = True

return res

def computer\_move():

for n in range(3):

if can\_win(field[n][0], field[n][1], field[n][2], 'O'):

return

if can\_win(field[0][n], field[1][n], field[2][n], 'O'):

return

if can\_win(field[0][0], field[1][1], field[2][2], 'O'):

return

if can\_win(field[2][0], field[1][1], field[0][2], 'O'):

return

for n in range(3):

if can\_win(field[n][0], field[n][1], field[n][2], 'X'):

return

if can\_win(field[0][n], field[1][n], field[2][n], 'X'):

return

if can\_win(field[0][0], field[1][1], field[2][2], 'X'):

return

if can\_win(field[2][0], field[1][1], field[0][2], 'X'):

return

while True:

row = random.randint(0, 2)

col = random.randint(0, 2)

if field[row][col]['text'] == ' ':

field[row][col]['text'] = 'O'

break

**Графический интерфейс**

Все элементы графического интерфейса мы будем размещать с помощью упаковщика grid. В цикле добавим кнопки игрового поля. Они будут храниться в двумерном список. В языке программирования Python добавляют элементы в [список](https://all-python.ru/osnovy/spiski.html) с помощью метода append.

Свойство colorspan у кнопки начала игры выставляем в 3, чтобы он занимал всю ширину таблицы

for row in range(3):

line = []

for col in range(3):

button = Button(root, text=' ', width=4, height=2,

font=('Verdana', 20, 'bold'),

background='lavender',

command=lambda row=row, col=col: click(row,col))

button.grid(row=row, column=col, sticky='nsew')

line.append(button)

field.append(line)

new\_button = Button(root, text='new game', command=new\_game)

new\_button.grid(row=3, column=0, columnspan=3, sticky='nsew')

root.mainloop()