

Муниципальное образовательное учреждение -  
средняя образовательная школа №4 им. В. Бурова

**Информационный проект на тему:  
«Язык программирования Python»**

Проект подготовил ученик 10 класса  
Соколов Виктор Александрович  
Руководитель Забелина М.В.

г. Бежецк  
2020-2021 уч. год

## Содержание

|   |    |
|---|----|
| 1. Введение.....  | 3  |
| 2. Глава 1. Теоретическая часть. Возможности языка программирования.....        | 5  |
| 3. Глава 2. Практическая часть. Создание игры «Камень, ножницы,<br>бумага»..... | 10 |
| 4. Заключение.....  | 12 |
| 5. Список использованной литературы.....  | 13 |

## Введение

В учебной школьной программе, присутствуют темы и практические работы заданиями на написание программ, где на выбор даются разные языки программирования. В школьной программе учащихся обучают несколько устаревшему языку Pascal, однако, по-моему, мнению, ему есть куда более простая альтернатива в виде Python, его простота и универсальность делают его одним из лучших языков программирования на данный момент.

**Актуальность работы** заключается в изучение современного и широко распространенного языка программирования, который подойдет для изучения на школьном уровне, так и пригодится для сдачи ЕГЭ по информатике и ИКТ.

**Цель проекта** - обучиться основам языка программирования Python на уровне наших школьных знаний о ЯП Pascal.

### **Задачи проекта:**

- выяснить возможности языка программирования,
- создание информационного продукта,
- изучить синтаксис языка, условные операторы, циклы.

**Гипотеза:** я предполагаю, что данный язык программирования является более востребованным и значительно увеличит быстроту решения задач на программирование в рамках сдачи ЕГЭ по информатике.

В ходе работы были использованы следующие **методы:**

1. Теоретические: анализ источников, систематизация полученных данных;
2. Эмпирические: компьютерное моделирование, сравнение.

### **Этапы реализация проекта**

#### **I этап. Подготовительный. Сентябрь 2020 г.**

Актуализация темы изучения языка программирования.

Разработка пошаговой программы деятельности.

#### **II этап. Изучение синтаксиса и возможностей языка программирования. Октябрь – ноябрь 2020 г.**

Систематизация полученной информации из интернет-источников.

Изучение этапов реализации программы.

Подготовка материалов для создания компьютерной игры.

#### **III этап. Творческий. Декабрь - апрель 2021 г.**

Создание компьютерной игры на основе языка программирования Python.

Запись видеоролика для демонстрации приложения.

#### **IV этап. Заключительный. Май 2021 г.**

Презентация работы педагогическому коллективу, учащимся школы.

**Практическая значимость работы:** язык программирования Python становится все более популярным и востребованным. Многие компьютерные игры и мобильные приложения (например, Сбербанк) написаны именно с

помощью этого языка. Поэтому я считаю, что его изучение пригодится в моей будущей профессии.

# Глава 1. Теоретическая часть. Возможности языка программирования

## Установка Python

Установить язык программирования python можно с официального сайта. Программа бесплатная. Заходим на сайт <https://python.org/downloads/windows/> и выбираем "latest python release" и python 3 (см. рисунок 1).



Рисунок 1. Установка программы.

Вот лишь некоторые вещи, которые умеет делать python:

- работа с xml/html файлами,
- работа с http запросами,
- GUI (графический интерфейс),
- создание веб-сценариев,
- работа с FTP,
- работа с изображениями, аудио и видео файлами,
- робототехника.
- программирование математических и научных вычислений.

Таким образом, python подходит для решения львиной доли повседневных задач, будь то резервное копирование, чтение электронной почты, либо же какая-нибудь игрушка. Язык программирования Python практически ничем не ограничен, поэтому также может использоваться в крупных проектах. К примеру, python интенсивно применяется IT-гигантами, такими как, например, Google и Yandex.

### Синтаксис ЯП

- Конец строки является концом инструкции (точка с запятой не требуется).
- Вложенные инструкции объединяются в блоки по величине отступов. Отступ может быть любым, главное, чтобы в пределах одного

вложенного блока отступ был одинаков. И про читаемость кода не забывайте. Отступ в 1 пробел, к примеру, не лучшее решение. Используйте 4 пробела (или знак табуляции, на худой конец).

- Вложенные инструкции в Python записываются в соответствии с одним и тем же шаблоном, когда основная инструкция завершается двоеточием, вслед за которым располагается вложенный блок кода, обычно с отступом под строкой основной инструкции.

## Арифметические операторы

### Операторы присваивания (см. рисунок 2)

| Оператор | Пример  | Так же как |
|----------|---------|------------|
| =        | x = 5   | x = 5      |
| +=       | x += 3  | x = x + 3  |
| -=       | x -= 3  | x = x - 3  |
| *=       | x *= 3  | x = x * 3  |
| /=       | x /= 3  | x = x / 3  |
| %=       | x %= 3  | x = x % 3  |
| //=      | x //= 3 | x = x // 3 |
| **=      | x **= 3 | x = x ** 3 |
| &=       | x &= 3  | x = x & 3  |
| =        | x  = 3  | x = x   3  |
| ^=       | x ^= 3  | x = x ^ 3  |
| >>=      | x >>= 3 | x = x >> 3 |
| <<=      | x <<= 3 | x = x << 3 |

Рисунок 2. Операторы присваивания

### Синтаксис инструкции if

Условная инструкция if-elif-else (её ещё иногда называют оператором ветвления) - основной инструмент выбора в Python. Проще говоря, она выбирает, какое действие следует выполнить, в зависимости от значения переменных в момент проверки условия.

Сначала записывается часть `if` с условным выражением, далее могут следовать одна или более необязательных частей `elif`, и, наконец, необязательная часть `else`.

Общая форма записи условной инструкции `if` выглядит следующим образом (см. рисунок 3):

```
if test1:
    state1
elif test2:
    state2
else:
    state3
```

Рисунок 3. Условный оператор

Простой пример (напечатает 'true', так как 1 - истина) (см. рисунок 4):

```
>>> if 1:
...     print('true')
... else:
...     print('false')
...
true
```

Рисунок 4. Простой пример условного оператора

Чуть более сложный пример (его результат будет зависеть от того, что ввёл пользователь) (см. рисунок 5):

```
if
```

Рисунок 5. Сложный пример условного оператора.

### Проверка истинности в Python

- Любое число, не равное 0, или непустой объект - истина.
- Числа, равные 0, пустые объекты и значение `none` - ложь

- Операции сравнения применяются к структурам данных рекурсивно
- Операции сравнения возвращают True или False
- Логические операторы and и or возвращают истинный или ложный объект-операнд

Трехместное выражение if/else (см. рисунок 6):

```
if X:
    A = Y
else:
    A = Z
```

Рисунок 6. Трехместное выражение if/else

Довольно короткая, но, тем не менее, занимает целых 4 строки. Специально для таких случаев и было придумано выражение if/else (см. рисунок 7):

```
A = Y if X else Z
```

Рисунок 7. Выражение if/else

### Цикл while

While - один из самых универсальных циклов в Python, поэтому довольно медленный. Выполняет тело цикла до тех пор, пока условие цикла истинно (см. рисунок 8).

```
>>> i = 5
>>> while i < 15:
...     print(i)
...     i = i + 2
...
5
7
9
11
13
```

Рисунок 8. Цикл while

### Цикл for

Цикл for уже чуточку сложнее, чуть менее универсальный, но выполняется гораздо быстрее цикла while. Этот цикл проходится по любому итерируемому объекту (например, строке или списку), и во время каждого прохода выполняет тело цикла (см. рисунок 9).



```

>>> for i in 'hello world':
...     if i == 'o':
...         break
...     print(i * 2, end='')
...
hheelllll

```

Рисунок 9. Цикл for

### Оператор continue и break

Оператор continue начинает следующий проход цикла, минуя оставшееся тело цикла, а break досрочно прерывает цикл (for или while) (см. рисунок 10).

```

>>> for i in 'hello world':
...     print(i * 2, end='')
...
hheellllloo  wwoorrllldd

```

```

>>> for i in 'hello world':
...     if i == 'o':
...         continue
...     print(i * 2, end='')
...
hheelllll  wwrrllldd

```

Рисунок 10. Оператор continue и break

## Глава 2. Практическая часть. Создание игры «Камень, ножницы, бумага»

В своей практической части я решил показать возможности языка программирования на примере создания игры «Камень, ножницы, бумага».

Тест программы:

```
fromtkinterimport *
```

```
import random asrdm
```

```
classMain(Frame):
```

```
def__init__(self, root):
```

```
super(Main, self).__init__(root)
```

```
self.startUI()
```

```
defstartUI
```

```
(self).btn = Button(root, text="Камень", font=("Times New Roman", 15),  
command=lambda x=1: self.btn_click(x))
```

```
btn2 = Button(root, text="Ножницы", font=("Times New Roman", 15),  
command=lambda x=2: self.btn_click(x))
```

```
btn3 = Button(root, text="Бумага", font=("Times New Roman", 15),  
command=lambda x=3: self.btn_click(x))
```

```
btn.place(x=10, y=100, width=120, height=50)
```

```
btn2.place(x=155, y=100, width=120, height=50)
```

```
btn3.place(x=300, y=100, width=120, height=50)
```

```
self.lbl = Label(root, text="Началойгры!", bg="#FFF", font=("Times New  
Roman", 21, "bold"))
```

```
self.lbl.place(x=150, y=25)
```

```
self.win = self.drow = self.lose = 0
```

```
self.lbl2 = Label(root, justify="left", font=("Times New Roman", 13),  
text=f"Побед: {self.win}\nПроигрышей:"f" {self.lose}\nНичей: {self.drow}",  
bg="#FFF") self.lbl2.place(x=5, y=5)
```

```
defbtn_click(self, choice):
```

```
comp_choice = rdm.randint(1, 3)
```

```
ifchoice == comp_choice:
```

```
self.drow += 1
```

```

self.lbl.configure(text="Ничья")
elifchoise == 1andcomp_choise == 2 \
orchoise == 2andcomp_choise == 3 \
orchoise == 3andcomp_choise == 1:
self.win += 1
self.lbl.configure(text="Победа")
else:
self.lose += 1
self.lbl.configure(text="Проигрыш")
self.lbl2.configure(text=f"Побед: {self.win}\nПроигрышей:"f" {self.lose}\
nНичей: {self.drow}")
delcomp_choise
if __name__ == '__main__':
root = Tk()
root.geometry("430x160+200+200")
root.title("Камень, ножницы, бумага")
root.resizable(False, False)
root["bg"] = "#FFF"
app = Main(root)
app.pack()
root.mainloop()

```

## **Заключение**

В ходе выполнения данного проекта я ознакомил вас с основными возможностями Python для решения задач школьного уровня.

Благодаря простоте и гибкости языка Python, его можно рекомендовать пользователям (математикам, физикам, экономистам и так далее) не являющимся программистами, но использующими вычислительную технику и программирование в своей работе.

Программы на Python разрабатываются в среднем в полтора-два (а порой и в два-три) раза быстрее, нежели на компилируемых языках (C, C++, Pascal). Поэтому, язык может представлять не малый интерес и для профессиональных программистов, разрабатывающих приложения, не критичные к скорости выполнения, а также программы, использующие сложные структуры данных. В частности, Python хорошо зарекомендовал себя при разработке программ работы с графами, генерации деревьев.

Можно сделать вывод о том, что запланированные задачи в ходе проекта были выполнены.

Созданный информационный ресурс может быть использован для самостоятельного изучения языка программирования Python.

## Список использованной литературы

### Интернет-ресурсы:

Python 3 для начинающих <https://pythonworld.ru/osnovy/cikly-for-i-while-operator-break-i-continue-volshebnoe-slovo-else.html>

Уроки по Python для начинающих <https://pythonru.com/uroki/operatory-v-python-uroki-po-python-dlja-nachinajushhih>

Уроки по Python <https://itproger.com/course/python>

Установка программы <https://www.python.org>