



Государственное бюджетное образовательное учреждение высшего образования
Московской области

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ

Колледж космического машиностроения и технологий

ОТЧЕТ

По учебной практике УП.01.01 Разработка программных модулей
программного обеспечения для компьютерных систем
специальность 09.02.03 Программирование в компьютерных системах

Выполнили студенты:

Планкин И.И. и Волков М.А.

_____ (подпись)

Гусятинер Л.Б.

_____ (подпись)

_____ (оценка)

Королев, 2020

Содержание

2.1. Установка интерпретатора Python 3. М.	3
2.2. Настройка окружения. М.	5
3.1. Техника работы в командной строке. М.	8
3.2. Техника работы в среде IDLE. М.	9
4.1 Техника работы с линейными программами. И.	11
4.2. Техника работы с разветвляющимися программами. И.	13
5.1. Техника работы с циклическими программами _ цикл while. И.	16
5.2. Техника работы с циклическими программами _ цикл while. И.	16
6.1. Техника работы с числами. М.	19
6.2. Техника работы с числами. И.	20
7.1. Техника работы со строками. М.	26
7.2. Техника работы со строками. М.	29
8.1. Техника работы со списками И.	30
8.2. Техника работы со списками. И.	31
9.1. Техника работы с циклом for и генераторами списков. И.	33
9.2. Техника работы с циклом for и генераторами списков. И.	34
10.1. Техника работы с функциями. М.	37
10.2. Техника работы с функциями. М.	37
11.1. Техника работы со словарями. И.	38
11.2. Техника работы со словарями. М.	38
12.1. Техника работы с множествами. М.	39
12.2. Техника работы с множествами. И.	39
13.1. Техника работы с кортежами И.	41
13.2. Техника работы с кортежами. И.	41
14.1. Техника работы с файлами. И.	42
15.1. Техника работы с модулями. М.	43
15.2. Техника работы с модулями. М.	43
15.3. Техника работы с модулями. М.	44
15.4. Техника работы с модулями. М.	45
16.1. Техника работы с классами. И.	46
16.2. Техника работы с классами. И.	46
16.3. Техника работы с классами. И.	47
16.4. Техника работы с классами. И.	48
Список литературы:	49

Если перед названием темы стоит буква М то задание выполнил Волков М.А., а если стоит И то выполнил Планкин И.И.

2.1. Установка интерпретатора Python 3. М.

1. Установить актуальную версию Python на Windows, Linux и подготовить руководство (видео / презентацию / веб-страницы)
2. Быть готовым продемонстрировать полностью процесс установки с пояснениями.



Рис.1 Установка нужных вам пакетов

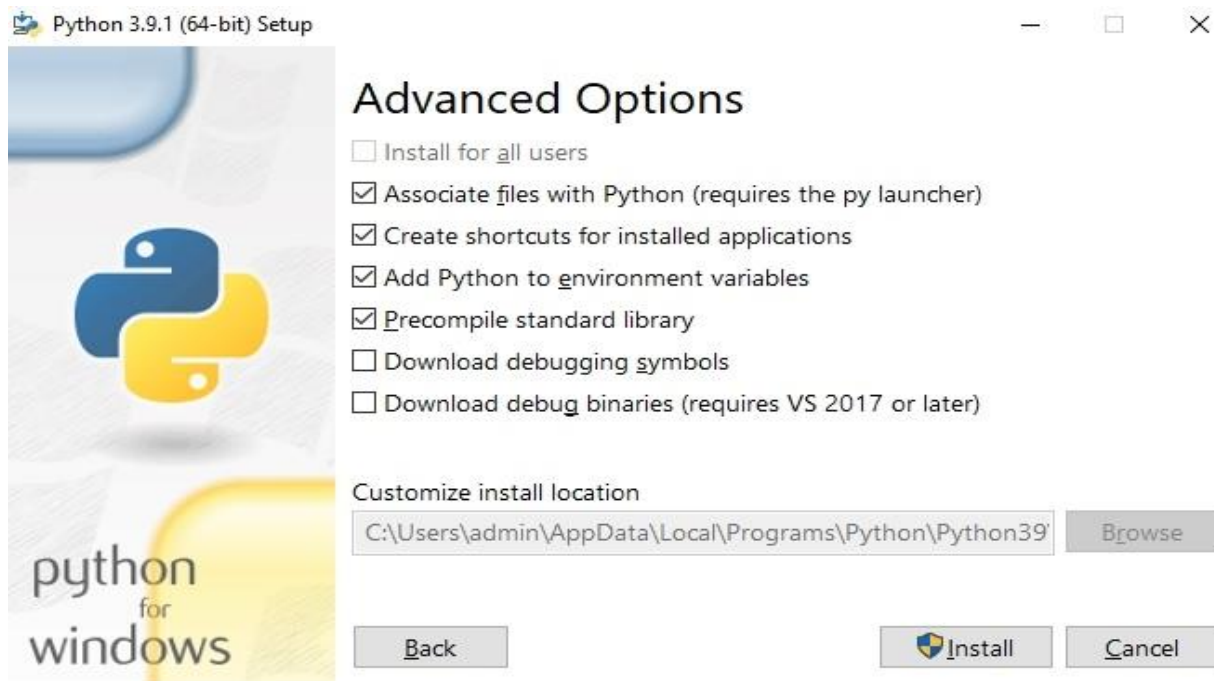


Рис. 2 Выбор оптимального Места для установки

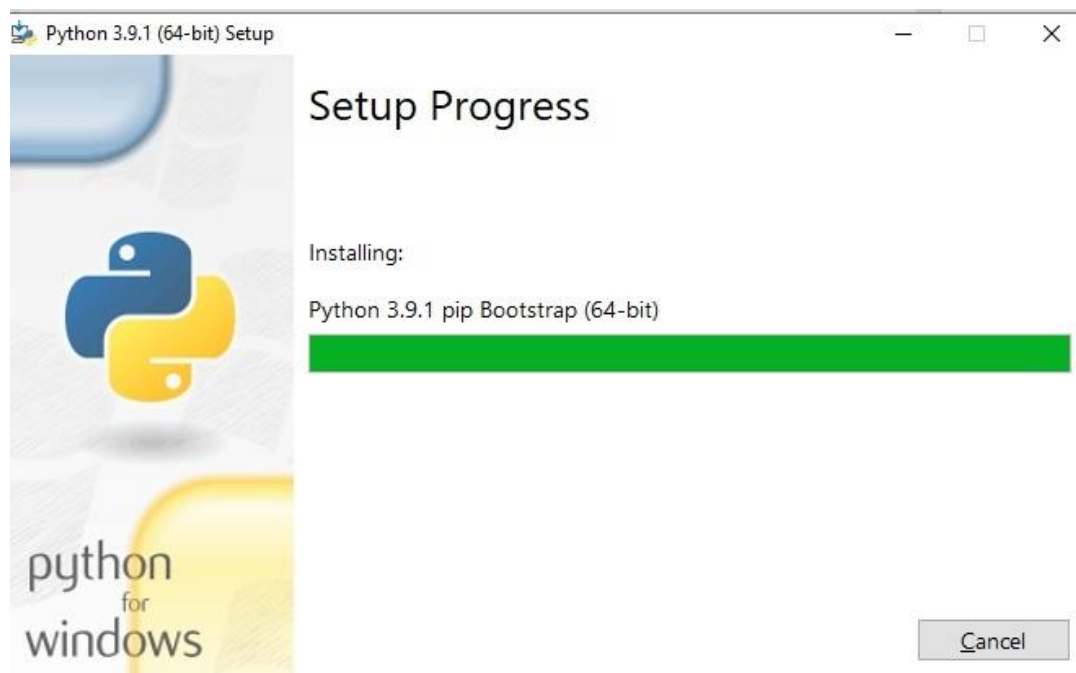


Рис. 3 Установка Python.

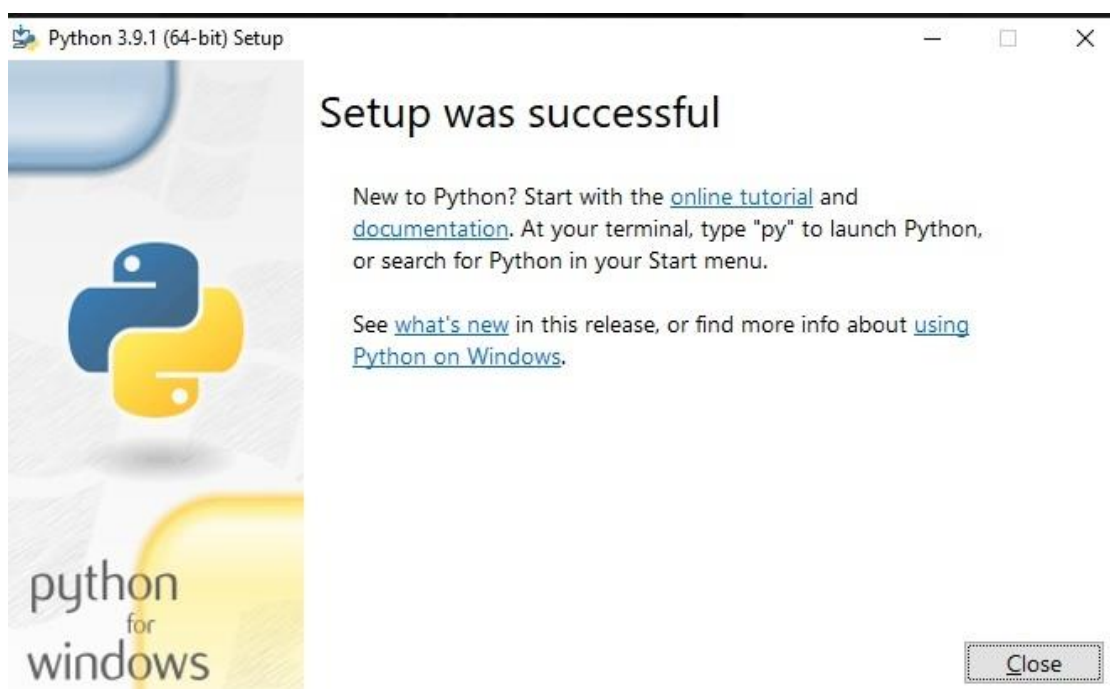


Рис 4. Завершение установки

2.2. Настройка окружения. М.

УП.01 Учебная практика по модулю ПМ.01

Консультация 2-2. Настройка окружения интерпретатора Python 3

Составитель: Гусятинер Л.Б., 24.11.2020, МГОТУ ККМТ, П1-18, П2-18

Задание.

1. Продемонстрировать процесс настройки окружения

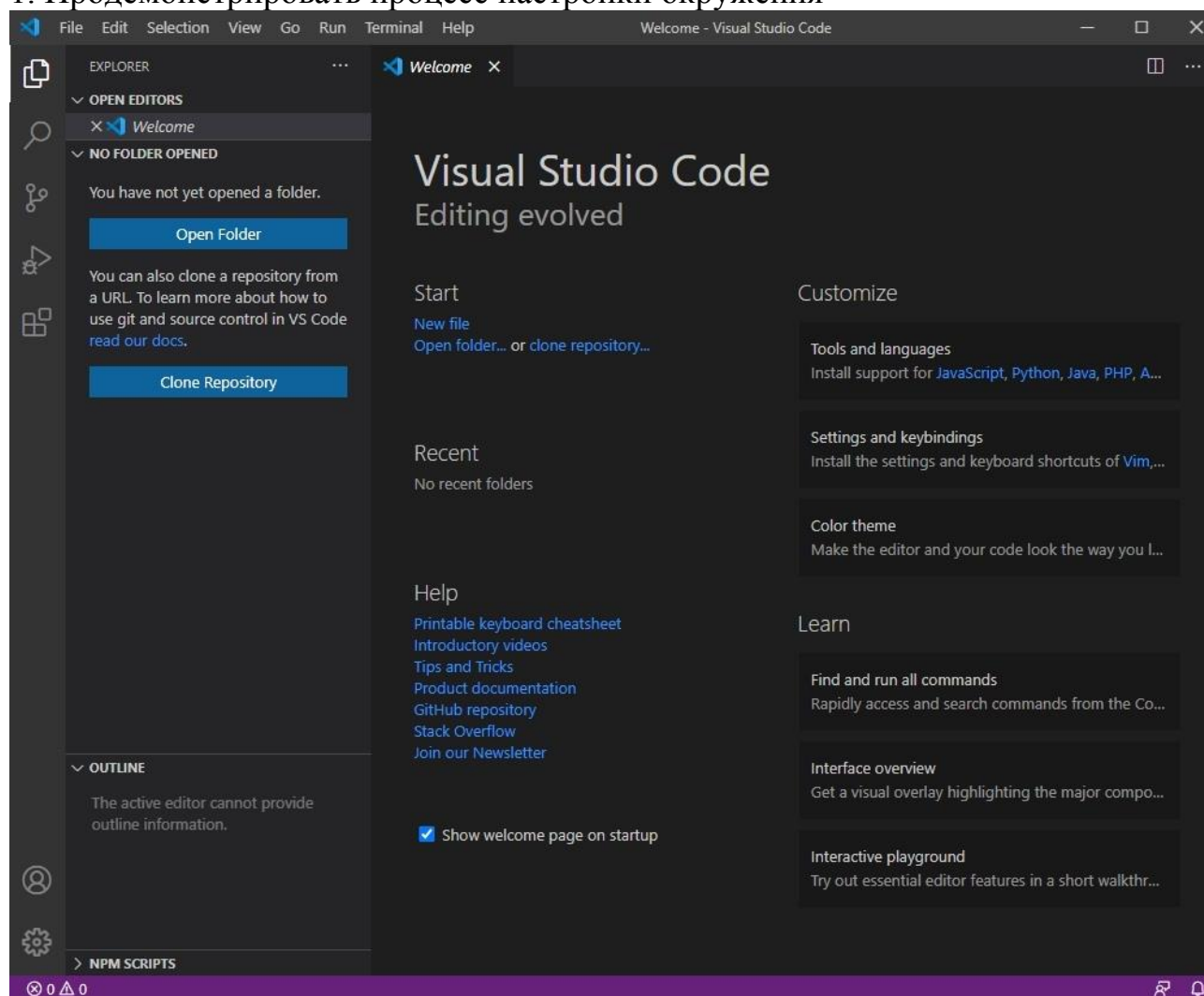


Рис.5

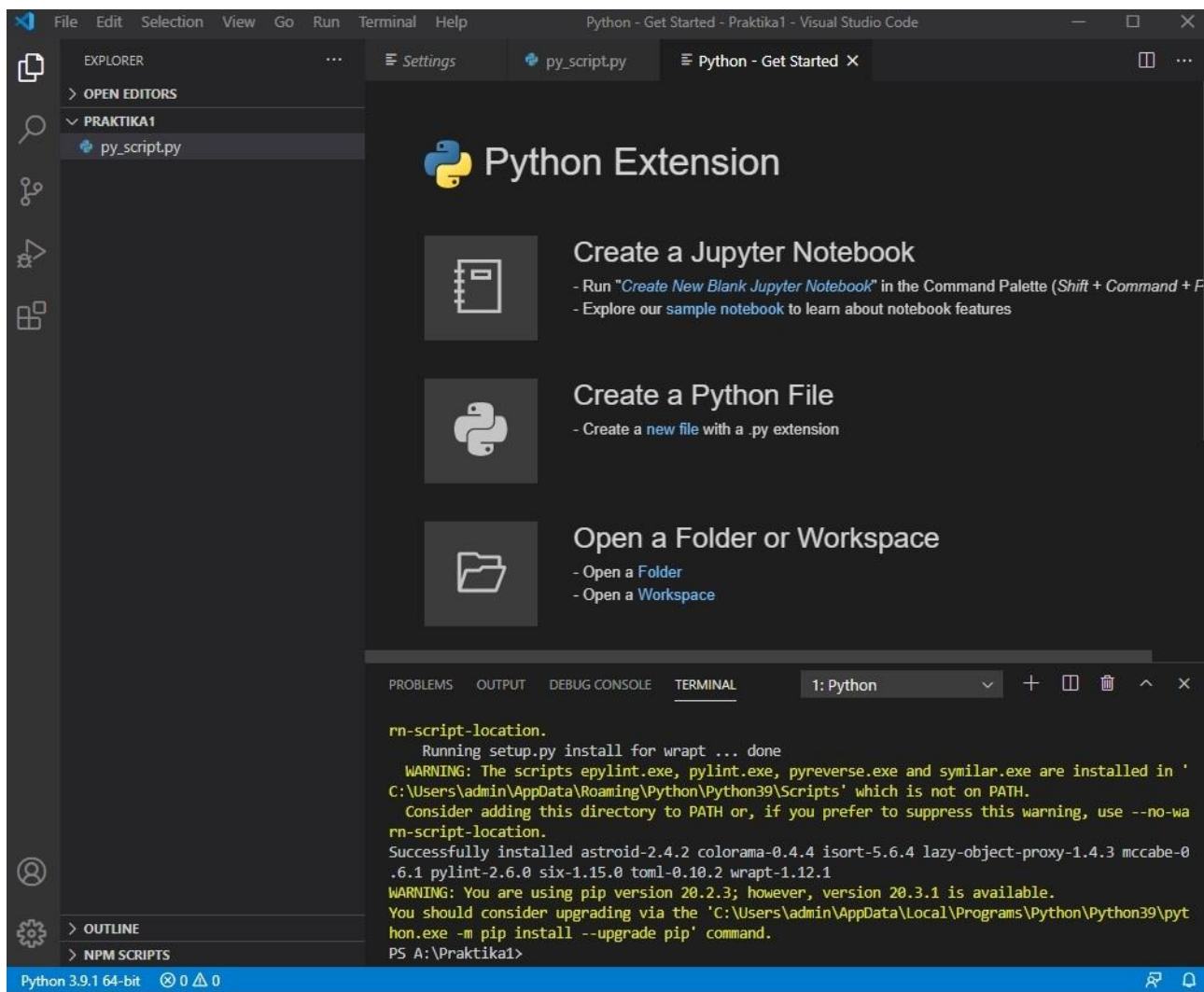


Рис.6

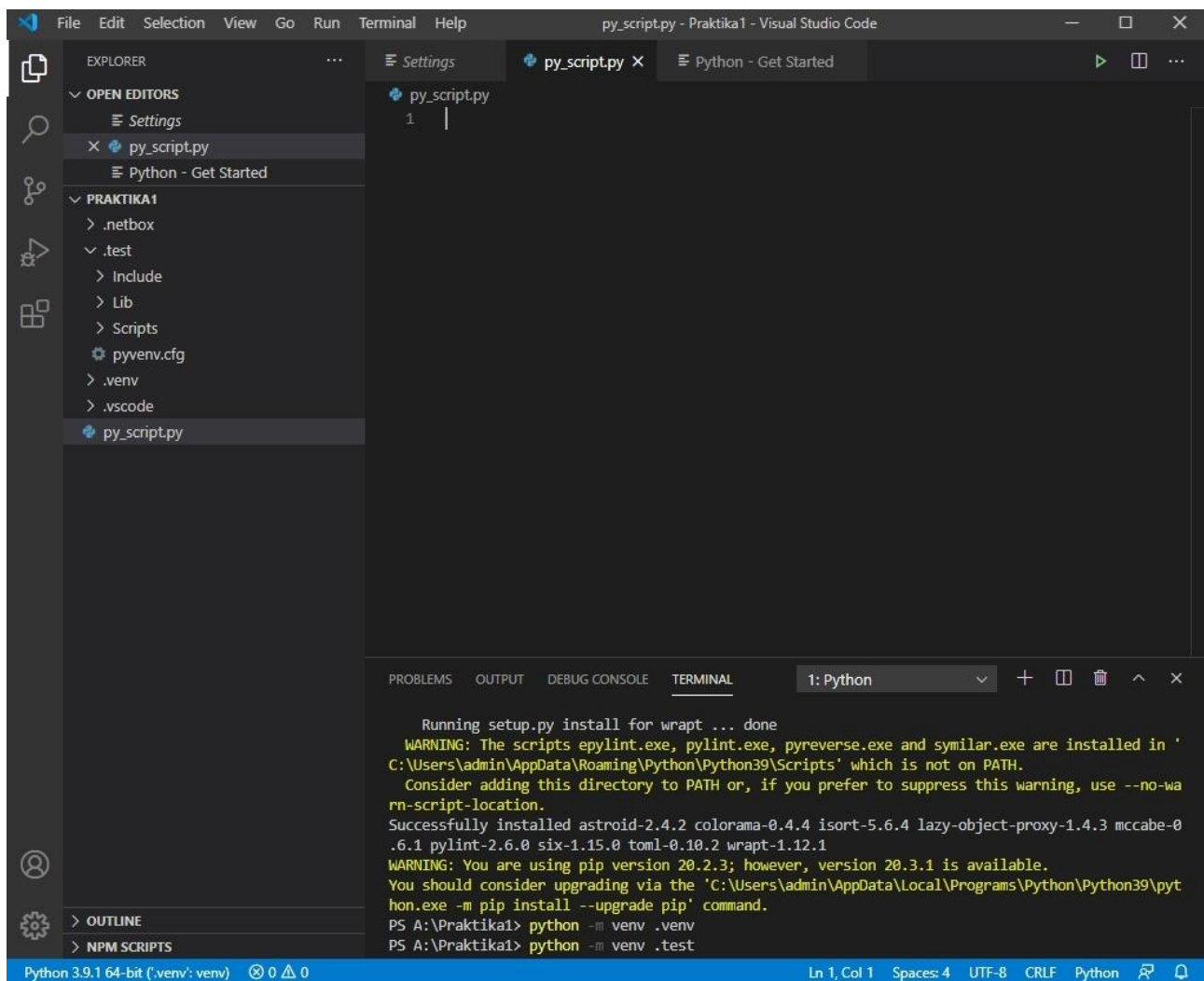


Рис.6

3.1. Техника работы в командной строке. М.

УП.01 Учебная практика по модулю ПМ.01

Консультация 3. Техника работы в командной строке и среде IDLE

3-1.

Составитель: Гусятинер Л.Б., 24.11.2020, МГОТУ ККМТ, П1-18, П2-18

Задание.

1. Продемонстрировать работу в командной строке, включая

- создание файла с кодом
- запуск
- import
- reload
- отработку ошибок

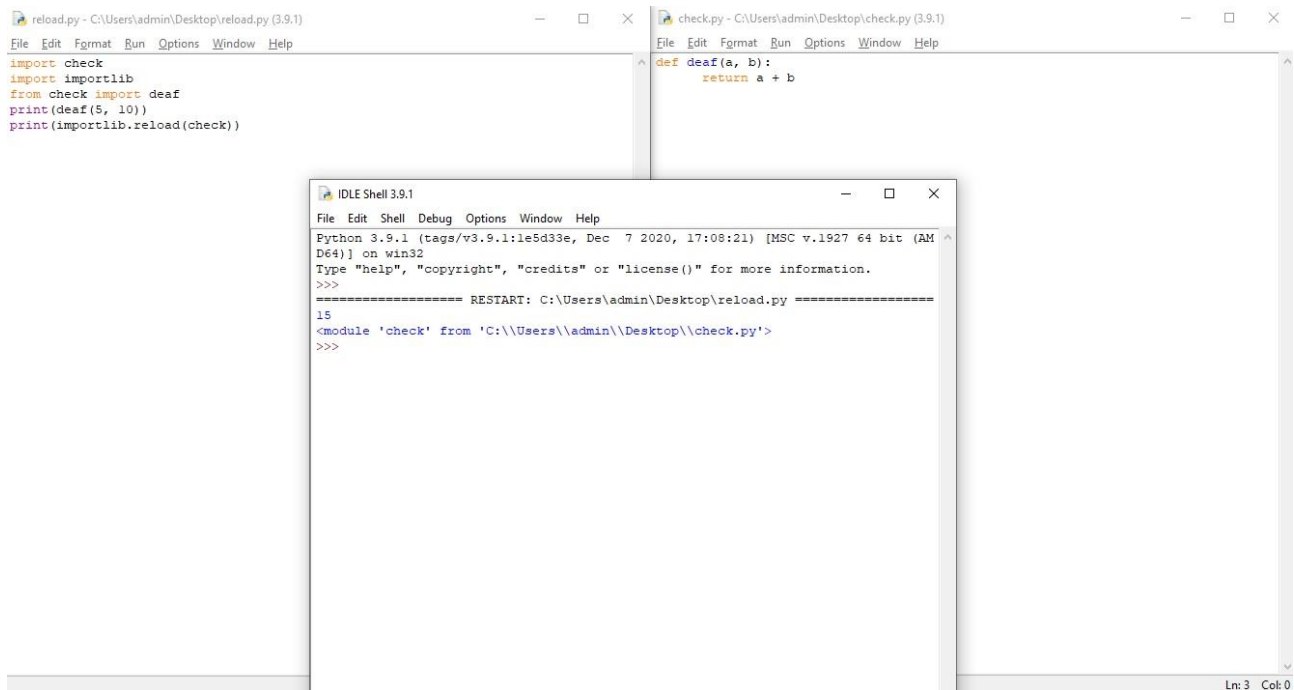


Рис.7 Демонстрация работы.

3.2. Техника работы в среде IDLE. М.

УП.01 Учебная практика по модулю ПМ.01

Консультация 3. Техника работы в командной строке и среде IDLE 3-1.

Составитель: Гусятинер Л.Б., 24.11.2020, МГОТУ ККМТ, П1-18, П2-18

Задание.

1. Показать работу в оболочке IDLE как в самой среде, так и путём запуска файлов.

2. Показать умение работать с меню.

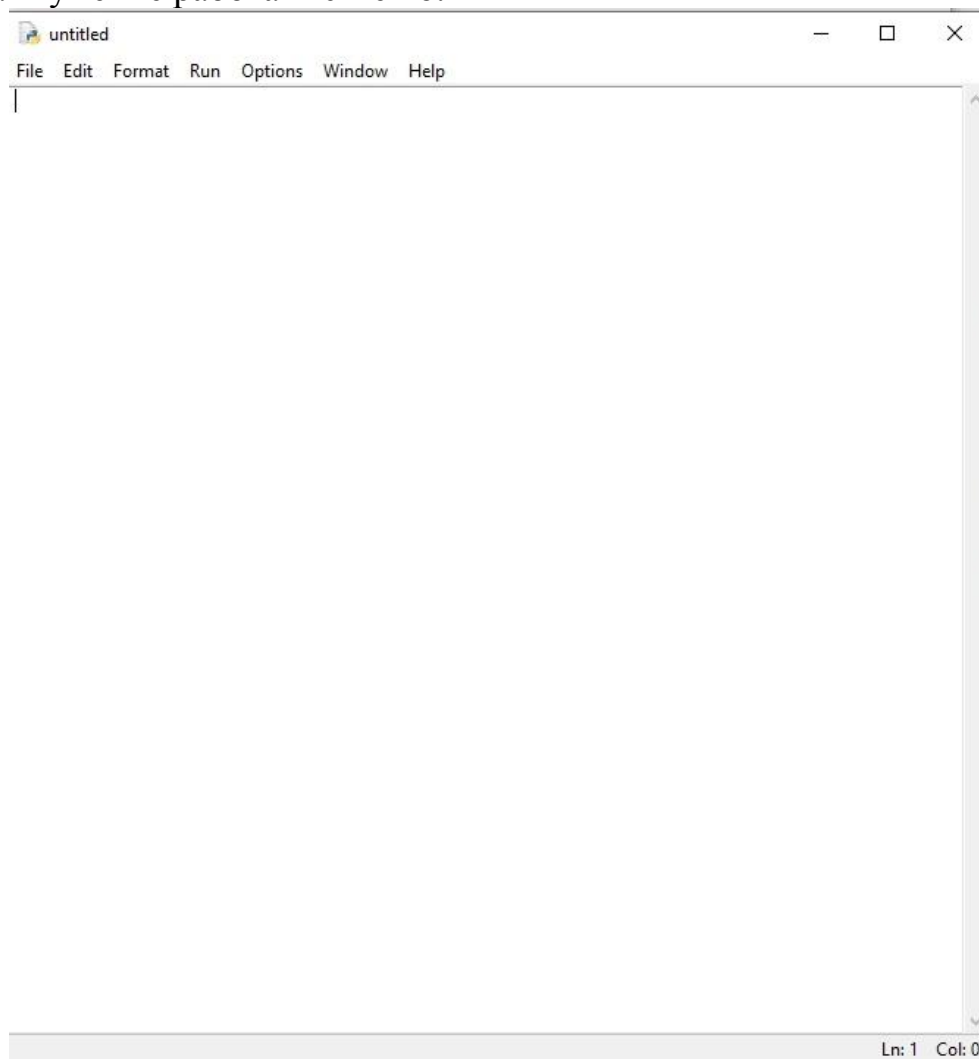
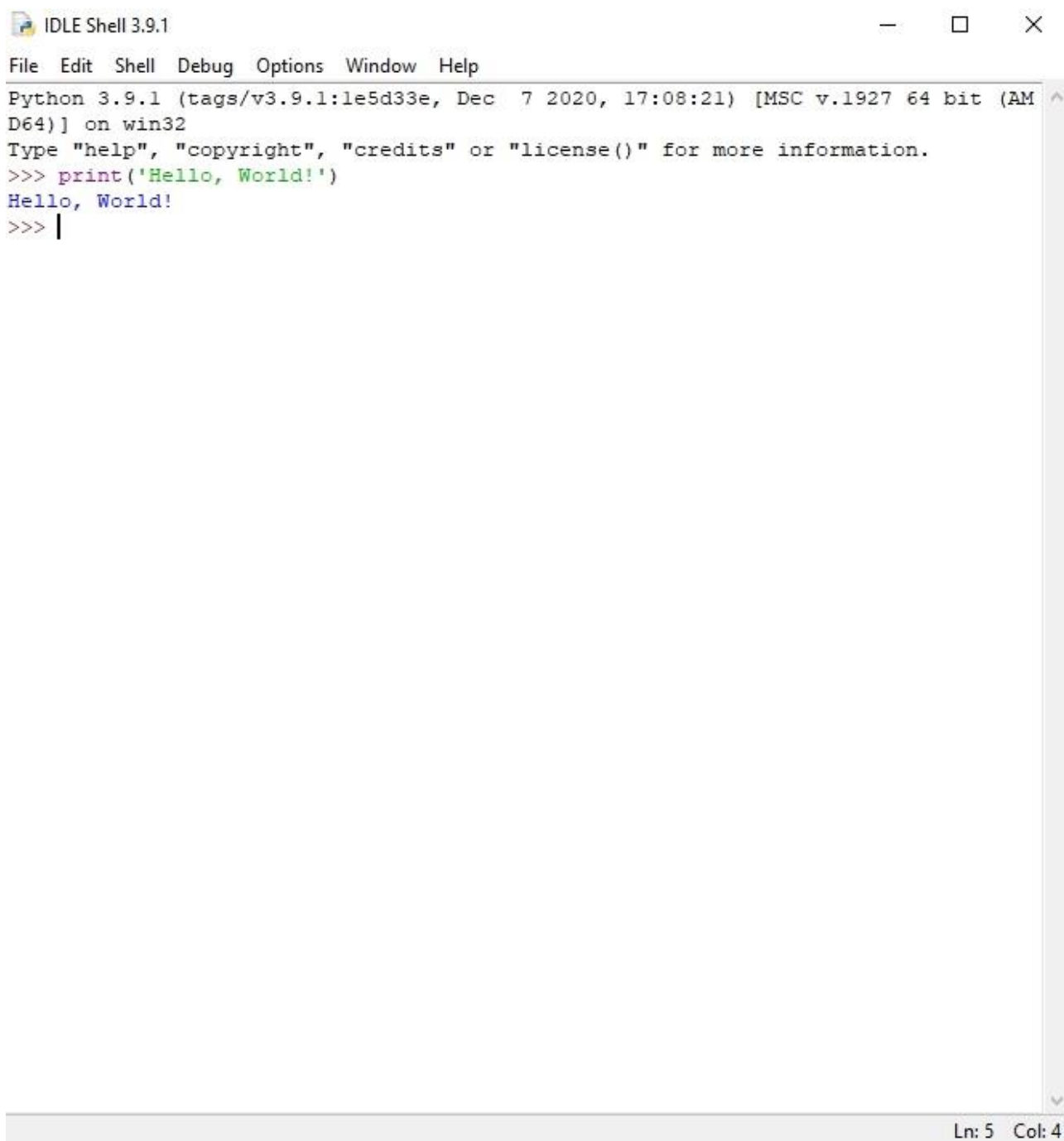


Рис.8 Открыть новый пустой файл.



The image shows a screenshot of the IDLE Shell 3.9.1 window. The window has a title bar with the text "IDLE Shell 3.9.1" and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with the following items: File, Edit, Shell, Debug, Options, Window, and Help. The main text area contains the following text:

```
Python 3.9.1 (tags/v3.9.1:1e5d33e, Dec 7 2020, 17:08:21) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print('Hello, World!')
Hello, World!
>>> |
```

The text is color-coded: "print" is in purple, "Hello, World!" is in green, and the prompt ">>>" is in blue. A vertical cursor is positioned at the end of the last line. The status bar at the bottom right of the window shows "Ln: 5 Col: 4".

Рис. 9 Демонстрация работы

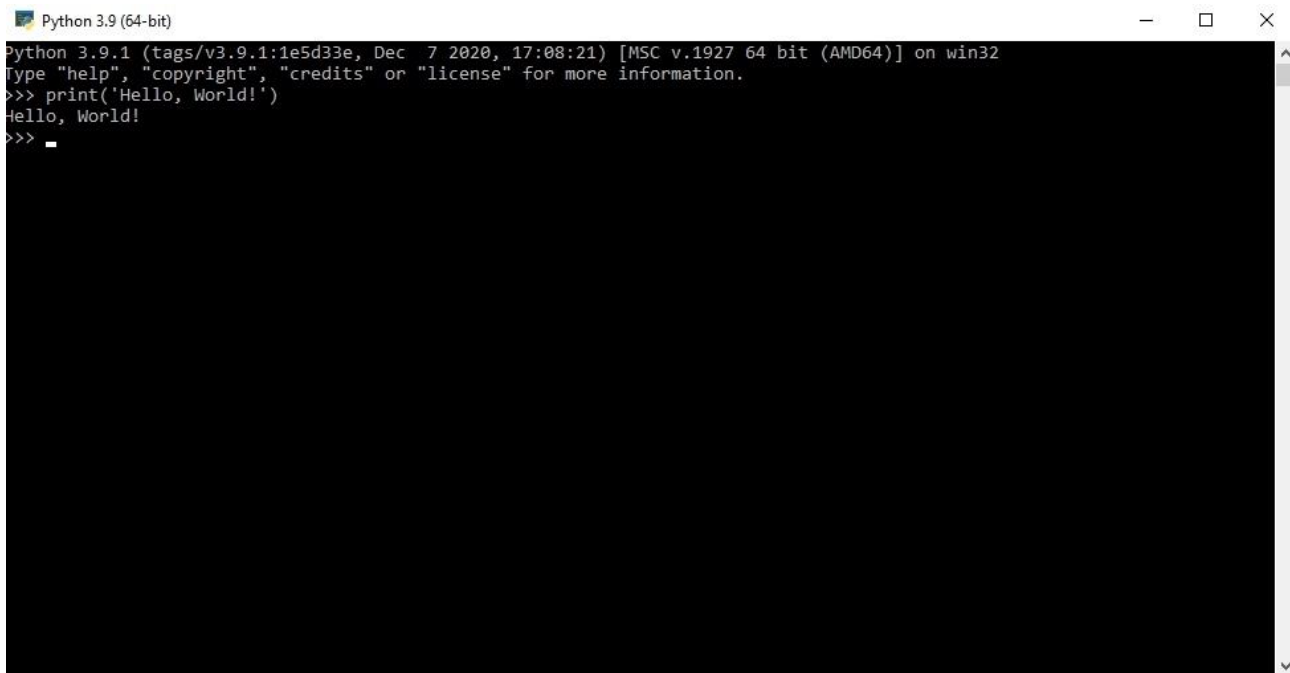


Рис.10 Демонстрация работы

4.1 Техника работы с линейными программами. II.

- print

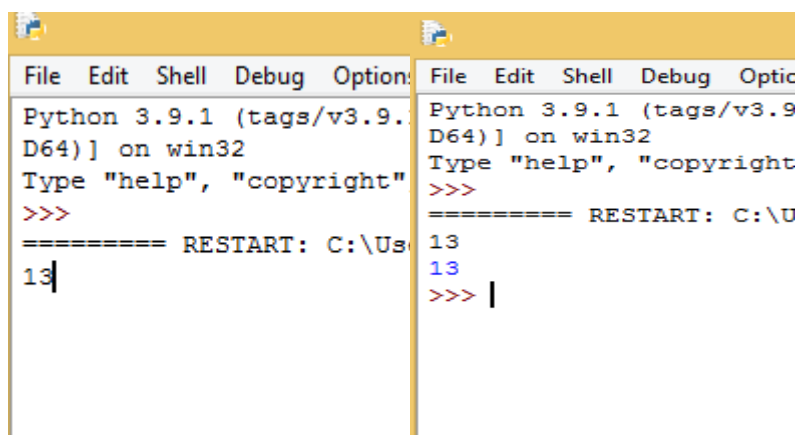
Функция print() в Python, печатает объект.

- input

Функция input() в Python, ввод данных с клавиатура.

```
a = input()
print(a)
```

Пример рабочей программы.

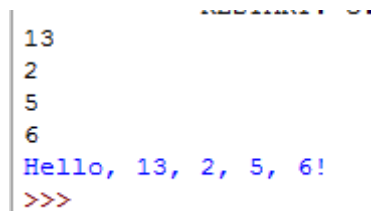


Рисуноки 11 и 12. Демонстрация работы программы.

- форматная строка и метод формат

```
print('Hello, {0}, {1}, {2}, {3}!'.format(input(),  
input(),input(),input()))
```

Пример кода на формат.



```
13  
2  
5  
6  
Hello, 13, 2, 5, 6!  
>>>
```

Рисунок 13. Демонстрация работы программы.

4.2. Техника работы с разветвляющимися программами. И.

Задание 1. Разработать программу для печати даты прописью

Пример ввода: 15.12.1983

Пример вывода: Пятнадцатое декабря одна тысяча девятсот восемьдесят третьего года

```
def get_date(date):
    days = ['первое', 'второе', 'третье', 'четвёртое',
            'пятое', 'шестое', 'седьмое', 'восьмое',
            'девятое', 'десятое', 'одиннадцатое', 'двенадцатое',
            'тринадцатое', 'четырнадцатое', 'пятнадцатое', 'шестнадцатое',
            'семнадцатое', 'восемнадцатое', 'девятнадцатое', 'двадцатое',
            'двадцать первое', 'двадцать второе', 'двадцать третье',
            'двадцать четвёртое', 'двадцать пятое', 'двадцать шестое',
            'двадцать седьмое', 'двадцать восьмое', 'двадцать
            девятое', 'тридцатое', 'тридцать первое']

    months = ['января', 'февраля', 'марта', 'апреля', 'мая',
              'июня',
              'июля', 'августа', 'сентября', 'октября', 'ноября',
              'декабря']

    date = date.split('.')
    return (days[int(date[0]) - 1] + ' ' + months[int(date[1]) -
1] + ' ' + date[2] + ' года')

date = input()
print(get_date(date))
```

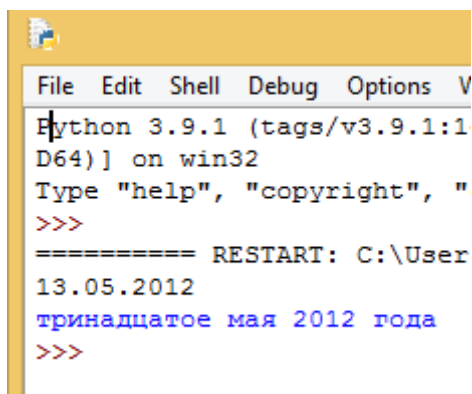


Рисунок 14. Демонстрация работы программы.

Задание 2. Разработать программу с меню для демонстрации работы с типами данных:

список(list), словарь(dict), множество(set)

Меню -> выбор типа данных -> выбор метода -> краткая справка

```
options = int(input("Choose from Menu:\n"
                    "list: 1.\n"
                    "dict: 2.\n"
                    "set: 3.\n"))
```

```
listMethods = ['list.append(x)      Добавляет элемент в конец
списка\n',
'list.extend(L)      Расширяет список list, добавляя в конец все
элементы списка L\n',
'list.insert(i, x)   Вставляет на i-ый элемент значение x\n',
'list.remove(x)      Удаляет первый элемент в списке, имеющий
значение x. ValueError, если такого элемента не существует\n'
'list.pop([i])      Удаляет i-ый элемент и возвращает его. Если индекс
не указан, удаляется последний элемент\n'
'list.index(x, [start [, end]])     Возвращает положение первого
элемента со значением x (при этом поиск ведется от start до
end)\n'
'list.count(x)      Возвращает количество элементов со значением x\n']
```

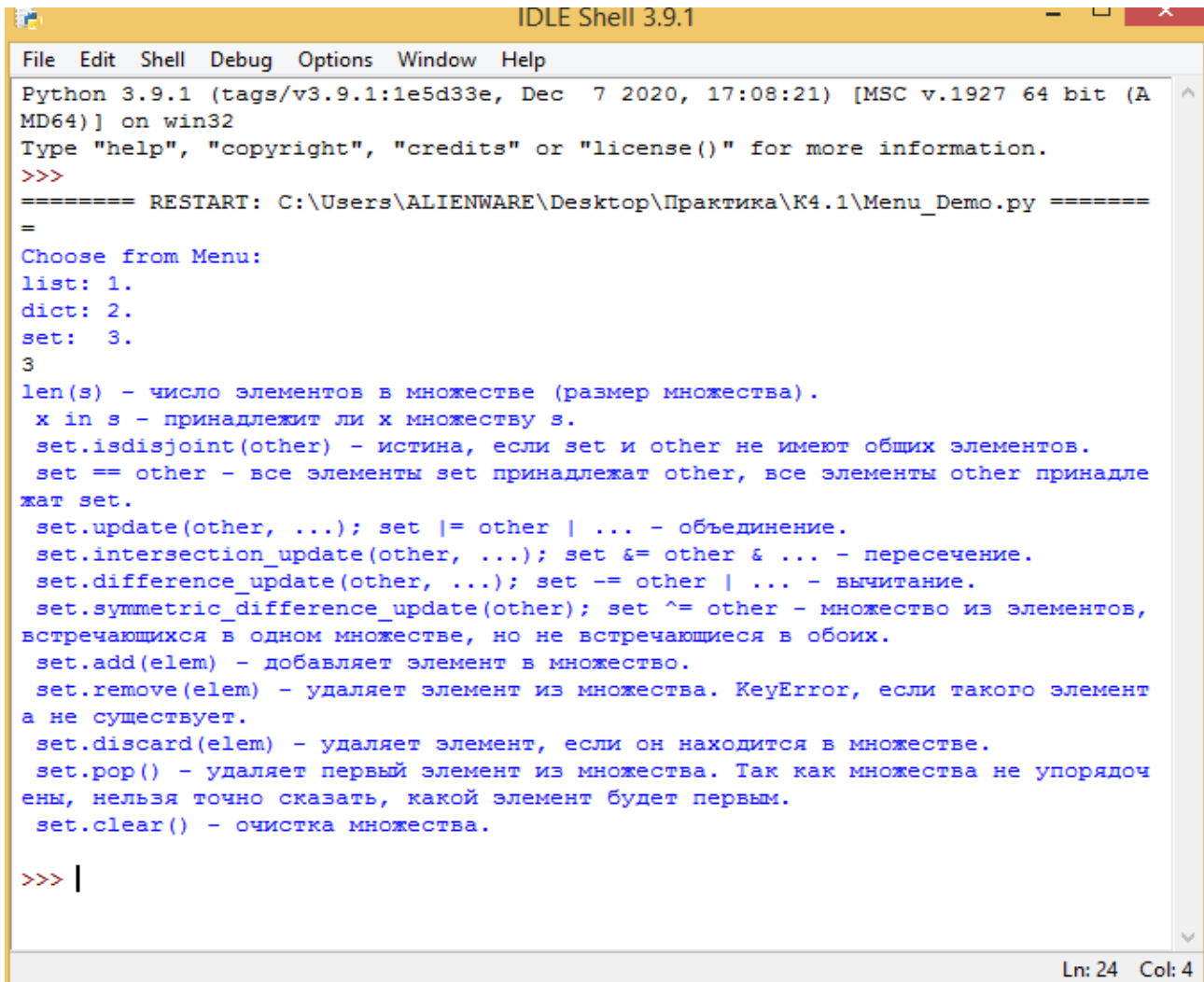
```
dictMethods = ['dict.clear() - очищает словарь.\n',
'dict.copy() - возвращает копию словаря.\n',
'classmethod dict.fromkeys(seq[, value]) - создает словарь с
ключами из seq и значением value (по умолчанию None).\n',
'dict.get(key[, default]) - возвращает значение ключа, но если его
нет, не бросает исключение, а возвращает default (по умолчанию
None).\n',
'dict.items() - возвращает пары (ключ, значение).\n',]
```

```
setMethods = ['len(s) - число элементов в множестве (размер
множества).\n',
'x in s - принадлежит ли x множеству s.\n',
'set.isdisjoint(other) - истина, если set и other не имеют общих
элементов.\n',
'set == other - все элементы set принадлежат other, все элементы
other принадлежат set.\n',
'set.update(other, ...); set |= other | ... - объединение.\n',
'set.intersection_update(other, ...); set &= other & ... -
пересечение.\n',
'set.difference_update(other, ...); set -= other | ... -
вычитание.\n',
'set.symmetric_difference_update(other); set ^= other - множество
из элементов, встречающихся в одном множестве, но не встречающиеся
в обоих.\n',
'set.add(elem) - добавляет элемент в множество.\n',
'set.remove(elem) - удаляет элемент из множества. KeyError, если
такого элемента не существует.\n',
```

```
'set.discard(elem) - удаляет элемент, если он находится в
множестве.\n',
'set.pop() - удаляет первый элемент из множества. Так как
множества не упорядочены, нельзя точно сказать, какой элемент
будет первым.\n',
'set.clear() - очистка множества.\n']
```

```
menuList = [listMethods, dictMethods, setMethods]
```

```
print(*menuList[options - 1])
```



```
IDLE Shell 3.9.1
File Edit Shell Debug Options Window Help
Python 3.9.1 (tags/v3.9.1:1e5d33e, Dec 7 2020, 17:08:21) [MSC v.1927 64 bit (A
MD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\ALIENWARE\Desktop\Практика\K4.1\Menu_Demo.py =====
=
Choose from Menu:
list: 1.
dict: 2.
set: 3.
3
len(s) - число элементов в множестве (размер множества).
x in s - принадлежит ли x множеству s.
set.isdisjoint(other) - истина, если set и other не имеют общих элементов.
set == other - все элементы set принадлежат other, все элементы other принадле
жат set.
set.update(other, ...); set |= other | ... - объединение.
set.intersection_update(other, ...); set &= other & ... - пересечение.
set.difference_update(other, ...); set -= other | ... - вычитание.
set.symmetric_difference_update(other); set ^= other - множество из элементов,
встречающихся в одном множестве, но не встречающихся в обоих.
set.add(elem) - добавляет элемент в множество.
set.remove(elem) - удаляет элемент из множества. KeyError, если такого элемент
а не существует.
set.discard(elem) - удаляет элемент, если он находится в множестве.
set.pop() - удаляет первый элемент из множества. Так как множества не упорядоч
ены, нельзя точно сказать, какой элемент будет первым.
set.clear() - очистка множества.

>>> |
```

Рисунок 15. Демонстрация работы программы.

5.1. Техника работы с циклическими программами _ цикл while. И.

Задача №2.

Придумать пример(ы) на использование break / continue /else.

```
count = 0
while (1):
    a = int(input())
    count += 1
    if (count >= 3):
        break
    else:
        continue
```

Входные данные:

3
5
5

Вывод:

5.2. Техника работы с циклическими программами _ цикл while. И.

Задача №1.

Вычислить значение $\sin(x)$ с точностью до epsilon при помощи разложения в ряд

Построить блок-схему.

```
def y(b):
    def foo(x):
        if x < 1:
            return x**2 - 2
        elif x == 1:
            return 1 + x
        else:
            return math.sqrt(b**3 + x**2)
    return foo

#test

y1 = y(1.25)
for a in map(y1, [0.0, 0.5, 1.0]):
    print(a)
```

Вывод:

-2.0
-1.75
2.0

Задача №2.

Напишите программу, которая считывает со стандартного ввода целые числа, по одному числу в строке, и после первого введенного нуля выводит сумму полученных на вход чисел.

```
a = int(input())
b = 0
while a != 0:
    b = b + a
    a = int(input())
print(b)
```

Входные данные:

```
5
-3
8
4
0
```

Вывод:

```
14
```

Задание 3.

Разработать программу для нахождения наибольшего общего делителя

#Алгоритм Евклида для нахождения НОД

```
a = int(input())
b = int(input())

while a!=0 and b!=0:
    if a > b:
        a = a % b
    else:
        b = b % a

print (a+b)
```

Входные данные:

```
12
64
```

Вывод:

```
4
```

Задача 4.

С использованием результата задания 2 разработать программу для нахождения наименьшего общего кратного

```

a = int(input())
b = int(input())

i = min(a, b)

while True:
    if i%a==0 and i%b==0:
        break
    i += 1

print(i)

```

Входные данные:

24
124

Вывод:

744

Задание 5.

Напишите программу, которая выводит часть последовательности 1 2 2 3 3 3 4 4 4 5 5 5 5 5 ...

(число повторяется столько раз, чему равно).

На вход программе передаётся неотрицательное целое число n — столько элементов

последовательности должна отобразить программа.

На выходе ожидается последовательность чисел, записанных через пробел в одну строку.

Например, если $n = 7$, то программа должна вывести 1 2 2 3 3 3 4.

```

a=int(input())
b=''
for i in range(a+1):
    b = b + (str(i) + ' ')*i
print(b[0:a*2])

```

Входные данные:

7

Вывод:

1 2 2 3 3 3 4

6.1. Техника работы с числами. М.

Задача №1. Составить и выполнить по 3 примера использования модулей для работы с дробными числами (fractions), для точных вычислений (decimal).

Задача №2. Подготовить инструкцию по использованию модулей fractions, decimal.
#Decimal обеспечивает поддержку правильного округления десятичной арифметики с плавающей точкой.

```
from decimal import Decimal
```

```
number1 = Decimal(input())  
number2 = Decimal(input())
```

```
print(number1 + number2)
```

#Округление осуществляется с помощью метода quantize().
#В качестве первого аргумента – объект Decimal, указывающий на формат округления:

```
from decimal import Decimal  
number = Decimal(input())
```

```
print(number.quantize(Decimal('1.000')))  
# округление до 3 чисел в дробной части
```

```
print(number.quantize(Decimal('1.00')))  
# округление до 2 чисел в дробной части
```

```
print(number.quantize(Decimal('1.0')))  
# округление до 1 числа в дробной части
```

```
from fractions import Fraction
```

```
a = Fraction(input())  
b = Fraction(input())
```

```
print(a + b)
```

```
#Fraction.limit_denominator(max_denominator=1000000)  
#- ближайшее рациональное число со знаменателем не больше данного.
```

```
x = Fraction(input())  
print(x.limit_denominator)
```

6.2. Техника работы с числами. II.

Если вы пишете программу для выполнения определенных задач, таких как изучение периодического движения или моделирования электрических цепей, вам нужно будет работать с тригонометрическими функциями, а также с комплексными числами. Хотя вы не можете использовать эти функции напрямую, вы можете получить к ним доступ, включив сначала два математических модуля. Эти модули являются math и cmath.

Первый дает вам доступ к гиперболическим, тригонометрическим и логарифмическим функциям для действительных чисел, а последний позволяет работать с комплексными числами. В этом уроке я рассмотрю все важные функции, предлагаемые этими модулями. Если явно не указано, все возвращаемые значения - это float.

Модуль **math**.

Модуль **Math** в Python обеспечивает доступ к некоторым популярным математическим функциям и константам, которые можно использовать в коде для более сложных математических вычислений. Библиотека является встроенным модулем Python, поэтому никакой дополнительной установки через pip делать не нужно. В данной статье будут даны примеры часто используемых функций и констант библиотеки Math в Python.

- Первой важной математической константой является число Пи (π). Оно обозначает отношение длины окружности к диаметру, его значение 3,141592653589793.
- Число Эйлера (e) является основанием натурального логарифма. Оно также является частью модуль **Math** в Python.
- Библиотека Math в Python поставляется с функцией `exp()`, которую можно использовать для вычисления значения e. К примеру, e^x — экспонента от x. Значение e равно 2.718281828459045.
- Арифметические функции используются для представления чисел в различных формах и осуществления над ними математических операций.
- Модуль **math** в Python поддерживает все тригонометрические функции
- Python может конвертировать начальный тип числа в другой указанный тип.

*Список функций модуля **math**.*

math.ceil(X) – округление до ближайшего большего числа.

math.copysign(X, Y) - возвращает число, имеющее модуль такой же, как и у числа X, а знак - как у числа Y.

math.fabs(X) - модуль X.

math.factorial(X) - факториал числа X.

math.floor(X) - округление вниз.

math.fmod(X, Y) - остаток от деления X на Y.

math.frexp(X) - возвращает мантиссу и экспоненту числа.

math.ldexp(X, I) - $X * 2^I$. Функция, обратная функции **math.frexp()**.

math.fsum(последовательность) - сумма всех членов последовательности. Эквивалент встроенной функции **sum()**, но **math.fsum()** более точна для чисел с плавающей точкой.

math.isfinite(X) - является ли X числом.

math.isinf(X) - является ли X бесконечностью.

math.isnan(X) - является ли X NaN (Not a Number - не число).

math.modf(X) - возвращает дробную и целую часть числа X. Оба числа имеют тот же знак, что и X.

math.trunc(X) - усекает значение X до целого.

math.exp(X) - e^X .

math.expm1(X) - $e^X - 1$. При $X \rightarrow 0$ точнее, чем **math.exp(X)-1**.

math.log(X, [base]) - логарифм X по основанию base. Если base не указан, вычисляется натуральный логарифм.

math.log1p(X) - натуральный логарифм $(1 + X)$. При $X \rightarrow 0$ точнее, чем **math.log(1+X)**.

math.log10(X) - логарифм X по основанию 10.

math.log2(X) - логарифм X по основанию 2.

math.pow(X, Y) - X^Y .

math.sqrt(X) - квадратный корень из X.

math.acos(X) - арккосинус X. В радианах.

math.asin(X) - арксинус X. В радианах.

math.atan(X) - арктангенс X. В радианах.

math.atan2(Y, X) - арктангенс Y/X . В радианах. С учетом четверти, в которой находится точка (X, Y).

math.cos(X) - косинус X (X указывается в радианах).

math.sin(X) - синус X (X указывается в радианах).

math.tan(X) - тангенс X (X указывается в радианах).

math.hypot(X, Y) - вычисляет гипотенузу треугольника с катетами X и Y (**math.sqrt(x * x + y * y)**).

math.degrees(X) - конвертирует радианы в градусы.

math.radians(X) - конвертирует градусы в радианы.

math.cosh(X) - вычисляет гиперболический косинус.
math.sinh(X) - вычисляет гиперболический синус.
math.tanh(X) - вычисляет гиперболический тангенс.
math.acosh(X) - вычисляет обратный гиперболический косинус.
math.asinh(X) - вычисляет обратный гиперболический синус.
math.atanh(X) - вычисляет обратный гиперболический тангенс.
math.erf(X) - функция ошибок.
math.erfc(X) - дополнительная функция ошибок ($1 - \text{math.erf}(X)$).
math.gamma(X) - гамма-функция X.
math.lgamma(X) - натуральный логарифм гамма-функции X.
math.pi - $\pi = 3,1415926...$
math.e - $e = 2,718281...$

Примеры применения модуля math и его функций:

```
# Импорт модуля math
import math

# Дробный номер
number=8.10

# выводим целую часть числа с округлением к большему
print("Верхний предел 8.10 это:",math.ceil(number))

# выводим целую часть числа с округлением к меньшему
print("Нижний предел 8.10 это:",math.floor(number))
```

Вывод:

```
Верхний предел 8.10 это: 9
Нижний предел 8.10 это: 8
```

```
# Импорт модуля math
import math

n = -8.10
# Вывод абсолютного значения числа
print(math.fabs(n))
```

Вывод:

```
8.1
```

Пример 2:

```
# Импорт модуля math
import math

number = 5

# вывод факториала числа
print("факториала числа", math.factorial(number))
```

Вывод:

факториала числа 120

Модуль *cmath*.

Модуль *cmath* – предоставляет функции для работы с комплексными числами.

Сложные числа хранятся внутри с использованием прямоугольных или декартовых координат.

При работе с комплексными числами модуль *cmath* может оказать большую помощь. Модуль комплексного числа может быть рассчитан с использованием встроенной функции `abs()`, и его фаза может быть рассчитана с использованием функции `phase(z)`, доступной в модуле *cmath*. Вы можете преобразовать комплексное число в прямоугольной форме в полярную форму, используя `polar(z)`, которая вернет пару `(r, phi)`, где $r = \text{abs}(z)$, а ϕ - `phase(z)`.

Аналогично, вы можете преобразовать комплексное число в полярной форме в прямоугольную форму с помощью `rect(r, phi)`. Комплексное число, возвращаемое этой функцией, равно `r * (math.cos (phi) + math.sin (phi) * 1j)`.

Модуль *cmath* также позволяет использовать регулярные математические функции со сложными числами. Например, вы можете вычислить квадратный корень из комплексного числа, используя `sqrt(z)` или его косинус, используя `cos(z)`.

Комплексные числа имеют множество приложений, таких как моделирование электрических цепей, динамика жидкости и анализ сигналов. Если вам нужно работать над любой из этих вещей, модуль *cmath* не разочарует вас.

Список функций модуля `cmath`.

`cmath.phase(x)` - возвращает фазу комплексного числа (её ещё называют аргументом). Эквивалентно `math.atan2(x.imag, x.real)`. Результат лежит в промежутке $[-\pi, \pi]$.

Получить модуль комплексного числа можно с помощью встроенной функции `abs()`.

`cmath.polar(x)` - преобразование к полярным координатам. Возвращает пару (r, phi) .

`cmath.rect(r, phi)` - преобразование из полярных координат.

`cmath.exp(x)` - e^x .

`cmath.log(x[, base])` - логарифм x по основанию `base`. Если `base` не указан, возвращается натуральный логарифм.

`cmath.log10(x)` - десятичный логарифм.

`cmath.sqrt(x)` - квадратный корень из x .

`cmath.acos(x)` - арккосинус x .

`cmath.asin(x)` - арксинус x .

`cmath.atan(x)` - арктангенс x .

`cmath.cos(x)` - косинус x .

`cmath.sin(x)` - синус x .

`cmath.tan(x)` - тангенс x .

`cmath.acosh(x)` - гиперболический арккосинус x .

`cmath.asinh(x)` - гиперболический арксинус x .

`cmath.atanh(x)` - гиперболический арктангенс x .

`cmath.cosh(x)` - гиперболический косинус x .

`cmath.sinh(x)` - гиперболический синус x .

`cmath.tanh(x)` - гиперболический тангенс x .

`cmath.isfinite(x)` - True, если действительная и мнимая части конечны.

`cmath.isinf(x)` - True, если либо действительная, либо мнимая часть бесконечна.

`cmath.isnan(x)` - True, если либо действительная, либо мнимая часть NaN.

`cmath.pi` - π .

`cmath.e` - e .

Примеры применения модуля cmath и его функций:

```
import cmath

print(cmath.polar(complex(1.0, 1.0)))
print(cmath.phase(complex(1.0, 1.0)))
print(abs(complex(1.0, 1.0)))
```

Вывод:

```
(1.4142135623730951, 0.7853981633974483)
0.7853981633974483
1.4142135623730951
```

```
import cmath

print(cmath.sqrt(complex(25.0, 25.0)))
print(cmath.cos(complex(25.0, 25.0)))
```

Вывод:

```
(5.49342056733905+2.2754493028111367j)
(35685729345.58163+4764987221.458499j)
```

7.1. Техника работы со строками. М.

Задача №1. С клавиатуры вводятся строки, последовательность заканчивается точкой.

Выведите буквы введенных слов в верхнем регистре, разделяя их пробелами.

```
b = []
a = input()
while a != '.':
    b.append(a.upper())
    a = input()
for i in range(len(b)):
    print(' '.join(b[i]))
```

Входные данные:

4
5
6
7
.

Вывод:

4
5
6
7

b делает пустой список, а ждет ввода данных, вайл а делает так, что мы будем добавлять слова в конец списка и их буквы будут увеличиваться пока не будет точки, фор делает так, что он проходит столько, сколько нам нужно слов(это 3 цикла) и делает пробелы между буквами в словах.

Задача №2.

Известно, что для логина часто не разрешается использовать строки содержащие пробелы.

Но пользователю нашего сервиса особенно понравилась какая-то строка.

Замените пробелы в строке на символы нижнего подчеркивания, чтобы строка могла сгодиться для логина. Если строка состоит из одного слова, менять ничего не нужно.

```
s = input()
s = s.replace(' ', '_')
print(s)
```

Входные данные:

Здравствуйте, Леонид Борисович!

Вывод:

Здравствуйте, _Леонид_Борисович!

s принимает на ввод данные, реплейс заменяет 1 заданное значение на другое, принт выводит данные

Задача 3.

Уберите точки из введенного IP-адреса. Выведите сначала четыре числа через пробел, а затем сумму получившихся чисел.

```
s = input()
s = s.replace('.', ' ')
print(s)
s = s.split()
summ = 0
for x in s:
    x = int(x)
    summ += x
print(summ)
```

Входные данные:

192.168.0.1

Вывод:

192 168 0 1

361

s принимает на ввод данные, реплейс заменяет 1 заданное значение на другое, принт выводит данные s, после применяем split, это еще способ разбить строки на части, фор делает так, чтобы посчитал все целочисленные значение потом выводит еще одни данные, который подсчитывает сумму этих чисел.

Задача №4.

Программист логирует программу, чтобы хорошо знать, как она себя ведет (эта весьма распространенная и важная практика).

Он использует разные типы сообщений для вывода ошибок (error), предупреждений (warning), информации (info) или подробного описания (verbose).

Сообщения отличаются по внешнему виду. Назовем модификаторами такие символы,

которые отличают сообщения друг от друга, позволяя программисту понять, к какому

из типов относится сообщения. Модификаторы состоят из двух одинаковых символов

и записываются по разу в начале и в конце строки.

@@ обозначает ошибку

!! обозначает предупреждение

// обозначает информационное сообщение

** обозначает подробное сообщение

Напишите программу, которая принимает строки до точки и выводит, какого типа это сообщение. Если сообщение не содержит модификаторов, проигнорируйте его.

Sample Input:

```
!! cannot resolve this method !!
```

```
@@ invalid type @@
```

```
@@ StackOverflowException @@
```

```
// here I change the variables name //
```

```
** this class is used for operating with the database, including CRUD operations and  
registering new users **
```

```
error on line 42
```

```
// TODO: optimize recursive calls //
```

```
.
```

Sample Output:

предупреждение

ошибка

ошибка

информация

подробное сообщение

информация

```
a = []  
b = input()  
while b != '.':  
    a.append(b)  
    b = input()  
for i in a:  
    if i[:2] == '!!':  
        print('предупреждение')
```

```

if i[:2] == '@@':
print('ошибка')
if i[:2] == '//':
print('информация')
if i[:2] == '**':
print('подробное сообщение')

```

а создает пустой список, б вводит данные, вайл вводит слова в конец строки до точки, фор печатает столько, сколько у нас всего строчек(без точки), ифы смотрят начало строк, если там есть заданные символы, то печатают те значение, которые мы вписали

7.2. Техника работы со строками. М.

П.01 Учебная практика по модулю ПМ.01

Консультация 7. Техника работы со строками

7-2.

Составитель: Гусятинер Л.Б., 27.11.2020, МГОТУ ККМТ, П1-18, П2-18

Задание 1. Подготовить сравнительную инструкцию по использованию форматирования строк



```

Python 3.8.0 Shell
File Edit Shell Debug Options Window Help
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 'hello {}'.format('world!')
'hello world!'
>>> '{0},{1},{2}'.format('a','b','c')
'a,b,c'
>>> '{0}{1}{0}'.format('man', 'can')
'mancanman'
>>> "int: {0:#d}; hex: {0:#x}; oct: {0:#o}; bin: {0:#b}".format(12)
'int: 12; hex: 0xc; oct: 0o14; bin: 0b1100'
>>> |

```

Рис.16 Техника работы со строками

8.1. Техника работы со списками II.

Задача №1 «Больше своих соседей»

Дан список чисел. Определите, сколько в этом списке элементов, которые больше двух своих соседей, и выведите количество таких элементов. Крайние элементы списка никогда не учитываются, поскольку у них недостаточно соседей.

```
l = input().split(' ')
c = 0

for i in range(1, len(l)-1):
    if l[i-1] <= l[i] >= l[i+1]:
        c += 1

print(c)
```

Входные данные:

1 2 1 3 4

Вывод:

1

Задача №2 «Количество совпадающих пар»

Дан список чисел. Посчитайте, сколько в нем пар элементов, равных друг другу. Считается, что любые два элемента, равные друг другу образуют одну пару, которую необходимо посчитать.

```
l = input().split(' ')
c = 0

for i in range(0, len(l)):
    for j in range(0, len(l)):
        if (l[i] == l[j]) and (i != j):
            c += 1

print(c // 2)
```

Входные данные:

1 1 1 1 1

Вывод:

10

Задача №3 (Л.Б.)

Дано N списков целых чисел (N вводится с клавиатуры, сами списки заполняются

случайным образом). Требуется сформировать

- список, содержащий уникальные значения, попадающие в каждый из N списков

- список, содержащий уникальные значения, попадающие хотя бы в один из N списков

Решение без использования set - дополнительный бонус

8.2. Техника работы со списками. II.

Задача №1 Array112.

Дан массив A размера N.

Упорядочить его по возрастанию методом сортировки простым обменом («пузырьковой» сортировкой):

просматривать массив, сравнивая его соседние элементы

(A0 и A1, A1 и A2 и т. д.) и меняя их местами,

если левый элемент пары больше правого; повторить описанные действия N - 1 раз. Для контроля за выполняемыми действиями выводить содержимое массива после каждого просмотра.

Учесть, что при каждом просмотре количество анализируемых пар можно уменьшить на 1.

```
import random

c = random.randrange(2,20)
b = [random.randrange(0,10) for i in range(c)]

print(b)

for i in range(1,c) :
    for j in range(0,c-i) :
        if b[j] > b[j+1] :
            b[j], b[j+1] = b[j+1], b[j]

print(b)
```

Вывод:

```
[0, 3, 3, 8, 9, 5]
[0, 3, 3, 5, 8, 9]
```

Задача №2 Array113.

Дан массив A размера N.

Упорядочить его по возрастанию методом сортировки простым выбором: найти максимальный элемент массива и поменять его местами с последним (N-1 м) элементом; выполнить описанные действия N - 1 раз, каждый раз уменьшая на 1 количество анализируемых элементов и выводя содержимое массива.

```
import random
```

```

c = random.randrange(2,10)
b = [random.randrange(0,10) for i in range(c)]

print(b)

for i in range(0,c) :
    max_val = max(b[:c-i])
    max_idx = b.index(max_val)
    b[max_idx], b[c-i-1] = b[c-i-1], b[max_idx]
    print(b)

```

Вывод:

```

[2, 4, 8]
[2, 4, 8]
[2, 4, 8]
[2, 4, 8]

```

Задача №3 Array114.

Дан массив A размера N. Упорядочить его по возрастанию методом сортировки простыми вставками: сравнить элементы A0 и A1 и, при необходимости меняя их местами, добиться того, чтобы они оказались упорядоченными по возрастанию; затем обратиться к элементу A2 и переместить его в левую (уже упорядоченную) часть массива, сохранив ее упорядоченность; повторить этот процесс для остальных элементов, выводя содержимое массива после обработки каждого элемента (от 1-го до N-1 го).

```

import random

c = random.randrange(2,10)
b = [random.randrange(0,10) for i in range(c)]

print(b)

for i in range(1,c) :
    a = False
    x = b[i]
    q = i - 1
    while q >= 0 :
        if x >= b[q] :
            break
        else :
            b[q+1] = b[q]
            a = True
        q -= 1
    if a and q >= -1:
        b[q+1] = x
    print(b)

```


Вывод:

```
[3, 9, 9, 6]
[3, 9, 9, 6]
[3, 9, 9, 6]
[3, 6, 9, 9]
```

9.1. Техника работы с циклом for и генераторами списков. И.

Задание 1. (Л.Б.) Для проведения конкурса проектов в ККМТ формируются группы

из 4х участников: coder, writer, tester, designer, программирующих на одном и том же языке.

Каждый студент может программировать только на одном языке и занимать только одну позицию.

Дан текстовый файл, содержащий перечень студентов с указанием языка и позиции

(каждый студент с новой строки)

Требуется

1. Получить список студентов с указанием языка и позиции
2. Сформировать список всевозможных команд
3. Вывести список команд с указанием состава и названия команды:

Команда 1

coder: ...

designer: ...

tester: ...

writer: ...

Команда 2

...

Пункты 1 и 2 выполнить с использованием генераторов списка

#№1

```
file = open("file.txt", "r")
```

```
def prof(mtrx,mas):
    if (len(mtrx) == 0):
        return True
    count = 0
    for i in range(len(mtrx)):
        if(mtrx[i][1] != mas[1]):
            count += 1
        if(mtrx[i][2] == mas[2]):
            count += 1
    if(count == len(mtrx)*2):
        return True
    return False
```

```
list = file.readlines()
```

```

out = [i.strip().split() for i in list]
for i in range(len(out)):
    print(f'{out[i][1]}: {out[i][0]} | {out[i][2]}')
print()
print(out)
print('Всего участников - ', len(out), '\n')

```

9.2. Техника работы с циклом for и генераторами списков. И.

Задача №1. Array55.

Дан целочисленный массив A размера N (≤ 15). Переписать в новый целочисленный массив B все элементы с нечетными порядковыми номерами (1, 3, ...) и вывести размер полученного массива B и его содержимое. Условный оператор не использовать.

```

import random

c = random.randrange(2,15)
b = [random.randrange(1,11) for i in range(c)]
a = b[1::2]

print(b)
print(len(a))
print(a)

```

Вывод:

```

[9, 1, 1, 6, 7, 4, 6, 4, 5, 5, 7, 9, 9, 3, 10, 9, 9]
17
[9, 10, 11, 17, 24, 28, 34, 38, 43, 48, 55, 64, 73, 76, 86, 95,
104]

```

Задача №2. Array57.

Дан целочисленный массив A размера N. Переписать в новый целочисленный массив B того же размера вначале все элементы исходного массива с четными номерами, а затем — с нечетными:
A[0], A[2], A[4], A[6], ..., A[1], A[3], A[5],
Условный оператор не использовать.

```

import random

c = random.randrange(2,21)
b = [random.randrange(1,11) for i in range(c)]
a = b[0::2] + b[1::2]

print(b)
print(len(a))
print(a)

```

Входные данные:

5
6

Вывод:

9 8 7 9 3 3
9 2 8 3 5 2
6 3 9 9 4 3
9 8 8 7 8 3
7 5 5 4 7 4

Задача №3. Array58.

Дан массив A размера N. Сформировать новый массив B того же размера по следующему правилу: элемент B[K] равен сумме элементов массива A с номерами от 0 до K.

```
import random

c = random.randrange(2,21)
b = [random.randrange(1,11) for i in range(c)]
a = []
a.append(b[0])

print(b)

for i in range(1,c) :
    a.append(b[i] + a[i-1])

print(len(a))
print(a)
```

Вывод:

[9, 1, 1, 6, 7, 4, 6, 4, 5, 5, 7, 9, 9, 3, 10, 9, 9]
17
[9, 10, 11, 17, 24, 28, 34, 38, 43, 48, 55, 64, 73, 76, 86, 95, 104]

Задача №4. Matrix3.

Даны целые положительные числа M, N и набор из M чисел. Сформировать матрицу размера M x N, у которой в каждом столбце содержатся все числа из исходного набора (в том же порядке).

```
import random

R = int(input())
C = int(input())

matrix = []
```

```

for i in range(R):
    c = []
    for j in range(C):
        c.append(random.randrange(2,10))
    matrix.append(c)

for i in range(R):
    for j in range(C):
        print(matrix[i][j], end = " ")
    print()

```

Входные данные:

5
6

Вывод:

```

9 8 7 9 3 3
9 2 8 3 5 2
6 3 9 9 4 3
9 8 8 7 8 3
7 5 5 4 7 4

```

Задача 5. Matrix56.

Дана матрица размера $M \times N$ (N — четное число). Поменять местами левую и правую половины матрицы.

```

from random import randint as rnd
n,m = int(input('n:\n')),int(input('m:\n'))
upr = []
botl = []
a = [[rnd(1,10) for _ in range(n)] for _ in range(m)]
print(*a, '\n', sep='\n')
for i in range(int(m/2)):
    upr.append(a[i][int(n/2):])
for i in range(int(m/2),m):
    botl.append(a[i][:int(n/2)])
for i in range(int(m/2)):
    a[i][int(n/2):] = botl[i]
for i in range(int(m/2),m):
    a[i][:int(n/2)] = upr[i]
print(*a, sep='\n')

```

Входные данные:

n:
2
m:
2

Вывод:

```

[1, 4]
[10, 8]

```

10.1. Техника работы с функциями. М.

Задача №2. Func6.

Описать функцию SumRange(A, B) целого типа, находящую сумму всех целых чисел от A до B включительно (A и B — целые). Если $A > B$, то функция возвращает 0.

С помощью этой функции найти суммы чисел от A до B и от B до C, если даны числа A, B, C.

```
def SumRange(a, b):
    s = 0
    i = a
    if a > b:
        return 0
    else:
        for i in range(b):
            s += i
            print(s)
    return s
a = int(input())
b = int(input())
print(SumRange(a, b))
```

Входные данные:

3
4

Вывод:

0

10.2. Техника работы с функциями. М.

Задача №3.

Array55. Дан целочисленный массив A размера N (≤ 15). Переписать в новый целочисленный массив B все элементы с нечетными порядковыми номерами (1, 3, ...) и вывести размер полученного массива B и его содержимое. Условный оператор не использовать.

```
import random

N = random.randrange(2, 15)
a = [random.randrange(1, 11) for i in range(N)]
b = a[1::2]

print(a)
print(b)
```

Вывод:

[6, 10, 9, 3, 8, 1, 2, 3, 5, 5, 1, 2, 10, 1]
[10, 3, 1, 3, 5, 2, 1]

11.1. Техника работы со словарями. И.

Задача №1.

Задача «Номер появления слова»

Условие. В единственной строке записан текст. Для каждого слова из данного текста

подсчитайте, сколько раз оно встречалось в этом тексте ранее.

Словом считается последовательность непробельных символов идущих подряд, слова разделены

одним или большим числом пробелов или символами конца строки.

```
a = {}
for i in input().split():
    a[i] = a.get(i, 0) + 1
    print(a[i] - 1, end=' ')
```

Входные данные:

one two one tho three

Вывод:

0 0 1 0 0

11.2. Техника работы со словарями. М

Задача №1. Телефонная книга.

Этап 1. Коля устал запоминать телефонные номера и заказал у Вас

программу, которая заменила бы ему телефонную книгу. Коля может послать программе

два вида запросов: строку, содержащую имя контакта и его номер, разделенные пробелом,

или просто имя контакта. В первом случае программа должна добавить в книгу новый номер,

во втором – вывести номер контакта. Ввод происходит до символа точки. Если введенное

имя уже содержится в списке контактов, необходимо перезаписать номер.

```
a = {}
while True:
    s = input().split()
    if s[0] == '.':
        break
    if len(s) == 2:
        a[s[0]] = s[1]
    if len(s) == 1 :
        print(a[s[0]])
```

Входные данные:

```
Ben 89001234050
Alice 210-220
Alice
Alice 404-502
Nick +16507811251
Ben
Alex +4(908)273-22-42
Alice
Nick
Robert 51234047129
Alex
```

Вывод:

```
210-220
89001234050
404-502
+16507811251
+4(908)273-22-42
```

12.1. Техника работы с множествами. М.

Задача №1.

Задача «Количество различных чисел»

Условие. Дан список чисел. Определите, сколько в нем встречается различных чисел.

```
print(len(set(input().split())))
```

Входные данные:

```
1 2 3 2 1
```

Вывод:

```
3
```

12.2. Техника работы с множествами. И.

Задача №1.

Простейшая система проверки орфографии может быть основана на использовании списка известных слов.

Если введённое слово не найдено в этом списке, оно помечается как "ошибка".

Попробуем написать подобную систему.

На вход программе первой строкой передаётся количество d известных нам слов, после чего

на d строках указываются эти слова.

Затем передаётся количество l строк текста для проверки, после чего l строк текста.

Выведите уникальные "ошибки" в произвольном порядке. Работу производите без учёта регистра.

```
a = int(input())
```

```

b = []
for i in range(a):
    b.append(input().lower())
c = set(b)
d = int(input())
e = []
for j in range(d):
    e.append(input().split())
m = set()
for row in e:
    s = set(row)
    for v in s:
        m.add(v)
f = []
for k in m:
    if k in c:
        continue
    else:
        f.append(k)
for o in f:
    print(o)

```

Входные данные:

```

4
champions
we
are
Stepik
3
We are the champignons
We Are The Champions
Stepic

```

Вывод:

```

Stepic
Are
Champions
the
The
We
Champignons

```

Почти правильно, но выводит лишнее.

13.1. Техника работы с кортежами II.

Задача №2. Убывающий ряд.

С клавиатуры вводятся целые числа $a > b$. Выведите убывающую последовательность чисел по одному числу в строке.

```
a=int(input())
b=int(input())
for i in range(b, a)[::-1]:
    print(i+1)
```

Входные данные:

```
3
-2
```

Вывод:

```
3
2
1
0
-1
```

13.2. Техника работы с кортежами. II.

Задача №1. Класс `namedtuple()` модуля `collections` в Python.

По приведённым примерам подготовить свои.

Кортежи в питоне представляют собой простую структуру данных для группировки произвольных объектов. Кортежи являются неизменяемыми — они не могут быть изменены после их создания.

Пример 1

```
>>> tup = ('hello', object(), 42)
>>> tup ('hello', <object object at 0x105e76b70>, 42)
>>> tup[2] 42
>>> tup[2] = 23 TypeError:
"'tuple' object does not support item assignment"
```

Обратная сторона кортежей — это то, что мы можем получать данные из них используя только числовые индексы. Вы не можете дать имена отдельным элементам сохранённым в кортеже. Это может повлиять на читаемость кода.

Пример 2

```
#Вот как выглядит именованный кортеж:
from collections import namedtuple
Car = namedtuple('Car' , 'color mileage')
```

Пример 3

```
c = MyCarWithMethods('red', 1234)
c.hexcolor()
```

Вывод:

```
'#ff0000'
```

14.1. Техника работы с файлами. II.

Задача №1. Text5.

Дана строка S и текстовый файл. Добавить строку S в конец файла.

```
with open('in.txt', 'a') as f_in:  
    f_in.write(input())
```

Входные данные:

Здравствуйте, Леонид Борисович

Вывод:

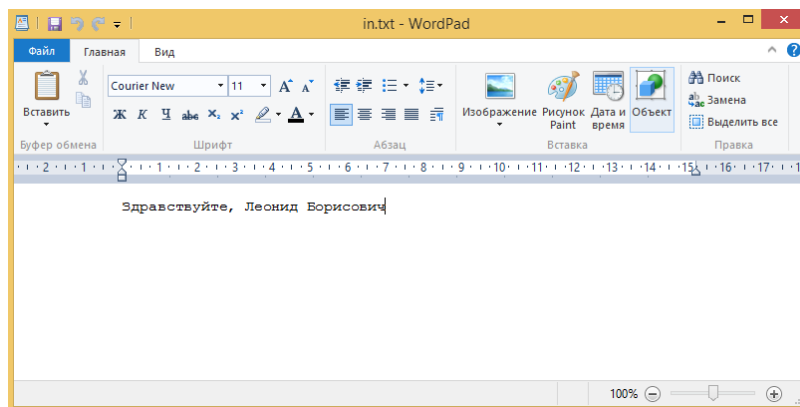


Рис.17 демонстрация работы программы

Входные данные:

!!!!!!!!!!

Вывод:

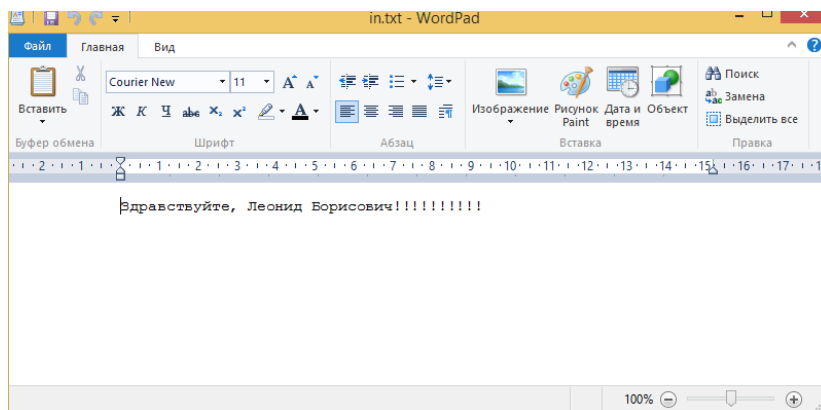


Рис. 18 Демонстрация работы программы

15.1. Техника работы с модулями. М

Задача №1. Контейнерные типы данных модуля collections.

```
from collections import deque
a = deque([1,2,3])
a.appendleft(0)
print(a)
a.popleft()
print(a)
print(a[0])
a.popleft()
print(a)
a.append(10)
print(a)
```

Вывод:

```
deque([0, 1, 2, 3])
deque([1, 2, 3])
1
deque([2, 3])
deque([2, 3, 10])
```

15.2. Техника работы с модулями. М.

Задача №1. Контейнерные типы данных модуля collections.

```
from collections import defaultdict
a = defaultdict(set, **{"day":{"night"}, "night": {"month"}})
a["month"].add("day")
a["year"].add("day")
print(a)
```

Вывод:

```
defaultdict(<class 'set'>, {'day': {'night'}, 'night': {'month'},
'month': {'day'}, 'year': {'day'}})
```

15.3. Техника работы с модулями. М.

Задача №1. Функция `argv` модуля `sys` в Python.

```
import sys
a = sys.stdin.readline()
sys.stdout.write(a + '\n')
sys.stdout.write(sys.version + '\n')
sys.stdout.write(str('Test') + '\n')
sys.exit()
```

Входные данные:

1234

Вывод:

1234

3.9.1 (tags/v3.9.1:1e5d33e, Dec 7 2020, 17:08:21) [MSC v.1927 64
bit (AMD64)]
Test

15.4. Техника работы с модулями. М.

- Задание 1.** Вывод текущей директории
- Задание 2.** Создание папки
- Задание 3.** Изменение директории
- Задание 4.** Создание вложенных папок
- Задание 5.** Создание файлов
- Задание 6.** Переименование файлов
- Задание 7.** Перемещение файлов
- Задание 8.** Список файлов и директорий
- Задание 9.** Удаление файлов
- Задание 10.** Удаление директорий
- Задание 11.** Получение информации о файлах

```
import os
print("Текущая директория:", os.getcwd())
os.mkdir("prac")
os.makedirs("prac/task/os")
text1 = open("prac/task/П1.txt", "w")
os.rmdir("prac/task/os")
print(os.stat("prac/task/П1.txt"))
print("Размер файла:", os.stat("prac/task/П1.txt").st_size)
```

Вывод:

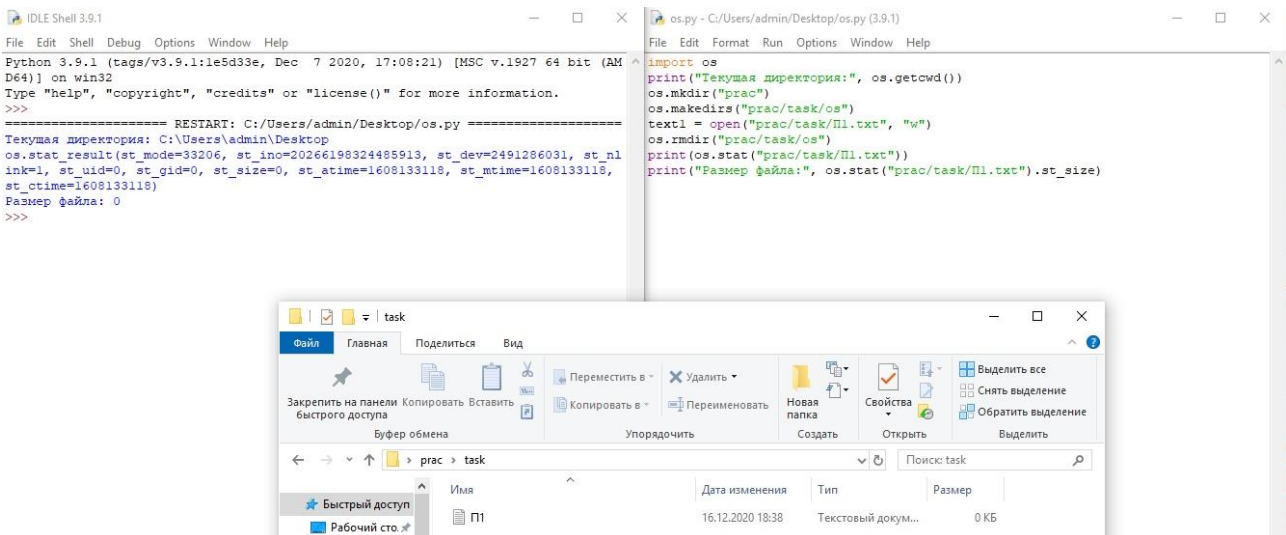


Рис.19 Демонстрация работы программы

16.1. Техника работы с классами. II.

Задача №1. Создание класса

```
class Employee:
    Emp_sis = 0
```

16.2. Техника работы с классами. II.

Задание 1. Создание классов

Задание 2. Создание экземпляров класса

Задание 3. Доступ к атрибутам

Задание 4. Встроенные атрибуты класса

Задание 5. Уничтожение объектов (сбор мусора)

```
class Virt_cookies ( object ):
    ''' Виртуальные печеньки '''
    def __init__ (self,name):
        print ( 'Появилась новая печенька.' )
        self.name=name
    def __str__ (self):
        return 'Класс - Virt_cookies, сорт печеньки - '
        +self.name
    def __del__ (self):
        print( 'Печенька съедена.' )
```

С помощью метода `__init__()`, который вызывается при каждом создании экземпляра класса (для `Virt_cookies` это будет виртуальная печенье), на экран выводится сообщение об этом и выполняется инициализация атрибута `name`, указывающего сорт этой печенье. Отметим, что первым параметром каждого метода в объявлении класса, за исключением статических методов (см. подраздел 2.2), является слово `self`, которое при вызовах этих методов не указывается. Метод `__str__` вызывается при использовании функции `print(obj)` и возвращает данные об объекте `obj` – в данном случае это имя класса и сорт созданной печенье. С помощью дескриптора класса (метода `__del__`) осуществляется вывод на экран сообщения "Печенька съедена " при удалении объекта класса `Virt_cookies`.

16.3. Техника работы с классами. II.

Задание 1. Наследование класса

Задание 2. Переопределение методов

Задание 3. Популярные базовые методы

Задание 4. Приватные методы и атрибуты класса

#Для начала создадим вспомогательный класс Point для хранения координат на плоскости:

```
class Point:
    def __init__(self, x = 0, y = 0):
        self.x = x
        self.y = y
```

#И после него объявим класс для работы с графическим примитивом линией:

```
class Line:
    def __init__(self, sp:Point, ep:Point, color:str = "red",
width:int = 1):
        self._sp = sp
        self._ep = ep
        self._color = color
        self._width = width

    def drawLine(self):
        print(f"Рисование линии: {self._sp}, {self._ep},
{self._color}, {self._width}")
```

16.4. Техника работы с классами. II.

Задание 1. Придумать собственный класс

Задание 2. Неформально описать функционал класса

Задание 3. Реализовать класс в модуле

Задание 4. Разработать скрипт для демонстрации работы с классом
(импортировать модуль,
создать экземпляры, вызвать методы)

```
class Student:
    def __init__(self):
        self.__university = 'RTU MIREA'
        self.__name = ''
        self.__surname = ''
        self.__marks = list()
    def set_name(self, name: str):
        self.__name = name
    def set_surname(self, surname: str):
        self.__surname = surname
    def set_marks(self, marks: list):
        self.__marks = marks
    def set_university(self, university: str):
        self.__university = university
    def get(self):
        return self.__name, self.__surname, self.__university,
self.__marks
```


Список литературы:

1. <https://pythonworld.ru/moduli/modul-math.html>
2. <https://pythonru.com/osnovy/modul-math-python>
3. <https://pythonworld.ru/moduli/modul-cmath.html>
4. <https://python-scripts.com/math>
5. <https://pythonru.com/osnovy/rabota-s-fajlami-v-python-s-pomoshhu-modulja-os>
6. <https://stepik.org/course/31182/syllabus>
7. <https://pythontutor.ru/>