



Государственное бюджетное образовательное учреждение высшего образования
Московской области

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ
имени дважды Героя Советского Союза, летчика-космонавта А.А. Леонова

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

Выполнил студент:

Батраков Д. С.

_____ (подпись)

Гусятинер Л. Б.

_____ (подпись)

_____ (оценка)

Содержание отчёта

Раздел 1. Техника решения задач с использованием структурного и объектно-ориентированного программирования.	3
1.1 Установка интерпретатора Python 3 и настройка окружения	3
1.2 Техника работы в командной строке и среде IDLE.....	6
1.3 Техника работы с линейными и разветвляющимися программами	7
1.4 Техника работы с циклическими программами, цикл while	10
1.5 Техника работы с числами.....	13
1.6 Техника работы со строками	16
1.7 Техника работы со списками	18
1.8 Техника работы с циклом for и генераторами списков	20
1.9 Техника работы с функциями	22
1.10 Техника работы со словарями.....	24

Раздел 1. Техника решения задач с использованием структурного и объектно-ориентированного программирования.

1.1 Установка интерпретатора Python 3 и настройка окружения

Для установки интерпретатора Python на компьютер, вам нужно скачать дистрибутив. Загрузить его последнюю версию можно с официального сайта, перейдя по ссылке <https://www.python.org/downloads/>



Рис 1. Официальный сайт Python

Порядок установки на Windows:

1. Запустить скачанный установочный файл.
2. Выбрать способ установки.



Рис 2. Установщик Python

3. Отметить необходимые опции установки (доступно при выборе Customize installation)

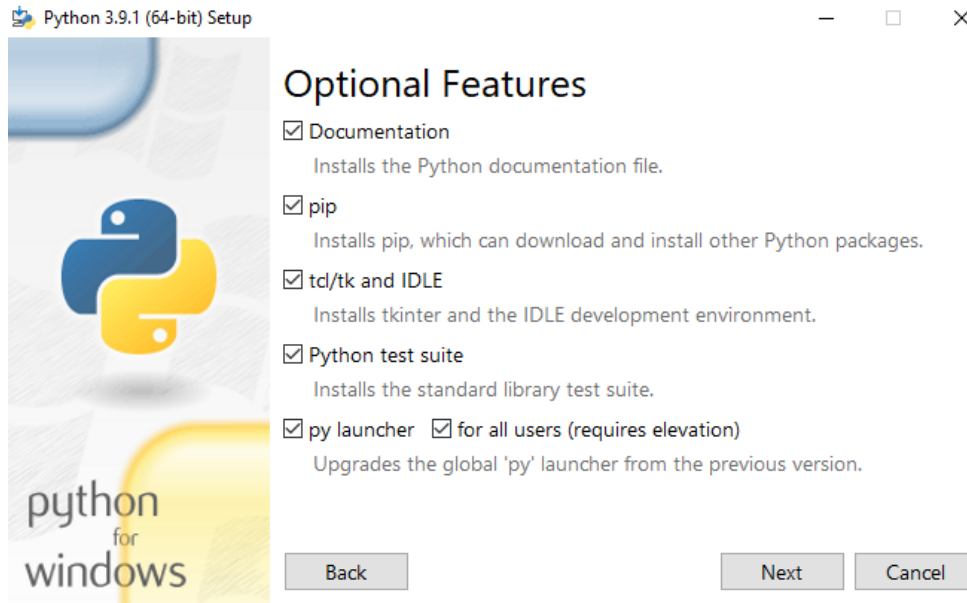


Рис 3. Опции установки

На этом шаге нам предлагается отметить дополнения, устанавливаемые вместе с интерпретатором Python. Выбираю:

- Documentation – установка документаций.
- pip – установка пакетного менеджера.
- tcl/tk and IDLE – установка интегрированной среды разработки и библиотеки.

4. Выбираем место установки (доступно при выборе Customize installation)

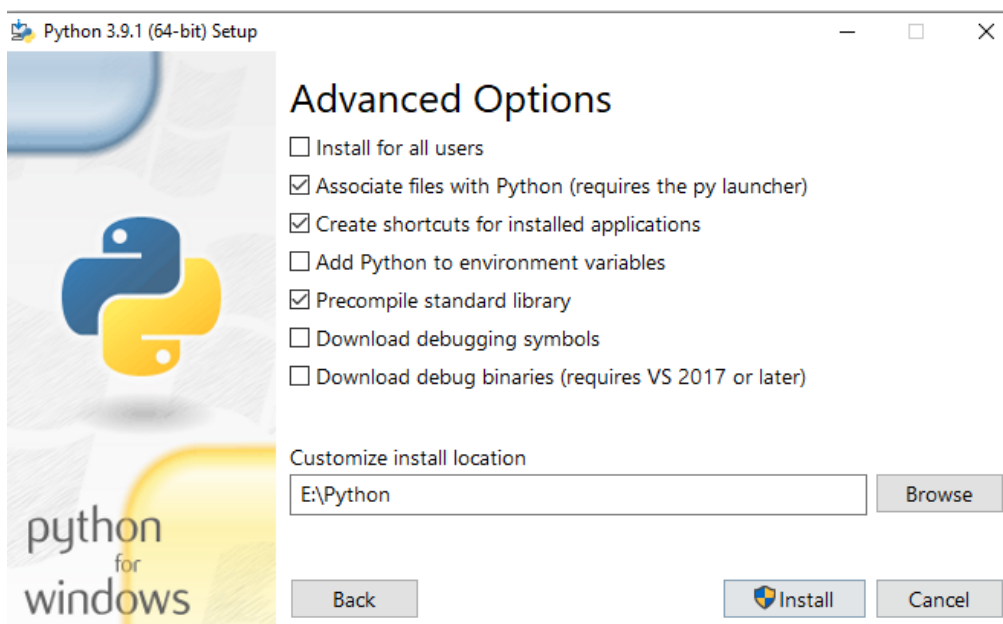


Рис 4. «Продвинутые» опции установки

5. После успешной установки python:

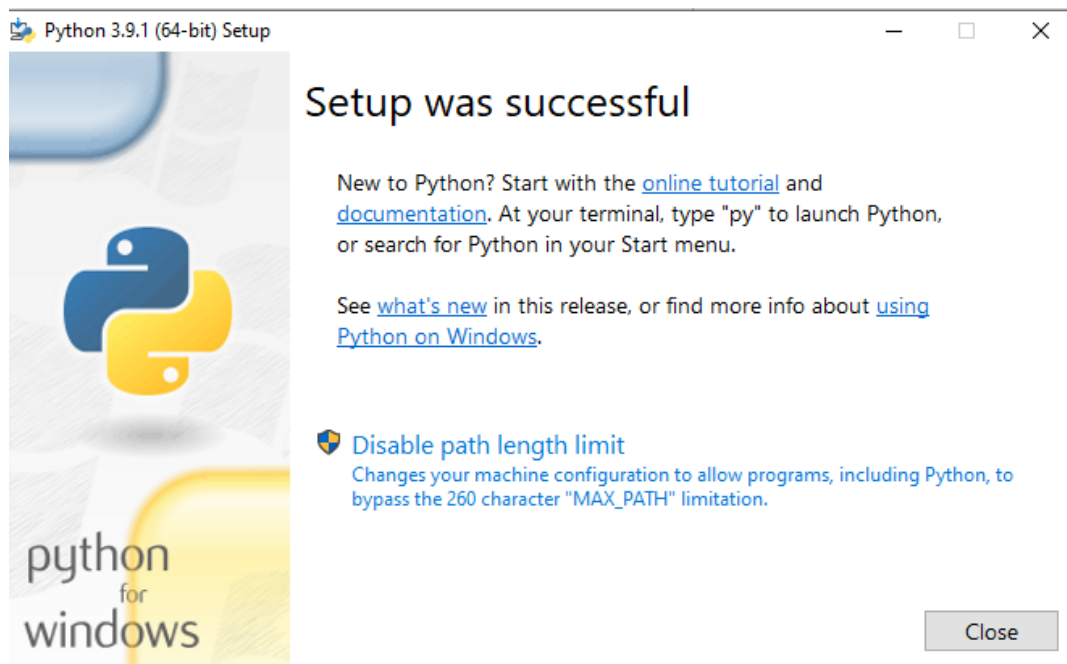
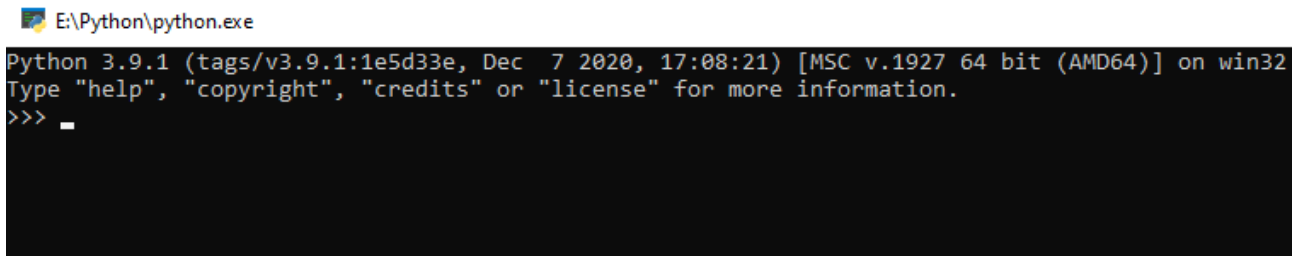


Рис 5. Сообщение об установке python

1.2 Техника работы в командной строке и среде IDLE

Выполняя (запуская) команду “python” в вашем терминале, вы получаете интерактивную оболочку Python.



```
E:\Python\python.exe
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.
>>> _
```

Рис 6. Интерактивная оболочка Python

Существует несколько способов закрыть оболочку Python: `>>> exit()` или же `>>> quit()`

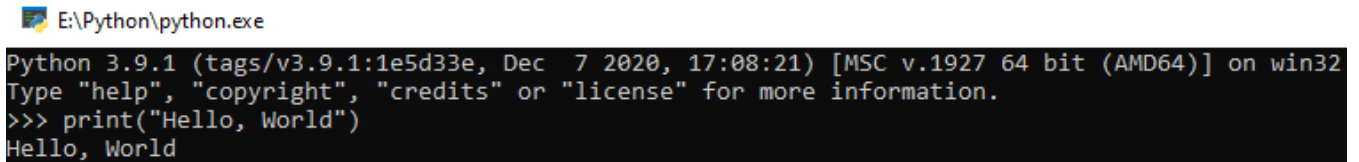
Теперь напишите в интерактивной оболочке следующий код:

```
>>> print("Hello, World ")
```

Нажмите **Enter** на вашей клавиатуре.

```
>>> print("Hello, World")
```

```
Hello, World
```



```
E:\Python\python.exe
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
```

Рис 7. Первая программа

1.3 Техника работы с линейными и разветвляющимися программами

Листинг 1. K4_1

```
'''
- input
Функция input() в Python, ввод данных с клавиатура.
https://docs-python.ru/tutorial/vstroennye-funktsii-interpretatora-python/funktsija-input/
- print
Функция print() в Python, печатает объект.
https://docs-python.ru/tutorial/vstroennye-funktsii-interpretatora-python/funktsija-print/
- форматная строка и метод формат
'''

print('Введите имя: ')
a = (input())
print('Hi, {0}, {1}!'.format(a, input()))
```

Листинг 2. K4_2_1

```
'''
K4_2. Техника работы с разветвляющимися программами
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):
    day_list = ['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 четвертое', 'двадцать пятое', 'двадцать
                шестое',
                'двадцать седьмое', 'двадцать восьмое', 'двадцать
                девятое',
                'тридцатое', 'тридцать первое']
    month_list = ['января', 'февраля', 'марта', 'апреля', 'мая',
                  'июня', 'июля', 'августа', 'сентября', 'октября', 'ноября',
                  'декабря']
```

```

#единицы
    year_list_unit = ['', 'первого', 'второго', 'третьего',
'четвёртого', 'пятого', 'шестого', 'седьмого', 'восьмого',
'девятого']
    year_list_unit2 = ['', 'одиннадцатого', 'двенадцатого',
'тринадцатого', 'четырнадцатого', 'пятнадцатого',
'шестнадцатого', 'семнадцатого',
'восемнадцатого', 'девятнадцатого']
#десятки
    year_list_dec =
['', '', 'двадцать', 'тридцать', 'сорок', 'пятьдесят', 'шестьдесят', 'сем
ьдесят', 'восемьдесят', 'девяносто']
    year_list_dec2 = ['', 'десятого', 'двадцатого', 'тридцатого',
'сорокового', 'пятидесятого', 'шестидесятого', 'семидесятого',
'восемидесятого',
'девяностого']
#сотни
    year_list_hun =
['', 'сто', 'двести', 'триста', 'четыреста', 'пятьсот', 'шестьсот', 'семь
сот', 'восемьсот', 'девятьсот']
    year_list_hun2 =
['', 'сотого', 'двухсотого', 'трёхсотого', 'четырёхсотого', 'пятьсотого',
'шестьсотого', 'семьсотого', 'восемисотого', 'девятьсотого']
#тысячи
    year_list_th = ['', 'тысяча', 'две тысячи', 'три тысячи']
    year_list_th2 = ['', 'тысячного', 'двух тысячного', 'трёх
тысячного']

    date_list = date.split('.')
    date_y = [int(date_list[2])//1000,
int(date_list[2])%1000//100,
int(date_list[2])%100//10, int(date_list[2])%10]

    if(date_y[2] == 1):
        return (day_list[int(date_list[0]) - 1] + ' ' +
month_list[int(date_list[1]) - 1] + ' ' +
year_list_th[date_y[0]] + ' ' +
year_list_hun[date_y[1]-1] + ' ' + year_list_unit2[date_y[3]] + '
' + 'года')
    elif (date_y[0] == date_y[1] == date_y[2] == date_y[3] == 0):
        return (day_list[int(date_list[0]) - 1] + ' ' +
month_list[int(date_list[1]) - 1] + ' ' +
'нулевого года')
    elif (date_y[1] == date_y[2] == date_y[3] == 0):
        return (day_list[int(date_list[0]) - 1] + ' ' +
month_list[int(date_list[1]) - 1] + ' ' +
year_list_th2[date_y[0]] + ' ' + 'года')
    elif(date_y[2] == date_y[3] == 0):
        return (day_list[int(date_list[0]) - 1] + ' ' +
month_list[int(date_list[1]) - 1] + ' ' +
year_list_th[date_y[0]] + ' ' +
year_list_hun2[date_y[1]] + ' ' + 'года')
    elif(date_y[3] == 0):

```



```

        return (day_list[int(date_list[0]) - 1] + ' ' +
                month_list[int(date_list[1]) - 1] + ' ' +
                year_list_th[date_y[0]] + ' ' +
year_list_hun[date_y[1]] + ' ' + year_list_dec2[date_y[2]] + ' ' +
'года')

```

```

        return (day_list[int(date_list[0]) - 1] + ' ' +
                month_list[int(date_list[1]) - 1] + ' ' +
                year_list_th[date_y[0]] + ' ' + year_list_hun[date_y[1]] +
' ' + year_list_dec[date_y[2]] + ' ' + year_list_unit[date_y[3]] +
' ' + 'года')

```

```

date = input()
while(date != 'stop'):
    print(get_date(date))
    date = input()

```

Листинг 3. K5_1_2

'''

Задание 2.

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

'''

```

while (1):
    print(''(P) -> print \n(R) -> repit \n(B) -> break'')
    i = input()
    if i == 'P':
        print("Hello world")
    elif i == 'R':
        print("Plz repit :)")
        continue
    elif i == 'B':
        break
    else:
        print("Incorrect Data, try again")

```

#or

```

x = input()
i = 0
while x != '.':
    if x == '!':
        print(i)
        break
    i += 1
    x = input()
else:
    print('not found')

```

1.4 Техника работы с циклическими программами, цикл while

Листинг 4. K5_2_1

'''

Задание 1. Вычислить значение $\sin(x)$ с точностью до ϵ при помощи разложения в ряд

'''

```
import math

eps = 1.0
n = 0
while eps + 1 > 1:
    eps /= 2
    n+=1
eps *= 2

x = int(input())
s = 0
for i in range(n):
    s += math.sin(x)
print(s)
```

Листинг 5.K5_2_2

'''

K5_2. Техника работы с циклическими программами _ цикл while;

Задание 2.

<https://stepik.org/lesson/3364/step/11?unit=947>

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.

'''

```
def main():
    a = int(input())
    summ = 0
    while a != 0:
        summ = summ + a
        a = int(input())
    print(summ)

if __name__ == "__main__":
    main()
```

Листинг 6. K5_2_3

'''

K5_2. Техника работы с циклическими программами _ цикл while;

Задание 3.

Разработать программу для нахождения наибольшего общего делителя

'''

```
def nod(a, b):
    assert a >= 0 and b >= 0
    if a == 0 or b == 0:
        return max(a, b)
    return nod(b % a, a)
if __name__ == "__main__":
    a, b = map(int, input().split())
    print(nod(a, b))
```

Листинг 7. K5_2_4

'''

K5_2. Техника работы с циклическими программами _ цикл while;

Задание 4.

С использованием результата задания 2 разработать программу для нахождения наименьшего

общего кратного

'''

```
def main():
    a, b = map(int, input().split())
    c = a * b
    while (n != 0 and c != 0):
        if (a > b):
            a = a % b
        else:
            b = b % a
    print(c // (a + b))

if __name__ == "__main__":
    main()
```

Листинг 8. K5_2_5

'''

K5_2. Техника работы с циклическими программами _ цикл while;

Задание 5.

<https://stepik.org/lesson/3369/step/8?unit=952>

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.

'''

```
def main():
    n = int(input())
    j = num = 1
    for i in range(1, n+1):
        print(num, end=' ')
        if j < num:
            j += 1
        else:
            j = 1
            num += 1
    print("\n")
if __name__ == "__main__":
    main()
```

1.5 Техника работы с числами

Листинг 9. К6_1_1

'''

Задание 1. Составить и выполнить по 3 примера использования модулей для работы с дробными числами (fractions), для точных вычислений (decimal).

'''

```
from decimal import Decimal
from fractions import Fraction

#decimal
print("Функции <<Decimal>>")
number = Decimal("0.444")
print(number)
number = number.quantize(Decimal("1.00"))
print(number)
number = Decimal("0.555678")
print(number.quantize(Decimal("1.00")))
number = Decimal("0.999")
print()

#fraction
print("Функции <<Fraction>>")
print(number.quantize(Decimal("1.00")))
print(Fraction(1, 3))
print(Fraction(3.1415))
a = Fraction(1, 7)
b = Fraction(1, 3)
print(a + b)
```

Листинг 10. К6_2

'''

К6_2. Техника работы с числами cmath

'''

```
import cmath

def main():
    print("Сложные функции")
    print("cmath.polar(complex(1.0, 1.0)) =",
    cmath.polar(complex(1.0, 1.0)))
    print("cmath.phase(complex(1.0, 1.0)) =",
    cmath.phase(complex(1.0, 1.0)))
    print("abs(complex(1.0, 1.0)) =", abs(complex(1.0, 1.0)))
    print("cmath.sqrt(complex(25.0, 25.0)) =",
    cmath.sqrt(complex(25.0, 25.0)))
    print("cmath.cos(complex(25.0, 25.0)) =",
    cmath.cos(complex(25.0, 25.0)))
```

```

print()

if (__name__ == "__main__"):
    main()

```

Листинг 11. К6_2

'''

К6_2. Техника работы с числами math

'''

```

import math

def getsin(x):
    multiplier = 1
    result = 0
    for i in range(1, 20, 2):
        result += multiplier * pow(x, i) / math.factorial(i)
        multiplier *= -1

    return result

def main():
    print("Арифметические функции")
    print("math.pow(3, 2) =", math.pow(3, 2))
    print("math.pow(9, 0.5) =", math.pow(9, 0.5))
    print("math.sqrt(9) =", math.sqrt(9))
    print("math.factorial(5) =", math.factorial(5))
    print("sin(pi/2) =", getsin(math.pi / 2))
    print()

    print("Тригонометрические функции")
    print("math.sin(math.pi/4) =", math.sin(math.pi / 4))
    print("math.cos(math.pi) =", math.cos(math.pi))
    print("math.tan(math.pi/6) =", math.tan(math.pi / 6))
    print("math.hypot(12,5) =", math.hypot(12, 5))
    print("math.atan(0.5773502691896257) =",
math.atan(0.5773502691896257))
    print("math.asin(0.7071067811865476) =",
math.asin(0.7071067811865476))
    print()

    print("Гиперболические функции")
    print("math.asinh(11.548739357257746) =",
math.asinh(11.548739357257746))
    print("math.acosh(11.591953275521519) =",
math.acosh(11.591953275521519))
    print("math.atanh(0.99627207622075) =",
math.atanh(0.99627207622075))
    print("math.sinh(math.pi) =", math.sinh(math.pi))
    print("math.cosh(math.pi) =", math.cosh(math.pi))
    print()

    print("Логарифмические функции")

```

```

    print("math.log(148.41315910257657) =",
math.log(148.41315910257657))
    print("math.log(148.41315910257657, 2) =",
math.log(148.41315910257657, 2))
    print("math.log(148.41315910257657, 10) =",
math.log(148.41315910257657,10))
    print("math.log(1.0000025) =", math.log(1.0000025))
    print("math.log1p(0.0000025)  =", math.log1p(0.0000025))
    print("math.exp(5) =", math.exp(5))
    print("math.e**5  =", math.e ** 5)
    print()

    print("Округление:")
    print("math.ceil(1.001) =", math.ceil(1.001))
    print("math.floor(1.001) =", math.floor(1.001))
    print("math.factorial(10) =", math.factorial(10))
    print("math.gcd(10,125) =", math.gcd(10, 125))
    print("math.trunc(1.001) =", math.trunc(1.001))
    print("math.trunc(1.999) =", math.trunc(1.999))
    print()

if (__name__ == "__main__"):
    main()

```

1.6 Техника работы со строками

Листинг 12. K7_1_1

'''

K7_1. Техника работы со строками;

Задание 1.

<https://stepik.org/lesson/201702/step/5?unit=175778>

С клавиатуры вводятся строки, последовательность заканчивается точкой.

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

'''

```
a = input()
while a != ".":
    print(" ".join(a.upper()))
    a = input()
```

Листинг 13. K7_1_2

'''

Задание 2. <https://stepik.org/lesson/201702/step/8?unit=175778>

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.

'''

```
string = str(input())
string = string.replace(' ', '_')
print(string)
```

Листинг 14. K7_1_3

''''''

Задание 3. <https://stepik.org/lesson/201702/step/9?unit=175778>

Уберите точки из введенного IP-адреса. Выведите сначала четыре числа через пробел,

а затем сумму получившихся чисел.

''''''

```
string = input().split('.')
print(*string)
s = 0
for i in string:
    s += int(i)
print(s)
```


Листинг 15. K7_1_4

.....

Задание 4. <https://stepik.org/lesson/201702/step/14?unit=175778>

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 сообщение. Модификаторы состоят из двух одинаковых символов

и записываются по разу в начале и в конце строки.

.....

```
while True:
    b = input()
    if b == ".":
        break
    elif "==" in b:
        print("ошибка")
    elif "!!" in b:
        print("предупреждение")
    elif "/" in b:
        print("информация")
    elif "***" in b:
        print("подробное сообщение")
```

Листинг 16. K8_1_1

"""

Задание 1. https://pythontutor.ru/lessons/lists/problems/more_than_neighbours/

Задача «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.

"""

```
n = [int(i) for i in input().split()]
a = 0
for i in range(2, len(n)):
    if n[i-2] < n[i-1] > n[i]:
        a += 1
print(a)
```

Листинг 17. K8_1_2

"""

Задание 2. https://pythontutor.ru/lessons/lists/problems/num_equal_pairs/

Задача «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.

"""

```
n = [int(i) for i in input().split()]
a = 0
for i in range(len(n)-1):
    for j in range(i + 1, len(n)):
        if n[i] == n[j]:
            a += 1
print(a)
```

Листинг 18. K8_2_1

"""

Задание 1. Array112. Дан массив A размера N.

Упорядочить его по возрастанию методом сортировки

простым обменом: просматривать массив, сравнивая его соседние элементы 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.

"""

```
from random import randint
n = int(input())
a = []
```

```

for i in range(n):
    a.append(randint(1,100))
print(a)
for i in range(n-1):
    for j in range(n-i-1):
        if a[j] > a[j+1]:
            a[j], a[j+1] = a[j+1], a[j]
    print(a)
print(a)

```

Листинг 19. К8_2_3

.....

Задание 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).

.....

```

a = []
n = int(input())
for i in range(n):
    a.append(int(input()))
for i in range(n - 1):
    for j in range(n - 1 - i):
        MIN, MAX = min(a[j], a[j + 1]), max(a[j], a[j + 1])
        a[j], a[j + 1] = MIN, MAX
print(a)

```

1.8 Техника работы с циклом for и генераторами списков

Листинг 19. K9_2_1

.....

Задание 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
n = int(input())
a = [random.randint(1, 20) for x in range(n)]
print('Размер полученного массива A:', len(a))
print(a)
b = [a[i] for i in range(1, len(a), 2)]
print('Размер полученного массива B:', len(b))
print(b)
```

Листинг 20. K9_2_2

'''

Задание 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.

'''

```
n = int(input())
a = [int(input()) for _ in range(n)]
b = []
for i in range(0, n, 2):
    b.append(a[i])
for i in range(1, n, 2):
    b.append(a[i])
print(*b)
```

Листинг 21. K9_2_3

.....

Задание 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.

.....

```
n = int(input())
a = [int(input()) for _ in range(n)]
```

```
b=[]  
summ = 0  
for k in range(0,len(a)):  
    summ += a[k]  
    b.append(summ)  
print(*b)
```

Листинг 22. K10_1_2

'''

Задание 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):
    summ = 0
    for i in range(a, b+1):
        summ += i
    return summ
print("Введите числа:")
a = [int(x) for x in input().split()]
lna = len(a)
summ = 0
for i in range(1, lna):
    summ += SumRange(a[i-1], a[i])
print(summ)
```

Листинг 23. K10_1_3

'''

Задание 3. Func10. Описать функцию IsSquare(K) логического типа, возвращающую True, если целый параметр K (> 0) является квадратом некоторого целого числа, и False в противном случае. С ее помощью найти количество квадратов в наборе из 10 целых положительных чисел.

'''

```
import math

def IsSquare(k):
    return math.sqrt(k).is_integer()

i = 0
count = 0
while i != 10:
    num = int(input())
    print(IsSquare(num))
    i += 1
```

Листинг 24. K10_2_2

"""

Задание 2. Использовать `map`, `lambda`. Квадраты в обратном порядке. Числа вводятся до точки. Через пробел выведите эти числа в обратном порядке, возводя их в квадрат.

"""

```
a = [int(x)**2 for x in iter(input, '.')]
b = list(map(lambda x: print(x, end=' '), a[::-1]))
print()
```

Листинг 25. K10_2_3

"""

Задание 3. Использовать `lambda`, `filter`.

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.

"""

```
n = int(input("Введите размер списка: "))
a = [int(input(f"{x}| ")) for x in range(n)]
b = list(filter(lambda x: x%2, a))
print("\n")
print("Список B:", *b, sep=" ")
print("Размер списка b:", len(b))
```

Листинг 26. K10_2_4

"""

Задание 4.

Быстрая инициализация. Программа получает на вход три числа через пробел — начало и конец диапазона, а также степень, в которую нужно возвести каждое число из диапазона. Выведите числа получившегося списка через пробел.

"""

```
a = int(input("Начало: "))
b = int(input("Конец: "))
pow = int(input("Степень: "))
arr = [x**pow for x in range(a, b+1)]
print(*arr)
```

1.10 Техника работы со словарями

Листинг 27. K11_1_1

Задание 1. https://pythontutor.ru/lessons/dicts/problems/occurency_index/

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.

```
d = dict()
i = 0
for key in input().split():
    d[word] = d.get(word, 0) + 1
    print(d[word] - 1, end = ' ')
print()
```