

Колледж космического машиностроения и технологии

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: Пихтов Виталий Алексеевич, Сумин Константин Евгениевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(подпись)

Проверил преподаватель: Гусятинер Леонид Борисович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(оценка)

Королев, 2020

Практическое занятие №2.

Материал подготовил студент группы П2-18:

1| Пихтов Виталий Алексеевич

Установка python в windows.

1. Заходим на сайт python.org, и переходим в Downloads -> Windows.

2. Ищем стабильную новейшую версию в списке Stable Releases, и находим установщик python которая соответствует разрядности системы (32 (x86) bit или 64 bit).

3. Запускаем скачанный установщик и важно установить флажок в разделе Add Python 3.x to PATH, чтобы интерпретатор был установлен в вашем пути выполнения. Нажать Install Now, где указан путь установки

4. В конце установки можно отключить ограничение длины пути выполнения.

Порядок установки записан в прикрепленном файле ниже, installpython.mp4.



Практическое занятие №3.

3.1

Материал подготовил студент группы П2-18:

1| Пихтов Виталий Алексеевич

Создание файла с кодом.

Код исходной программы:  
print('Hello, world') **(test.py)**

Запуск.

Для того, чтобы открыть существующий файл с кодом, нужно:  
1. В папке с нужным файлом нажать сочетание клавиш Shift + ПКМ.

2. В появившемся диалоговом окне нажать “Открыть окно PowerShell здесь”. (см. приложение 1)

3. Ввести команду “python название-файла”.

4. Получить результат выполнения программы (см. приложение 2).

Import.

Чтобы подключить определенную библиотеку, нужно:  
1. В папке с нужным файлом нажать сочетание клавиш Shift + ПКМ.

2. В появившемся диалоговом окне нажать “Открыть окно PowerShell здесь”. (см. приложение 1)

3. Ввести в консоль команду “python”.

4. Ввести в консоль команду “import название-библиотеки”. (см. приложение 3)

Reload.

Чтобы перезагрузить определенное нечто, требуется:  
1. В папке с нужным файлом нажать сочетание клавиш Shift + ПКМ.

2. В появившемся диалоговом окне нажать “Открыть окно PowerShell здесь”. (см. приложение 1)

3. Ввести в консоль команду “python”.  
4. Импортировать нужный файл. (см. приложение 4)  
5. Перезаписать исходный файл.   
Код измененной программы:

print('Hello') **(test.py)**

6. Написать команду “reload sample”.

Отработка ошибок.

Примеры часто встречающихся ошибок:



Данная ошибка говорит о не прописанной команде “python”.



Данная ошибка говорит о том, что интерпретатор не может найти переменную с указанным именем.



(Приложение 1)



(Приложение 2)



(Приложение 3)



3-2.

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.



Практическое занятие №4.

Задачи на Boolean.

**Задача 1: ( begin1.py )**

'''Выполнил Пихтов В. А. П2-18.

Задача "Площадь прямоугольного треугольника".

Напишите программу, которая считывает длины двух катетов в прямоугольном треугольнике и выводит его площадь. Каждое число записано в отдельной строке.'''

a = int(input())

b = int(input())

print(a \* b / 2)

**Задача 2: ( begin2.py )**

'''Выполнил Пихтов В. А. П2-18.

Ввод и вывод данных. Задача «Дележ яблок».

n школьников делят k яблок поровну, неделящийся остаток остается в корзинке. Сколько яблок достанется каждому школьнику? Сколько яблок останется в корзинке? Программа получает на вход числа n и k и должна вывести искомое количество яблок (два числа).'''

n = int(input())

k = int(input())

print(k // n)

print(k % n)

Задачи на Case.

**Задача 1: ( case1.py )**

'''Выполнил Пихтов В. А. П2-18.

Пятилетний Петя учит цвета радуги. Напишите программу, которая принимает на вход натуральное число n и печатает первые n цветов радуги с большой буквы. При этом, если n > 7, программа должна ответить "Радуга состоит только из семи цветов".

RainbowSix = ("Красный", "Оранжевый", "Желтый", "Зеленый", "Голубой", "Синий", "Фиолетовый")'''

a = int(input())

if a <= 7:

for b in range(a):

print(RainbowSix[b])

elif a > 7:

print("Радуга состоит только из семи цветов")

**Задача 2: ( case2.py )**

'''Выполнил Пихтов В. А. П2-18.

Напишите программу, которая печатает названия месяцев до месяца n (n натуральное, вводится с клавиатуры) согласно шаблону, представленному в примере. Если n > 12, программа должна вывести "Error".

a = ["Первый месяц - январь", "Второй месяц - февраль", "Третий месяц - март", "Четвертый месяц - апрель", "Пятый месяц - май", "Шестой месяц - июнь", "Седьмой месяц - июль", "Восьмой месяц - август", "Девятый месяц - сентябрь", "Десятый месяц - октябрь", "Одиннадцатый месяц - ноябрь", "Двенадцатый месяц - декабрь",]'''

b = int(input())

if b <= 12:

for i in range(0, b):

print(a[i])

if b > 12:

print("Error")

**Задача 3: ( case3.py )**

'''Выполнил Пихтов В. А. П2-18.

Некий гражданин NN пришел в магазин с n рублями. Просмотрев свой список покупок, он считает стоимость всех товаров и соотносит ее с той суммой, что у него есть. Если ему хватает денег, то выведите "Покупает", иначе "Не покупает".

Формат ввода: на первой строке вводится натуральное число – количество денег, на второй строке – количество товаров n, на следующих n строках – цена каждого из товаров.

Формат вывода: строки "Покупает" или "Не покупает".'''

money = int(input())

count = int(input())

counter = 0

for n in range(count):

a = int(input())

counter = counter + a

if counter <= money:

b = "Покупает"

else:

b = "Не покупает"

print(b)

Задачи на If.

**Задача 1: ( if1.py )**

'''Выполнил Пихтов В. А. П2-18.

Напишите функцию fib(n), которая по данному целому неотрицательному n возвращает n-e число Фибоначчи. В этой задаче нельзя использовать циклы — используйте рекурсию.'''

def fib(n):

if (n == 1) or (n == 2):

return 1

else:

return fib(n - 1) + fib(n - 2)

n = int(input())

print(fib(n))

**Задача 2: ( if2.py )**'''Выполнил Пихтов В. А. П2-18.

Выведите все четные элементы списка. При этом используйте цикл for, перебирающий элементы списка, а не их индексы!'''

spisok = input().split()

for n in spisok:

if int(n) % 2 == 0:

print(n, end = ' ')

**Задача 3: ( if3.py )**

'''Выполнил Пихтов В. А. П2-18.

Дан список чисел. Выведите все элементы списка, которые больше предыдущего элемента.'''

spisok = [int(n) for n in input().split()]

for n in range(1, len(spisok)):

if spisok[n] > spisok[n - 1]:

print(spisok[n])

**Задача 1: ( boolean1.py )**

'''Выполнил Пихтов В. А. П2-18.

Пользователь вводит числа по одному в строке, последовательность оканчивается числом 0. Выведите сумму чисел, введенных до нуля.'''

summa = 0

a = int(input())

while a != 0:

summa += a

a = int(input())

print(summa)

**Задача 2: ( boolean2.py )**

'''Выполнил Пихтов В. А. П2-18.

Минимальный делитель.

Программа получает на вход натуральное число n > 1. Выведите минимальный делитель этого числа, отличный от единицы.'''

a = int(input())

n = 2

while a % n != 0:

n += 1

print(n)

Блок-схемы для задач к практическому занятию №4

Блок-схема для задачи boolean1:



(Рис. 1)

Блок-схема для задачи boolean2:  


(Рис. 2)

Практическое занятие №5.

Задачи на For.

**Задача 1: ( for1.py )**

'''Выполнил Пихтов В. А. П2-18.

Цикл с условием.

С клавиатуры вводятся числа a и b. Необходимо вывести те числа на полуинтервале [a; b) или [b; a), которые являются четными и дают остаток 1 при делении на 7.

Примечание: запись [a; b) означает промежуток из чисел от a включительно до b невключительно. В языке Python это соответствует записи range(a, b). Аналогично, промежуток [b; a) соответствует записи range(b, a). '''

a = int(input())

b = int(input())

if a < b:

for c in range(a, b):

if c % 7 == 1 and c % 2 == 0:

print(c)

else:

for c in range(b ,a):

if c % 7 == 1 and c % 2 == 0:

print(c)

**Задача 2: ( for2.py )**

'''Выполнил Пихтов В. А. П2-18.

Посчитайте сумму всех чисел на полуинтервале [a; b) или [b; a), в зависимости от того, какое число больше.'''

a = int(input())

b = int(input())

sum = 0

if a < b:

for c in range(a, b):

sum += c

else:

for c in range(b, a):

sum += c

print(sum)

**Задача 3: ( for3.py )**

'''Выполнил Пихтов В. А. П2-18.

Вы уже знакомы с термином "факториал", однако предыдущее наше решение не идеально – оно считает факториал только одного числа. Модифицируйте вашу программу из задачи "18!" таким образом, чтобы она принимала на вход натуральное число n и печатала значение выражения n!'''

a = int(input())

factorial = 1

for n in range(2, a+1):

factorial \*= n

print(factorial)

**Задача 4: ( for4.py )**

'''Выполнил Пихтов В. А. П2-18.

Дан список из чисел и индекс элемента в списке k. Удалите из списка элемент с индексом k, сдвинув влево все элементы, стоящие правее элемента с индексом k.

Программа получает на вход список, затем число k. Программа сдвигает все элементы, а после этого удаляет последний элемент списка при помощи метода pop() без параметров.

Программа должна осуществлять сдвиг непосредственно в списке, а не делать это при выводе элементов. Также нельзя использовать дополнительный список. Также не следует использовать метод pop(k) с параметром.'''

spisok = [int(s) for s in input().split()]

a = int(input())

for n in range(a + 1, len(spisok)):

spisok[n - 1] = spisok[n]

spisok.pop()

print(' '.join([str(n) for n in spisok]))

**Задача 5: ( for5.py )**

'''Выполнил Пихтов В. А. П2-18.

Выведите все элементы списка с четными индексами (то есть A[0], A[2], A[4], ...).'''

spisok = input().split()

for index in range(0, len(spisok), 2):

print(spisok[index])

Задачи на While.

**Задача 1: ( while1.py )**

'''Выполнил Пихтов В. А. П2-18.

По данному целому числу N распечатайте все квадраты натуральных чисел, не превосходящие N, в порядке возрастания.'''

b = 1

a = int(input())

while b \*\* 2 <= a:

print(b \*\* 2)

b += 1

**Задача 2: ( while2.py )**

'''Выполнил Пихтов В. А. П2-18.

Дано целое число, не меньшее 2. Выведите его наименьший натуральный делитель, отличный от 1.'''

b = 2

a = int(input())

while a % b != 0:

b += 1

print(b)

**Задача 3: ( while3.py )**

'''Выполнил Пихтов В. А. П2-18.

По данному натуральному числу N найдите наибольшую целую степень двойки, не превосходящую N. Выведите показатель степени и саму степень.

Операцией возведения в степень пользоваться нельзя!'''

a = int(input())

b = 1

c = 2

while c <= a:

c \*= 2

b += 1

print(b - 1, c // 2)

**Задача 4: ( while4.py )**

'''Выполнил Пихтов В. А. П2-18.

В первый день спортсмен пробежал x километров, а затем он каждый день увеличивал пробег на 10% от предыдущего значения. По данному числу y определите номер дня, на который пробег спортсмена составит не менее y километров.

Программа получает на вход действительные числа x и y и должна вывести одно натуральное число.'''

x = int(input())

y = int(input())

n = 1

while y > x:

x \*= 1.1

n += 1

print(n)

**Задача 5: ( while5.py )**

'''Выполнил Пихтов В. А. П2-18.

Программа получает на вход последовательность целых неотрицательных чисел, каждое число записано в отдельной строке. Последовательность завершается числом 0, при считывании которого программа должна закончить свою работу и вывести количество членов последовательности (не считая завершающего числа 0). Числа, следующие за числом 0, считывать не нужно.'''

dlinna = 0

while int(input()) != 0:

dlinna += 1

print(dlinna)

Блок-схемы для задач к практическому занятию №5

Блок-схема для задачи while1:



(Рис. 1)

Блок-схема для задачи while5:



(Рис. 2)

5.2

'''Выполнил: Сумин Константин Евгениевич

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.

Sample Input 1:

5

-3

8

4

0

Sample Output 1:

14

(5.2-2.py)'''

mas=[-1]

print("Введите числа: ")

while(mas.count(0)!=1):

mas.insert(1,int(input()))

mas[0]=mas[0]+mas[1]

#print(mas) если нужно посмотреть лист

print("Сумма: ",mas[0]+1)

'''Выполнил: Сумин Константин Евгениевич

Задание 3.

Разработать программу для нахождения наибольшего общего делителя

(5.2-3.py)'''

print("Ввелите 2 числа: ")

a=int(input())

b=int(input())

if a<b:

a,b=b,a

for i in range (a,0,-1):

if a%i==0 and b%i==0:

break

print("Наибольший общий делитель: ",i)

'''Выполнил: Сумин Константин Евгениевич

Задание 4.

С использованием результата задания 2 разработать программу для нахождения наименьшего

общего кратного

(5.2-4.py)'''

mas=[-1]

print("Введите числа: ")

while(mas.count(0)!=1):

mas.insert(1,int(input()))

mas[0]=mas[0]+mas[1]

#print(mas) #если нужно посмотреть лист

print("Сумма: ",mas[0]+1)

c=1

for i in range (2,len(mas)-1,+1):

a=abs(mas[i])

b=abs(mas[i+1])

if a<b:

a,b=b,a

if a%b==0:

mas[i+1]=a

c=a

else:

c=a

mas[1]=0

while(mas[1]==0):

if c%b==0 and c%a==0:

mas[i+1]=c

mas[1]=1

else:

c+=1

print("НОК: ",c)

'''выполнил: Пихтов Виталий Алексеевич

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 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.

Sample Input:

7

Sample Output:

1 2 2 3 3 3 4

(5.2-5.py)'''

a = int(input())

b = []

for n in range(a):

j = 0

while j < n + 1:

b.append(n + 1)

j += 1

if len(b) > a: break

b = b[0:a]

for n in b:

print(n, end=" ")

Практическое занятие №6.

Decimal.

'''Выполнил: Пихтов Виталий Алексеевич

Группа: П2-18

Decimal- это вычисления с заданной точностью

2.4 + 3.8 # Ответом будет 6.199999999999999, в нашем же случае 6.2'''

from decimal import Decimal

a = Decimal('2.4') + Decimal('3.8')

print(a)

2.4 + 3.8 == 6.2 # должно быть True, но с числами типа Decimal все верно:

c = Decimal('2.4') + Decimal('3.8')

print(c)

'''С помощью дополнительных знаков мы можем определить, сколько будет символов в дробной части числа:'''

from decimal import Decimal

num = Decimal("0.20")

num = 3 \* num

print(num) # 0.60

'''Однако нельзя смешивать в операциях дробные числа float и Decimal:'''

number = Decimal("0.1")

number = number + 0.1 # здесь возникнет ошибка

'''Округление чисел

Объекты Decimal имеют метод quantize(),который позволяет округлять числа.

В этот метод в качестве первого аргумента передается также объект Decimal,

который указывает формат округления числа:'''

number = Decimal("0.888")

number = number.quantize(Decimal("1.00"))

print(number) # 0.89

number = Decimal("0.1112345")

print(number.quantize(Decimal("1.00"))) # 0.11

number = Decimal("0.999")

print(number.quantize(Decimal("1.00"))) # 1.00

'''Как отделить целое от дроби в вещественных числах ?

Decimal вместо float:'''

a = Decimal('580.140')

b = int(a)

c = int(100 \* (a - b))

print(b, c) # => 580 14

**(decimal1.py)**

'''Использовался сайт

<https://metanit.com/python/tutorial/6.4.php> '''

Fraction.

'''Пихтов Виталий Алексеевич

Модуль fractions'''

from fractions import Fraction

from decimal import Decimal

import math

'''Fraction.limit\_denominator(max\_denominator=1000000)- ближайшее

рациональное число со знаменателем не больше данного.'''

a = Fraction(3.1415) ''' Целое и вещественное число, так же можно преобразовать в обыкновенную дробь'''

print(a) # 7074029114692207/2251799813685248

print(a.limit\_denominator()) # 6283/2000

# по умолчанию numerator = 0, denominator = 1

print(Fraction())

# равносильно Fraction(1, 2)

print(Fraction(numerator=1, denominator=2))

'''Если указанные числитель и знаменатель имеют общие делители,

то перед созданием рационального числа они будут сокращены'''

print(Fraction(8, 16), Fraction(15, 30))

'''Если указанные числитель и знаменатель имеют общие делители,

то перед созданием рационального числа они будут сокращены'''

print(Fraction(3, Fraction(1, 2)))

#Округляет до ближайшего четного числа.

print(Fraction('1/2').\_\_round\_\_())

'''создает обыкновенную дробь, которая является точным

представлением

десятичной дроби указанной в dec, где dec – это экземпляр класса decimal.Decimal'''

print(Fraction.from\_decimal(Decimal('0.7')))

'''принимает flt – число типа float и возвращает обыкновенную дробь отношение числителя

к знаменателю которой максимально приближается к значению flt.'''

print(Fraction.from\_float(0.5))

#fractions.gcd(a, b) - наибольший общий делитель чисел a и b.

print(math.gcd(1000, 3))

print(math.gcd(4, 6))

'''Использовался сайт <https://pyprog.pro/python/st_lib/fractions.html>

**(fraction1.py)'''**

Практическое занятие №7.

Техника работы со строками

'''выполнил: Пихтов Виталий Алексеевич

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.

(spiski1.py)'''

a = input()

List = []

while a != '.':

List.append(a)

a = input()

for n in List:

b = ''

for l in n:

b += l.upper() + ' '

print(b)

'''выполнил: Пихтов Виталий Алексеевич

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.

(spiski2.py)'''

a = input()

b = a.replace(" ", "\_")

print(b)

'''выполнил: Пихтов Виталий Алексеевич

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.

(spiski3.py)'''

l = []

a = input().split(".")

for n in a:

l.append(int(n))

a = " ".join(a)

print(a, "\n", sum(l))

'''выполнил: Пихтов Виталий Алексеевич

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 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

(spiski4.py)'''

a=input()

while a!='.':

if '@@' in a:

print('ошибка')

elif '!!' in a:

print('предупреждение')

elif '//' in a:

print('информация')

elif '\*\*' in a:

print('подробное сообщение')

a=input()

7.2

'''выполнил: Пихтов Виталий Алексеевич

1 Форматирование строк (оператор %)

Строки в Python содержат встроенную операцию, доступ к которой можно получить через оператор %. Это позволяет упростить позиционное форматирование. Вот простой пример:'''

Python

print('Hello, %s' % name)

# Вывод: "Hello, Bob"

print('Hello, %s' % name)

'''Вывод: "Hello, Bob"

Существуют другие определители формата, которые позволяют контролировать формат выдачи. Например, возможно конвертировать числа в шестнадцатеричную нотацию или добавлять пробелы для создания хорошо отформатированных таблиц и отчетов. (См Python Docs: ““printf-style String Formatting”.)

Можно использовать определитель формата %x для конвертации значения int в строку и представить его в качестве шестнадцатеричного числа:'''

Python

errno = 50159747054

print('%x' % errno)

# Вывод: 'badc0ffee'

errno = 50159747054

print('%x' % errno)

'''Вывод: 'badc0ffee'

Синтаксис форматирования строки немного меняется. Так как оператор % принимает только один аргумент, нужно обернуть правую часть в кортеж, вот так:'''

Python

errno = 50159747054

name = 'Bob'

print('Hey %s, there is a 0x%x error!' % (name, errno))

# 'Hey Bob, there is a 0xbadc0ffee error!'

errno = 50159747054

name = 'Bob'

print('Hey %s, there is a 0x%x error!' % (name, errno))

''''Hey Bob, there is a 0xbadc0ffee error!'

Также возможно сослаться на заменители переменных по имени в строке формата, если вы передадите сопоставление оператору %:'''

Python

print(

'Hey %(name)s, there is a 0x%(errno)x error!' % {

"name": name, "errno": errno

}

)

# Вывод: 'Hey Bob, there is a 0xbadc0ffee error!'

print(

'Hey %(name)s, there is a 0x%(errno)x error!' % {

"name": name, "errno": errno

}

)

'''Вывод: 'Hey Bob, there is a 0xbadc0ffee error!'

Это помогает сделать настройки формата более простым процессом, а также упрощает его модификацию в будущем. Недостаток — это то, что эта техника требует немного больше ручного ввода.

2 Форматирование строк (str.format)

Python 3 предоставил новый способ форматирования, который был внесен в раннюю версию Python 2.7. Это форматирование строк избавляется от специального синтаксиса оператора % и делает синтаксис для форматирования строк более регулярным. Теперь форматирование обрабатывается вызовом .format() в объекте строки.'''

Python

print('Hello, {}'.format(name))

# Вывод: 'Hello, Bob'

print('Hello, {}'.format(name))

'''Вывод: 'Hello, Bob'

Или можно сослаться на подстановки переменных по имени, и использовать их в том порядке, в котором хочется. Это достаточно мощный способ, так как он позволяет повторно упорядочить порядок отображения без изменения переданных функции format() аргументов:'''

Python

print(

'Hey {name}, there is a 0x{errno:x} error!'.format(

name=name, errno=errno

)

)

# Вывод: 'Hey Bob, there is a 0xbadc0ffee error!'

print(

'Hey {name}, there is a 0x{errno:x} error!'.format(

name=name, errno=errno

)

)

'''Вывод: 'Hey Bob, there is a 0xbadc0ffee error!'

Это также демонстрирует, что синтаксис формата переменной int — это шестнадцатеричная измененная строка.

Простой синтаксис

Синтаксис аналогичен тому, который вы используете в str.format(), но не такой перегруженный. Посмотрите на эту читабельность:'''

Python

name = "Eric"

age = 74

print(f"Hello, {name}. You are {age}.")

# Вывод: 'Hello, Eric. You are 74.'

name = "Eric"

age = 74

print(f"Hello, {name}. You are {age}.")

'''Вывод: 'Hello, Eric. You are 74.'

Произвольные выражения

Так как f-строки оцениваются по мере выражения, можно внести любую или все доступные выражения Python в них. Это позволит делать интересные вещи, например следующее:'''

Python

print(f"{2 \* 37}")

# Вывод: '74'

1

2

print(f"{2 \* 37}")

'''Вывод: '74'

Скорость

Буква f в f-strings может также означать и “fast”. f-строки заметно быстрее чем % и str.format() форматирования, f-строки являются выражениями, которые оцениваются по мере выполнения, а не постоянные значения.'''

Практическое занятие №8.

'''8-1.

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

выполнил: Пихтов Виталий Алексеевич

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 (spisok1.py).'''

a = [int(n) for n in input().split()]

b = 0

for n in range(1, len(a) - 1):

if a[n - 1] < a[n] > a[n + 1]:

b += 1

print(b)

'''выполнил: Пихтов Виталий Алексеевич

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 (spisok2.py).'''

a = [int(n) for n in input().split()]

summa = 0

b = 0

while b != len(a) - 1:

for n in range(b + 1, len(a)):

if a[b] == a[n]:

summa += 1

b += 1

print(summa)

'''Выполнил: Сумин Константин Евгениевич

Задание 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 (spisok3.py).'''

from random import randint

li=[]

mas=[]

jojo=[]

jostar=[]

bug=[]

n = int(input("Введите количество списков: "))

temp=0

for i in range (0,n,+1):

for j in range(0,randint(2,5),+1):

a=randint(0,5)

mas.insert(j, a)

if li.count(a)==0:

li.append(a)

if bug.count(a)==0:

bug.append(a)

temp+=1

print("Список №",temp,": ",mas)

jojo.extend(bug)

bug.clear()

mas.clear()

for i in range(0,6,+1):

if jojo.count(i)==n:

jostar.append(i)

print("В каждом списке есть: ",jostar)

print("Хотя бы в одном списке есть: ",li)

8.2

'''выполнил: Пихтов Виталий Алексеевич

Задание 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

n = random.randrange(2,10)

a = [random.randrange(0,10) for i in range(n)]

print("N = ", n)

print("Array:")

print(a)

for i in range(1, n) :

for j in range(0,n - i) :

if a[j] > a[j + 1] :

a[j], a[j + 1] = a[j + 1], a[j]

print("Sort Array:")

print(a)

'''выполнил: Пихтов Виталий Алексеевич

Задание 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

n = random.randrange(2, 10)

a = [random.randrange(1, 8) for i in range(n)]

print(n)

print(a)

for i in range(1, n):

d = False

b = a[i]

c = i - 1

while c >= 0 :

if b >= a[c] :

break

else :

a[c + 1] = a[c]

d = True

c -= 1

if d and c >= -1:

a[c + 1] = b

print(i)

print(a)

print(a)

'''выполнил: Пихтов Виталий Алексеевич

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).'''

import random

n = random.randrange(2, 10)

a = [random.randrange(1, 8) for i in range(n)]

print(n)

print(a)

for i in range(1, n):

d = False

b = a[i]

c = i - 1

while c >= 0 :

if b >= a[c] :

break

else :

a[c + 1] = a[c]

d = True

c -= 1

if d and c >= -1:

a[c + 1] = b

print(i)

print(a)

print(a)

Практическое занятие №9.

9.1

'''Выполнил: Сумин Константин Евгениевич

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 (kkmt.py)'''

kkmt=[]

kod=[]

file = open('name.txt', encoding='utf-8')

name=list(tuple(item.replace('\n', '') for item in line.split(' ')) for line in file)

for i in range(0,len(name),+1):

if kod.count(name[i][1]) == 0:

kod.append(name[i][1])

for i in range(0,len(kod),+1):

for j in range(0,len(name),+1):

temp=0

while temp<3:

if name[j][1] == kod[i] and temp==0 and name[j][2=='coder'] and kkmt.count(name[j])!=name.count(name[j]):

kkmt.append(name[j])

temp+=1

elif name[j][1] == kod[i] and temp==1 and name[j][2=='designer'] and kkmt.count(name[j])!=name.count(name[j]):

kkmt.append(name[j])

temp += 1

elif name[j][1] == kod[i] and temp==2 and name[j][2=='tester'] and kkmt.count(name[j])!=name.count(name[j]):

kkmt.append(name[j])

temp += 1

elif name[j][1] == kod[i] and temp==3 and name[j][2=='writer'] and kkmt.count(name[j])!=name.count(name[j]):

kkmt.append(name[j])

temp += 1

else:

temp=4

try:

n=0

pr=0

bug=0

aaa=0

for j in range(0,len(kod),+1):

coun=1

temp=0

for k in range (0,len(kkmt),+1):

temp+=kkmt[k].count(kod[j])

if temp/4!=0:

coun=int(temp/4)

if coun==1:

for i in range(n,n+temp,+1):

if bug%4==0:

print("Команда №",int(pr/4)+1)

print(kkmt[i])

pr += 1

aaa += 1

if bug!=3:

bug+=1

else:

bug=0

else:

for i in range(n, n + temp, +1):

if bug%4==0 and i%2!=0:

print("Команда №",int(pr/4)+1)

if i%2!=0:

print(kkmt[i])

pr += 1

aaa += 0.5

if bug != 3:

bug += 1

else:

bug = 0

for i in range(n, n + temp, +1):

if bug%4==0 and i%2==0:

print("Команда №",int(pr/4)+1)

if i%2==0:

print(kkmt[i])

pr+=1

aaa += 0.5

if bug != 3:

bug += 1

else:

bug = 0

n+=temp

if aaa!=len(kod)\*4:

print("Всязи с недостатком участников, некоторые команды были сформированы некорректно\n"

"Пожалуйста, уберите участников которым не хватает людей для команды, или найдите\n"

"им людей в команду")

except:

print("Всязи с недостатком участников, некоторые команды были сформированы некорректно\n"

"Пожалуйста, уберите участников которым не хватает людей для команды, или найдите\n"

"им людей в команду")

9.2

'''Одномерные массивы

<http://ptaskbook.com/ru/tasks/array.php>

выполнил: Пихтов Виталий Алексеевич

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. (array55.py)'''

import random

n = random.randrange(2, 24)

a = [i for i in range(n)]

b = a[1::2]

print(n)

print(a)

print(len(b))

print(b)

'''выполнил: Пихтов Виталий Алексеевич

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. (array57.py)'''

import random

n = random.randrange(2, 22)

a = [i for i in range(n)]

b = a[0::2] + a[1::2]

print(n)

print(a)

print(len(b))

print(b)

'''выполнил: Пихтов Виталий Алексеевич

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. (array58.py)'''

import random

n = random.randrange(2,22)

a = [i for i in range(n)]

b = []

b.append(a[0])

print(n)

print(a)

for i in range(1, n) :

b.append(a[i] + b[i - 1])

print(len(b))

print(b)

'''Двумерные массивы (матрицы)

<http://ptaskbook.com/ru/tasks/matrix.php>

выполнил: Пихтов Виталий Алексеевич

Задание 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). (matrix3.py)'''

import random

import numpy

a = random.randrange(2,10)

b = random.randrange(2,10)

print("a: ", a, "b: ", b)

c = numpy.zeros((a, b))

d = [random.randrange(1,5) for i in range(a)]

print(a)

print(d)

for i in range(a):

for j in range(b):

c[i][j] = d[i]

print(c)

matrix = []

for i in range(a):

matrix.append([d[i] for j in range(b)])

print(matrix)

'''Выполнил: Сумин Константин Евгениевич

Задание 5. Matrix56. Дана матрица размера M x N (N — четное число). Поменять местами

левую и правую половины матрицы. (matrix56.py)'''

matrix = [[1, 3, 5, 0], [3, 1, 8, 7], [9, 0, 6, 7]]

def swap(m: list):

size = int(len(m[0]) / 2)

left\_half = []

right\_half = []

for s in m:

left\_half.append(list(s[i] for i in range(size)))

right\_half.append(list(s[i] for i in range(size, len(s))))

res = []

for i in range(len(m)):

res.append(right\_half[i] + left\_half[i])

return res

for string in matrix:

for item in string:

print(item, end=' ')

print()

print()

result = swap(matrix)

for string in result:

for item in string:

print(item, end=' ')

print()

'''Выполнил: Сумин Константин Евгениевич

Задание 6. Matrix88. Дана квадратная матрица порядка M. Обнулить элементы матрицы,

лежащие ниже главной диагонали. Условный оператор не использовать. (matrix88.py)'''

matrix = [[1, 3, 5, 0], [3, 1, 8, 7], [9, 0, 6, 7], [0, 1, 7, 3]]

def m\_z(m: list):

for i in range(len(m)):

for j in range(i):

m[i][j] = 0

for string in matrix:

for item in string:

print(item, end=' ')

print()

print()

m\_z(matrix)

for string in matrix:

for item in string:

print(item, end=' ')

print()

print()

Практическое занятие №11.

'''выполнил: Пихтов Виталий Алексеевич

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. (stroki1.py)'''

a = dict()

for c in input().split():

a[c] = a.get(c, 0) + 1

print(a[c] - 1, end=' ')

'''выполнил: Пихтов Виталий Алексеевич

Задание 2. <https://pythontutor.ru/lessons/dicts/problems/permissions/>

Задача «Права доступа»

Условие. В файловую систему одного суперкомпьютера проник вирус, который сломал контроль

за правами доступа к файлам. Для каждого файла известно, с какими действиями можно к

нему обращаться:

запись W,

чтение R,

запуск X.

В первой строке содержится число N — количество файлов содержащихся в данной файловой

системе. В следующих N строчках содержатся имена файлов и допустимых с ними операций,

разделенные пробелами. Далее указано чиcло M — количество запросов к файлам. В последних

M строках указан запрос вида Операция Файл. К одному и тому же файлу может быть применено

любое колличество запросов.

Вам требуется восстановить контроль над правами доступа к файлам (ваша программа для

каждого запроса должна будет возвращать OK если над файлом выполняется допустимая

операция, или же Access denied, если операция недопустима. (stroki2.py)'''

a = [input().split() for i in range(int(input()))]

b = dict()

for i in a:

for j in range(1, len(i)):

if i[j] == 'R':

b[i[0]] = b.get(i[0], ['']) + ['read']

elif i[j] == 'W':

b[i[0]] = b.get(i[0], ['']) + ['write']

else:

b[i[0]] = b.get(i[0], ['']) + ['execute']

for i in range(int(input())):

act, name = input().split()

if act in b[name]:

print('OK')

else:

print('Access denied')

'''выполнил: Пихтов Виталий Алексеевич

Задание 3. <https://pythontutor.ru/lessons/dicts/problems/most_frequent_word/>

Задача «Самое частое слово»

Условие. Дан текст: в первой строке задано число строк, далее идут сами строки.

Выведите слово, которое в этом тексте встречается чаще всего. Если таких слов несколько,

выведите то, которое меньше в лексикографическом порядке. (stroki3.py)'''

a = int(input())

b = {}

max = 0

key = ''

for i in range(a):

S = input().split()

for j in S:

b[j] = b.get(j, 0) + 1

for k, v in sorted(b.items()):

if v > max:

max = v

key = k

print(key)

11.2

'''Выполнил: Сумин Константин Евгениевич

Задание 1. <https://stepik.org/lesson/243394/step/4?unit=215740>

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.

Sample Input:

Ben 89001234050

Alice 210-220

Alice

Alice 404-502

Nick +16507811251

Ben

Alex +4(908)273-22-42

Alice

Nick

Robert 51234047129

Alex

.

Sample Output:

210-220

89001234050

404-502

+16507811251

+4(908)273-22-42

'''

contacts={}

a=0

while a!='.':

a=input()

b=a.split(' ')

if a!='.':

if len(b)==2:

contacts[b[0]]=b[1]

elif len(b)==1:

print(contacts[b[0]])

Практическое занятие №12.

'''выполнил: Пихтов Виталий Алексеевич

Задание 1. <https://pythontutor.ru/lessons/sets/problems/number_of_unique/>

Задача «Количество различных чисел»

Условие. Дан список чисел. Определите, сколько в нем встречается различных чисел.''' (spisok1.py)

spisok = input().split()

mnogestvo = set(spisok)

dlina = len(mnogestvo)

print(dlina)

'''выполнил: Пихтов Виталий Алексеевич

Задание 2. <https://pythontutor.ru/lessons/sets/problems/number_of_coincidental/>

Задача «Количество совпадающих чисел»

Условие. Даны два списка чисел. Посчитайте, сколько чисел содержится одновременно как

в первом списке, так и во втором.''' (spisok2.py)

a = {int(s) for s in input().split()}

b = {int(s) for s in input().split()}

c = a & b

print(len(c))

'''выполнил: Пихтов Виталий Алексеевич

Задание 3. <https://pythontutor.ru/lessons/sets/problems/sets_intersection/>

Задача «Пересечение множеств»

Условие. Даны два списка чисел. Найдите все числа, которые входят как в первый,

так и во второй список и выведите их в порядке возрастания.''' (spisok3.py)

a = [int(x) for x in input().split()]

b = [int(x) for x in input().split()]

c = [str(x) for x in sorted(list(set(a) & set (b)))]

d = ' '.join(c)

print(d)

'''выполнил: Пихтов Виталий Алексеевич

Задание 4. <https://pythontutor.ru/lessons/sets/problems/number_of_words/>

Задача «Количество слов в тексте»

Условие. Дан текст: в первой строке записано число строк, далее идут сами строки.

Определите, сколько различных слов содержится в этом тексте.

Словом считается последовательность непробельных символов идущих подряд, слова разделены

одним или большим числом пробелов или символами конца строки.''' (spisok4.py)

a = set()

for n in range(int(input())):

a.update(input().split())

print(len(a))

'''выполнил: Пихтов Виталий Алексеевич

Задание 5. <https://pythontutor.ru/lessons/sets/problems/polyglotes/>

Задача «Полиглоты»

Условие. Каждый из некоторого множества школьников некоторой школы знает некоторое

количество языков. Нужно определить сколько языков знают все школьники, и сколько языков

знает хотя бы один из школьников.

В первой строке задано количество школьников. Для каждого из школьников сперва записано

количество языков, которое он знает, а затем - названия языков, по одному в строке.

В первой строке выведите количество языков, которые знают все школьники. Начиная со

второй строки - список таких языков. Затем - количество языков, которые знает хотя бы

один школьник, на следующих строках - список таких языков. Языки нужно выводить в

лексикографическом порядке, по одному на строке.''' (spisok5.py)

all = int(input())

a = set()

sum = set()

b = set()

for i in range(all):

a = int(input())

for i in range(a):

b.update(input().split())

sum |= b

a = sum & b

b = set()

print(len(a))

print(\*sorted(a))

Практическое занятие №13.

'''выполнил: Пихтов Виталий Алексеевич

Задание 1. <https://stepik.org/lesson/193753/step/4?unit=168148>

Вывести чётные

Необходимо вывести все четные числа на отрезке [a; a \* 10].

Sample Input:

2

Sample Output:

(2, 4, 6, 8, 10, 12, 14, 16, 18, 20)'''

a = int(input())

start = a if a % 2 == 0 else a + 1

end = a \* 10 + 1

k = tuple(range(start, end, 2))

print(k)

'''Выполнил: Сумин Константин Евгениевич

Задание 2. <https://stepik.org/lesson/193753/step/5?unit=168148>

Убывающий ряд.

С клавиатуры вводятся целые числа a > b. Выведите убывающую последовательность чисел

по одному числу в строке.

Sample Input:

-2

-8

Sample Output:

-2

-3

-4

-5

-6

-7'''

a = int(input())

b = int(input())

k = tuple(range(a, b, -1))

print(\*k, sep='\n')

'''Выполнил: Сумин Константин Евгениевич

Задание 3. (Л.Б.) В каждой строке файла хранится информация о пунктах и их координатах

относительно некоторого центра.

Требуется

1. Прочесть файл в список кортежей

2. Найти диаметр множества точек, то есть расстояние между наиболее удалёнными точками.

Указать наиболее удалённые пары

3. Сформировать список пар городов, имеющих одинаковое расстояние до центра

4. Отсортировать список одним из методов, реализованных в предыдущих работах

Результаты вывести на экран

Пример входного файла

Москва 0 0

Ивантеевка 20 15

Щёлково 10 30

Пушкино 15 5'''

from math import sqrt

def abc(s:str):

try:

int(s)

return True

except ValueError:

return False

def distance(x1, y1, x2, y2):

dx = x1 - x2 if x1 > x2 else x2 - x1

dy = y1 - y2 if y1 > y2 else y2 - y1

return sqrt(dx \* dx + dy \* dy)

def distancee(x1,y1):

return sqrt(x1\*x1+y1\*y1)

'''1 Прочесть файл в список кортежей'''

li= {}

f = open('goroda.txt', encoding='utf-8')

cities = list(tuple(int(item.replace('\n', '')) if abc(item.replace('\n', ''))

else item.replace('\n', '') for item in line.split(' ')) for line in f)

'''2 Найти диаметр множества точек, то есть расстояние между наиболее удалёнными точками.

Указать наиболее удалённые пары'''

for i in range(0,len(cities)-1,+1):

x1=cities[i][1]

y1=cities[i][2]

for j in range(i+1, len(cities), +1):

x2=cities[j][1]

y2=cities[j][2]

if li.get(distance(x1, y1, x2, y2)) is None:

li[distance(x1, y1, x2, y2)] = cities[i][0] + "-" + cities[j][0]

else:

li[distance(x1, y1, x2, y2)] += ' '+cities[i][0]+"-"+cities[j][0]

max = 0

for k in li.keys():

if max < k:

max = k

for l in li.keys():

if max == l:

print(li[l])

'''3 Сформировать список пар городов, имеющих одинаковое расстояние до центра'''

lii={}

for i in range(0,len(cities),+1):

x1=cities[i][1]

y1=cities[i][2]

if lii.get(distancee(x1, y1)) is None:

lii[distancee(x1, y1)] = cities[i][0]

else:

lii[distancee(x1, y1)] += ' '+cities[i][0]

print(lii.values())

'''4 Отсортировать список одним из методов, реализованных в предыдущих работах

Результаты вывести на экран'''

liii=[]

for j in range(0,len(cities),+1):

max = 0

for i in range(0,len(cities),+1):

x1=cities[i][1]

y1=cities[i][2]

if distancee(x1, y1)>max and liii.count(distancee(x1, y1))== 0:

max=distancee(x1, y1)

if liii.count(max)== 0:

liii.append(max)

print("От большего к меньшему",liii)

13.2

# Класс namedtuple() модуля collections возвращает новый подкласс кортежа с именем typename. Новый подкласс используется для создания объектов, похожих на кортежи, которые имеют индексируемые и итерируемые поля, доступные для поиска по атрибутам. Экземпляры подкласса также имеют полезную строку документации с typename и field\_names, а так же метод \_\_repr\_\_(), который перечисляет содержимое кортежа в формате name=value.

# Пример:

from collections import namedtuple

Book = namedtuple('Book', ['id', 'title', 'authors'])

Book.\_\_doc\_\_ += ': Hardcover book in active collection'

Book.id.\_\_doc\_\_ = '13-digit ISBN'

Book.title.\_\_doc\_\_ = 'Title of first printing'

Book.authors.\_\_doc\_\_ = 'List of authors sorted by last name'

Практическое занятие №14.

'''выполнил: Пихтов Виталий Алексеевич

Задание 1. <http://ptaskbook.com/ru/tasks/text.php>

Text5. Дана строка S и текстовый файл. Добавить строку S в конец файла.'''

v = open('text.txt','a');

v.write('S');

'''выполнил:

Задание 2. <http://ptaskbook.com/ru/tasks/text.php>

Text12. Дана строка S и текстовый файл. Заменить в файле все пустые строки на строку S.'''

'''выполнил: Пихтов Виталий Алексеевич

Задание 3. <http://ptaskbook.com/ru/tasks/text.php>

Text20. Дан текстовый файл. Заменить в нем все подряд идущие пробелы на один пробел.'''

file1 = "text20.txt"

file2 = "text20\_2.txt"

flag = True

try:

with open(file1, 'r') as infile, \

open(file2, 'w') as outfile:

for line in infile:

line\_new = ""

for c in line.strip("\n"):

if c != " ":

line\_new += c

flag = True

elif flag:

line\_new += c

flag = False

outfile.write(line\_new+"\n")

except IOError:

print('Open error: ', file1)

'''выполнил:

Задание 4. <http://ptaskbook.com/ru/tasks/text.php>

Text44. Дан текстовый файл, каждая строка которого изображает целое число, дополненное слева и справа

несколькими пробелами. Вывести количество этих чисел и их сумму.'''

'''выполнил:

Задание 5. <http://ptaskbook.com/ru/tasks/text.php>

Text53. Дан текстовый файл. Создать символьный файл, содержащий все знаки препинания,

встретившиеся в текстовом файле (в том же порядке).'''

Практическое занятие №15.

collections.deque

Составитель: Гусятинер Л.Б., 3.12.2020, МГОТУ ККМТ, П1-18, П2-18

Выполнил: Пихтов Виталий Алексеевич

**collections.deque(iterable, [maxlen])** - создаёт очередь из итерируемого объекта с максимальной длиной maxlen. Очереди очень похожи на списки, за исключением того, что добавлять и удалять элементы можно либо справа, либо слева.

Методы, определённые в deque:

**append(x)** - добавляет x в конец.

**appendleft(x)** - добавляет x в начало.

**clear()** - очищает очередь.

**count(x)** - количество элементов, равных x.

**extend(iterable)** - добавляет в конец все элементы iterable.

**extendleft(iterable)** - добавляет в начало все элементы iterable (начиная с последнего элемента iterable).

**pop()** - удаляет и возвращает последний элемент очереди.

**popleft()** - удаляет и возвращает первый элемент очереди.

**remove(value)** - удаляет первое вхождение value.

**reverse()** - разворачивает очередь.

**rotate(n)** - последовательно переносит n элементов из начала в конец (если n отрицательно, то с конца в начало).

collections.Counter

**collections.Counter** - вид словаря, который позволяет нам считать количество неизменяемых объектов (в большинстве случаев, строк). Пример:

import collections

c = collections.Counter()

for word in ['spam', 'egg', 'spam', 'counter', 'counter', 'counter']:

... c[word] += 1

...

print(c)

Counter({'counter': 3, 'spam': 2, 'egg': 1})

print(c['counter'])

# 3

print(c['collections'])

# 0

Но возможности Counter на этом не заканчиваются. У него есть несколько специальных методов:

**elements()** - возвращает список элементов в лексикографическом порядке.

c = Counter(a=4, b=2, c=0, d=-2)

list(c.elements())

['a', 'a', 'a', 'a', 'b', 'b']

**most\_common([n])** - возвращает n наиболее часто встречающихся элементов, в порядке убывания встречаемости. Если n не указано, возвращаются все элементы.

Counter('abracadabra').most\_common(3)

[('a', 5), ('r', 2), ('b', 2)]

subtract([iterable-or-mapping]) # - вычитание

c = Counter(a=4, b=2, c=0, d=-2)

d = Counter(a=1, b=2, c=3, d=4)

c.subtract(d)

Counter({'a': 3, 'b': 0, 'c': -3, 'd': -6})

Наиболее часто употребляемые шаблоны для работы с Counter:

sum(c.values()) # - общее количество.

c.clear() # - очистить счётчик.

list(c) # - список уникальных элементов.

set(c) # - преобразовать в множество.

dict(c) # - преобразовать в словарь.

c.most\_common()[:-n:-1] # - n наименее часто встречающихся элементов.

c += Counter() # - удалить элементы, встречающиеся менее одного раза.

Counter также поддерживает сложение, вычитание, пересечение и объединение:

c = Counter(a=3, b=1)

d = Counter(a=1, b=2)

c + d

Counter({'a': 4, 'b': 3})

c - d

Counter({'a': 2})

c & d

Counter({'a': 1, 'b': 1})

c | d

Counter({'a': 3, 'b': 2})

15.2

collections.defaultdict

Выполнил: Пихтов Виталий Алексеевич

**collections.defaultdict** ничем не отличается от обычного словаря за исключением того, что по умолчанию всегда вызывается функция, возвращающая значение:

import collections

defdict = collections.defaultdict(list)

print(defdict)

defaultdict(<class 'list'>, {})

for i in range(5):

... defdict[i].append(i)

...

print(defdict)

defaultdict(<class 'list'>, {0: [0], 1: [1], 2: [2], 3: [3], 4: [4]})

collections.OrderedDict

**collections.OrderedDict** - ещё один похожий на словарь объект, но он помнит порядок, в котором ему были даны ключи. Методы:

popitem(last=True) - удаляет последний элемент если last=True, и первый, если last=False.

move\_to\_end(key, last=True) - добавляет ключ в конец если last=True, и в начало, если last=False.

d = {'banana': 3, 'apple':4, 'pear': 1, 'orange': 2}

OrderedDict(sorted(d.items(), key=lambda t: t[0]))

OrderedDict([('apple', 4), ('banana', 3), ('orange', 2), ('pear', 1)])

OrderedDict(sorted(d.items(), key=lambda t: t[1]))

OrderedDict([('pear', 1), ('orange', 2), ('banana', 3), ('apple', 4)])

OrderedDict(sorted(d.items(), key=lambda t: len(t[0])))

OrderedDict([('pear', 1), ('apple', 4), ('orange', 2), ('banana', 3)])

15.3

sys.argv

выполнил: Пихтов Виталий Алексеевич

Метод argv возвращает аргументы командной строки, переданные скрипту Python, в виде списка. Важно отметить, что первый аргумент (с индексом 0) в списке — это название самого скрипта. Остальные представлены в виде последовательности.В следующем примере файл с кодом называется test.py.

import sys

print("Привет {}. Добро пожаловать в руководство по {} на {}".format(sys.argv[1], sys.argv[2], sys.argv[3]))

Запуск test.py в командной строке вместе со следующими аргументами:

python test.py Студенты sys PythonRu

В указанной выше команде первым аргументом функции argv является элемент Студенты. sys и PythonRu — второй и третий аргументы.

Вывод:

Привет Студенты. Добро пожаловать в руководство по sys на PythonRu

Имя используемой OS

Модуль os предоставляет множество функций для работы с операционной системой, причём их поведение, как правило, не зависит от ОС, поэтому программы остаются переносимыми. Здесь будут приведены наиболее часто используемые из них.

Будьте внимательны: некоторые функции из этого модуля поддерживаются не всеми ОС.

**os.name** - имя операционной системы. Доступные варианты: 'posix', 'nt', 'mac', 'os2', 'ce', 'java'.

import os

print(os.name) # ответ: nt

**os.environ** - словарь переменных окружения. Изменяемый (можно добавлять и удалять переменные окружения).

import os

print(os.environ)

**os.getlogin()** - имя пользователя, вошедшего в терминал (Unix).

**os.getpid()** - текущий id процесса.

**os.uname()** - информация об ОС. возвращает объект с атрибутами: sysname - имя операционной системы, nodename - имя машины в сети (определяется реализацией), release - релиз, version - версия, machine - идентификатор машины.

**os.access(path, mode, \*, dir\_fd=None, effective\_ids=False, follow\_symlinks=True)** - проверка доступа к объекту у текущего пользователя. Флаги: os.F\_OK - объект существует, os.R\_OK - доступен на чтение, os.W\_OK - доступен на запись, os.X\_OK - доступен на исполнение.

**os.chdir(path)** - смена текущей директории.

**os.chmod(path, mode, \*, dir\_fd=None, follow\_symlinks=True)** - смена прав доступа к объекту (mode - восьмеричное число).

**os.chown(path, uid, gid, \*, dir\_fd=None, follow\_symlinks=True)** - меняет id владельца и группы (Unix).

**os.getcwd()** - текущая рабочая директория.

**os.link(src, dst, \*, src\_dir\_fd=None, dst\_dir\_fd=None, follow\_symlinks=True)** - создаёт жёсткую ссылку.

**os.listdir(path=".")** - список файлов и директорий в папке.

**os.mkdir(path, mode=0o777, \*, dir\_fd=None)** - создаёт директорию. OSError, если директория существует.

**os.makedirs(path, mode=0o777, exist\_ok=False)** - создаёт директорию, создавая при этом промежуточные директории.

**os.remove(path, \*, dir\_fd=None)** - удаляет путь к файлу.

**os.rename(src, dst, \*, src\_dir\_fd=None, dst\_dir\_fd=None)** - переименовывает файл или директорию из src в dst.

**os.renames(old, new)** - переименовывает old в new, создавая промежуточные директории.

**os.replace(src, dst, \*, src\_dir\_fd=None, dst\_dir\_fd=None)** - переименовывает из src в dst с принудительной заменой.

os.rmdir(path, \*, dir\_fd=None) - удаляет пустую директорию.

**os.removedirs(path)** - удаляет директорию, затем пытается удалить родительские директории, и удаляет их рекурсивно, пока они пусты.

**os.symlink(source, link\_name, target\_is\_directory=False, \*, dir\_fd=None)** - создаёт символическую ссылку на объект.

**os.sync()** - записывает все данные на диск (Unix).

**os.truncate(path, length)** - обрезает файл до длины length.

**os.utime(path, times=None, \*, ns=None, dir\_fd=None, follow\_symlinks=True)** - модификация времени последнего доступа и изменения файла. Либо times - кортеж (время доступа в секундах, время изменения в секундах), либо ns - кортеж (время доступа в наносекундах, время изменения в наносекундах).

**os.walk(top, topdown=True, onerror=None, followlinks=False)** - генерация имён файлов в дереве каталогов, сверху вниз (если topdown равен True), либо снизу вверх (если False). Для каждого каталога функция walk возвращает кортеж (путь к каталогу, список каталогов, список файлов).

**os.system(command)** - исполняет системную команду, возвращает код её завершения (в случае успеха 0).

**os.urandom(n)** - n случайных байт. Возможно использование этой функции в криптографических целях.

**os.path** - модуль, реализующий некоторые полезные функции на работы с путями.

Различные сведения о версии Python

**sys.copyright**

Строка, дающая информацию об авторских правах на интерпретатор Python.

**sys.version**

Строка, состоящая из номера версии Python, а также дополнительной информации о номере сборки и используемом компиляторе.

Например:

3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:21:23) [MSC v.1916 32 bit

(Intel)]

**sys.api\_version**

Информацию об используемой интерпретатором версии C API. Применяется при отладке конфликтов, возникающих при взаимодействии Python и дополнительных модулей.

Пример значения: 1013.

**sys.version\_info:**

Переменная sys.api\_version возвращает кортеж, содержащий пять компонентов номера версии:

* major,
* minor,
* micro,
* releaselevel,
* serial.

import sys

sys.version\_info.major

**sys.implementation**

Объект, дающий информацию о запущенном в данный момент интерпретаторе Python.

Объект должен содержать атрибуты: имя, версия, версия в шестнадцатеричном представлении, дескриптор кэша.

Пример:

namespace(cache\_tag='cpython-38', hexversion=50856176, name='cpython',

version=sys.version\_info(major=3, minor=8, micro=0, releaselevel='final',

serial=0))

**sys.hexversion**

Номер версии интерпретатора Python, закодированный одним числом. Это число увеличивается с каждой версией, включая все виды релизов.

sys.hexversion используют, чтобы удостовериться, поддерживает ли интерпретатор какую-либо функцию. Если нет, то функция заменяется каким-либо поддерживаемым аналогом.

Это число называется «hexversion», потому что оно принимает понятный вид, только если его передать в функцию hex().

Пример версии: 50856176.

**sys.winver**

Содержит номер версии Python, который используется в реестре Windows. Например, «3.8-32».

Используется только в информационных целях, то есть даже если изменить значения «sys.winver», значения в реестре Windows не изменятся.

Каталоги и пути интерпретатора Python.

**sys.prefix**

Атрибут sys.prefix это строка, предоставляет специфичный для площадки префикс каталога, в котором установлены независимые от платформы файлы Python. По умолчанию, это строка '/usr/local'.

Значение sys.prefix устанавливается во время сборки Python с помощью аргумента --prefix для скрипта configure, например $ ./configure --prefix=/opt/python-3.x.x/". Основная коллекция модулей библиотеки Python установлена в каталоге prefix/lib/pythonX.Y, а независимые от платформы заголовочные файлы (все, кроме pyconfig.h) хранятся в prefix/include/pythonX.Y, где XY - номер версии Python, например 3.7.

import sys

sys.prefix

sys.real\_prefix

**sys.base\_prefix**

Атрибут sys.base\_prefix устанавливается при запуске Python перед запуском site.py то же значение, что и атрибут sys.prefix. Если не работает в виртуальной среде, значения останутся прежними. Если site.py обнаружит, что используется виртуальная среда, то значения sys.prefix и sys.exec\_prefix будут изменены на значения установки виртуальной среды, тогда как sys.base\_prefix и sys.base\_exec\_prefix останутся указывать на базовую установку Python (ту, из которой была создана виртуальная среда).

**sys.exec\_prefix**

Атрибут sys.exec\_prefix это строка, задает префикс каталога для конкретной программы, где установлены зависящие от платформы файлы Python. По умолчанию это '/usr/local'.

Значение sys.prefix устанавливается во время сборки Python с помощью аргумента --exec-prefix для скрипта configure. В частности, все файлы конфигурации (например, заголовочный файл pyconfig.h) установлены в каталоге exec\_prefix/lib/pythonX.Y/config, а модули разделяемой библиотеки установлены в exec\_prefix/lib/pythonX.Y/lib-dynload, где XY номер версии Python, например 3.7.

**sys.base\_exec\_prefix:**

Атрибут sys.exec\_prefix устанавливает при запуске Python до запуска site.py то же значение, что и sys.exec\_prefix. Если программа не работает в виртуальной среде, то значения останутся прежними. Если site.py обнаружит, что используется виртуальная среда, то значения sys.prefix и sys.exec\_prefix будут изменены на значения установки виртуальной среды, тогда как sys.base\_prefix и sys.base\_exec\_prefix останутся указывать на базовую установку Python (ту, из которой была создана виртуальная среда).

**sys.executable:**

Атрибут sys.executable это строка, задающая абсолютный путь исполняемого двоичного файла для интерпретатора Python в системах, где это имеет смысл. Если Python не может получить реальный путь к своему исполняемому файлу, sys.executable будет пустой строкой или None.

**sys.platlibdir:**

Атрибут sys.exec\_prefix это имя каталога библиотеки для конкретной платформы. Он используется для построения пути к стандартной библиотеке и путей установленных модулей расширения. (Новое в Python 3.9.)

На большинстве платформ он равен lib. В Fedora и SuSE на 64-битных платформах он равен lib64, что дает следующие пути sys.path (где X.Y - версия Python major.minor):

* /usr/lib64/pythonX.Y/: Стандартная библиотека (например, os.py модуля os),
* /usr/lib64/pythonX.Y/lib-dynload/: модули расширения языка C стандартной библиотеки (например, модуль errno, точное имя файла зависит от платформы),
* /usr/lib/pythonX.Y/site-packages/ (всегда используйте lib, а не sys.platlibdir): сторонние модули,
* /usr/lib64/pythonX.Y/site-packages/: модули расширения языка C сторонних пакетов.

Объекты stdin, stdout, stderr модуля sys в Python.

Файловые объекты интерпретатора для стандартного ввода, вывода.

Синтаксис:

import sys

sys.stdin

sys.stdout

sys.stderr

Описание:

Файловые объекты sys.stdin, sys.stdout и sys.stderr используются интерпретатором для стандартного ввода, вывода и ошибок:

* sys.stdin - используется для всех интерактивных входных данных, включая вызовы input();
* sys.stdout - используется для вывода оператором print() и выражений, которые возвращают значение, а также для подсказок input();
* sys.stderr - сообщения об ошибках и собственные запросы переводчика.

Примечание. При некоторых условиях sys.stdin, sys.stderr и sys.stdout, а также исходные значения sys.\_\_stdin\_\_, sys.\_\_stdout\_\_ и sys.\_\_stderr\_\_ могут быть None. Обычно это относится к приложениям с графическим интерфейсом Windows, которые не подключены к консоли, а приложения Python запускаются с pythonw.exe(1).

Примеры использования sys.stdout, sys.stdin:

import sys

for line in sys.stdin:

print(line.rstrip('\n')[::-1])

''''''''''''''''''''''''''''''''''''''''''''''''

import sys

sys.stdout.write("Hello, ")

sys.stdout.write("World!")

# Hello, World!

Функция exit() модуля sys в Python.

Быстрый выход из программы при возникновении ошибки.

Синтаксис:

import sys

sys.exit([arg])

Параметры:

arg - статус выхода, по умолчанию 0

Возвращаемое значение:

целое число, указывающее статус выхода.

Описание:

Функция exit() модуля sys - выход из Python. Она реализуется путем вызова исключения SystemExit, поэтому выполняются действия по очистке, указанные в предложениях finally операторов try и можно перехватить попытку выхода на внешнем уровне.

Пример:

ref = osgeo.ogr.Open(reference)

if ref is None:

raise SystemExit('Unable to open %s' % reference)

или:

ref = osgeo.ogr.Open(reference)

if ref is None:

print('Unable to open %s' % reference)

sys.exit(-1)

15.2

Примеры работы с классами в Python

<https://pythonru.com/primery/primery-raboty-s-klassami-v-python>

Задание 1. Создание классов

Чтобы определить новый класс в своей программе, необходимо напечатать ключевое слово class, а после него добавить имя для создаваемой структуры данных, завершив ввод вставкой двоеточия. Следующий пример демонстрирует генерацию пустого класса с именем Example. Как можно заметить, в нем полностью отсутствует какая-либо информация.

class Example:

pass

example = Example()

Несмотря на пустое тело класса Example, на его основе уже можно создать определенный объект, обладающий уникальным идентификатором. Последняя строка кода, находящегося выше, представляет собой пример генерации объекта с именем example и типом данных Example. Здесь используется оператор присваивания, а также пустые круглые скобки после названия класса, прямо как в вызове метода не имеющего никаких аргументов.

Определив новый класс, можно создавать сколько угодно объектов на его основе. Как уже было сказано выше, такая структура данных может включать в себя некие свойства, то есть переменные, которыми будет наделен каждый экземпляр класса. Ниже приведен простой пример класса и объекта Python 3. В примере описывается класс под названием Data со строкой word и числом number.

class Data:

word = "Python"

number = 3

data = Data()

print(data.word + " " + str(data.number))

Python 3

Если создать объект, основанный на классе Data, то он получит обе переменные, а также их значения, которые были определены изначально. Таким образом, был сгенерирован объект data. Получить доступ к его полям с именами word и number можно с помощью оператора точки, вызвав его через экземпляр класса. Функция print поможет вывести значения полей объекта data на экран. Не стоит забывать и о том, что число следует привести к строчному виду для того чтобы обработать его в методе print вместе с текстовым значением.

Помимо полей, пользовательский класс может включать в себя и методы, которыми будут наделены все его экземпляры. Вызвать выполнение определенного метода через созданный объект можно так же, как и получить доступ к его полям, то есть с помощью точки. Данный пример демонстрирует класс Data с функцией sayHello, которая выводит текст на экран.

class Data:

def sayHello(self):

print("Hello World!")

data = Data()

data.sayHello()

Hello World!

Задание 2. Создание экземпляров класса

Чтобы создать экземпляры классов, нужно вызвать класс с использованием его имени и передать аргументы, которые принимает метод \_\_init\_\_.

Задание 3. Доступ к атрибутам

Задание 4. Встроенные атрибуты класса

Задание 5. Уничтожение объектов (сбор мусора)