Лабораторная работа № 3

Вариант 13

Задание №1

Постановка задачи:

В текстовом файле находится текст. Слова отделяются друг от друга одним или несколькими пробелами и знаками препинания (“,.;:?!-“). Составить частотный словарь слов, используя для хранения пap. Слова, записанные строчными и прописными буквами в одном падеже, считаются одинаковыми. Два слова, отличающиеся окончанием, считаются разными словами.

* Вывести словарь, упорядоченный по частоте встречаемости, в новый текстовый файл.
* Удалить из словаря слова на заданную вводом с клавиатуры букву. Использовать операцию del (del <имя словаря>[<ключ>]). Результат записать в новый текстовый файл. Использовать преобразование списка ключей к типу list.

Код:

def Zapoln():

sim = ' ,.;:?!—-\n«»()'

word = ''

slov = {}

s = Fin.readline()

while s: # если есть слова в файле

for i in s:

if i in sim: # если i символ

if word in slov.keys(): # если есть слово в словаре

slov[word] += 1

elif word != '': # если нет слова в словаре

slov[word] = 1

word = ''

else: # если i буква

word += i.lower()

s = Fin.readline()

return slov

def SSort(a, n):

for i in range(1, n):

z = a[i]

j = i - 1

while j >= 0 and z >= a[j]:

a[j+1] = a[j]

j -= 1

a[j+1] = z

return a

def Chast\_Sort(slov):

n = 0 # Кол-во слов

m = [] # Частота встречаемости: список

K = [] # Список ключевых слов

for i in slov.keys():

m += [slov[i]]

K += [i]

n += 1

m = SSort(m, n) # Сортировка частоты встречаемости по убыванию

K = SSort(K, n) # Сортировка слов по убыванию

Sort\_Y = {} # Словарь отсортированный по убыванию частоты и по убыванию слов

for i in m:

for z in range(n):

for j in slov.keys():

if slov[j] == i and K[z] == j:

Sort\_Y[j] = i

return Sort\_Y

Fin = open('input.txt', "r")

Slovar = open('slovar.txt', "w")

Fout = open('output.txt', "w")

slov = Zapoln()

if not slov:

print('Файл пустой')

else:

print(slov)

slov = Chast\_Sort(slov)

print(slov)

# Занесения словаря в файл

Slovar.write("Словарь упорядоченный по убыванию частоты встречаемости:\n")

for i in slov.keys():

Slovar.write("{} - {}\n".format(i, slov[i]))

# Удаление слов на определённую букву

b = input('Введите первую букву, слова на которую надо удалить: ').lower()

a = list(slov.keys())

for i in a:

if i[0] == b:

del(slov[i])

for i in slov.keys():

Fout.write("{} - {}\n".format(i, slov[i]))

Fin.close()

Slovar.close()

Fout.close()

Тесты:

| № | Исходные данные | Ожидаемый результат | Вывод программы |
| --- | --- | --- | --- |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |

Задание №2

Постановка задачи:

Описать структуру с двумя полями – счётчик (int), слово. Создать массив структур, заполнив его данными из задания 1. Записать массив структур в бинарный файл всеми способами (по записям, весь массив целиком).

Создать новый массив структур, заполнить его данными из созданного бинарного файла. Вывести значения массива структур в новый текстовый файл.

Код:

def Zapoln():

sim = ' ,.;:?!—-\n«»()'

word = ''

slov = {}

s = Fin.readline()

while s: # если есть слова в файле

for i in s:

if i in sim: # если i символ

if word in slov.keys(): # если есть слово в словаре

slov[word] += 1

elif word != '': # если нет слова в словаре

slov[word] = 1

word = ''

else: # если i буква

word += i.lower()

s = Fin.readline()

return slov

def SSort(a, n):

for i in range(1, n):

z = a[i]

j = i - 1

while j >= 0 and z >= a[j]:

a[j+1] = a[j]

j -= 1

a[j+1] = z

return a

def Chast\_Sort(slov):

n = 0 # Кол-во слов

m = [] # Частота встречаемости: список

K = [] # Список ключевых слов

for i in slov.keys():

m += [slov[i]]

K += [i]

n += 1

m = SSort(m, n) # Сортировка частоты встречаемости по убыванию

K = SSort(K, n) # Сортировка слов по убыванию

Sort\_Y = {} # Словарь отсортированный по убыванию частоты и по убыванию слов

for i in m:

for z in range(n):

for j in slov.keys():

if slov[j] == i and K[z] == j:

Sort\_Y[j] = i

return Sort\_Y

class CU:

pass

def Files(slov, f1, f2):

import pickle

n = 0

m = []

for i in slov.keys():

n += 1

Z = CU()

Z.count = slov[i]

Z.word = i

m += [Z]

# Запись по байтам

F1 = open(f1, 'wb')

for i in range(n):

pickle.dump(m[i].count, F1)

pickle.dump(m[i].word, F1)

F1.close()

# Чтение байт

F1 = open(f1, 'rb')

m = []

for i in range(n):

word = pickle.load(F1)

count = pickle.load(F1)

Z = CU()

Z.count = count

Z.word = word

m += [Z]

F1.close()

# Запись словаря

F1 = open(f1, 'w')

for i in range(n):

F1.write("{} - {}\n".format(m[i].count, m[i].word))

F1.close()

# Запись байт для второго файла массива целиком

F2 = open(f2, 'wb')

pickle.dump(m, F2)

F2.close()

# Чтение байт второго файла

m = []

F2 = open(f2, 'rb')

m = pickle.load(F2)

F2.close()

# Запись словаря во второй файл

F2 = open(f2, 'w')

for i in range(n):

F2.write("{} - {}\n".format(m[i].count, m[i].word))

F2.close()

Fin = open('input.txt', "r")

Slovar = open('slovar.txt', "w")

Fout = open('output.txt', "w")

slov = Zapoln()

if not slov:

print('Файл пустой')

else:

print(slov)

slov = Chast\_Sort(slov)

print(slov)

# Занесения словаря в файл

Slovar.write("Словарь упорядоченный по убыванию частоты встречаемости:\n")

for i in slov.keys():

Slovar.write("{} - {}\n".format(i, slov[i]))

Files(slov, 'F1.txt', 'F2.txt')

# Удаление слов на определённую букву

b = input('Введите первую букву, слова на которую надо удалить: ').lower()

a = list(slov.keys())

for i in a:

if i[0] == b:

del(slov[i])

for i in slov.keys():

Fout.write("{} - {}\n".format(i, slov[i]))

Fin.close()

Slovar.close()

Fout.close()

Тесты:

| № | Исходные данные | Ожидаемый результат | Вывод программы |
| --- | --- | --- | --- |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |

Задание №3

Постановка задачи:

Описать новый тип МНОГОЧЛЕН (поля – степень многочлена; массив коэффициентов).

(В файле в первой строке единственным числом написана старшая степень вводимого многочлена, затем второй строкой записана последовательность 1 число - степень, второе число - коэффициент (степени с коэффициентами записываются в порядке убывания степеней x)).

Код:

class MN(): # Класс Многочлен

step = 0

coef = []

Fin = open('input.txt', "r")

s = Fin.readline().split()

if not s:

print('Файл пустой')

elif int(s[0]) == 0:

print('Многочлен задан одним числом без x')

else:

SS = int(s[0]) # Старшая степень

M = MN() # Переменная для многочлена

Fin.close()

Тесты:

| № | Исходные данные | Ожидаемый результат | Вывод программы |
| --- | --- | --- | --- |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |

Задание №4

Постановка задачи:

Создать функции, реализующие операции с многочленами: СУММА, РАЗНОСТЬ, ПРОИЗВЕДЕНИЕ, ЗНАЧЕНИЕ\_МНОГОЧЛЕНА при заданном значении аргумента.

(В файле в первой строке единственным числом написана старшая степень вводимого многочлена, затем второй строкой записана последовательность 1 число - степень, второе число - коэффициент (степени с коэффициентами записываются в порядке убывания степеней x). Второй многочлен пишется после первого и аналогично ему в 3 и 4 строках).

Код:

class MN(): # Класс Многочлен

step = 0

coef = []

def OMG(M, SS): # Инициализация

s = Fin.readline().split()

M.coef = [0] \* (SS + 1)

if SS != int(s[0]):

print('Неправильно введена старшая степень')

else:

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

M.step = int(s[(SS + 1) - i])

M.coef[M.step] = int(s[(SS + 1) - i + 1])

return M

def Fun1(M, Z, NT):

L1 = len(M.coef)

L2 = len(Z.coef)

T = True

if L1 >= L2:

NT.coef = [0] \* L1

else:

NT.coef = [0] \* L2

L1, L2 = L2, L1

T = False

for i in range(L1):

if i >= L2:

if T == True:

NT.coef[i] = M.coef[i]

else:

NT.coef[i] = Z.coef[i]

else:

NT.coef[i] = M.coef[i] + Z.coef[i]

return NT

def Fun2(M, Z, NT):

for i in range(len(Z.coef)):

Z.coef[i] \*= -1

NT = Fun1(M, Z, NT)

return NT

def Fun3(M, Z, NT):

L1 = len(M.coef)

L2 = len(Z.coef)

NT.coef = [0]\*(L1 + L2 - 1)

for i in range(L1):

for j in range(L2):

NT.coef[i+j] += M.coef[i]\*Z.coef[j]

return NT

def Fun4(M):

N = int(input('Введите значение X: '))

Z = 0

for i in range(len(M.coef)):

Z += N\*\*(i)\*M.coef[i]

return Z

def VV(M): # Ввод многочлена в строку для вывода

s = ''

L = len(M.coef)

for i in range(L - 1, -1, -1):

if M.coef[i] < 0 and s == '':

s += '-'

if s != '' and s[len(s) - 1] != '+' and s[len(s) - 1] != '-':

if M.coef[i] > 0:

s += '+'

elif M.coef[i] < 0:

s += '-'

if i == 0 and M.coef[i] != 0:

s += str(abs(M.coef[i]))

else:

if M.coef[i] == 1 or M.coef[i] == -1:

if i == 1:

s += 'x'

else:

s += 'x^(' + str(i) + ')'

elif M.coef[i] != 0:

if i == 1:

s += str(abs(M.coef[i])) + '\*' + 'x'

else:

s += str(abs(M.coef[i])) + '\*' + 'x^(' + str(i) + ')'

return s

Fin = open('input.txt', "r")

Fout = open('output.txt', "w+")

s = Fin.readline().split()

if not s:

print('Файл пустой')

Fout.write('Файл пустой\n')

elif int(s[0]) == 0:

print('Многочлен задан одним числом без x')

Fout.write('Многочлен задан одним числом без x\n')

else:

SS = int(s[0]) # Старшая степень

M = MN() # Переменная для первого многочлена

M = OMG(M, SS) # Занесение данных многочлена в переменную

if M.coef == [0] \* (SS + 1):

print('Неправильный ввод. Выполнение операций невозможно.')

Fout.write('Неправильный ввод. Выполнение операций невозможно.\n')

else:

print('Какой тип операции выполнить?')

print('1) Сложение')

print('2) Вычитание')

print('3) Произведение')

print('4) Значение многочлена')

Fout.write('Какой тип операции выполнить?\n')

Fout.write('1) Сложение\n')

Fout.write('2) Вычитание\n')

Fout.write('3) Произведение\n')

Fout.write('4) Значение многочлена\n')

N = int(input('Введите число операции: '))

Fout.write("Число операции: {}\n".format(N))

NT = MN() # Переменная для итога

if N == 1:

s = Fin.readline().split()

if not s:

print('Нет второго многочлена в файле')

Fout.write('Нет второго многочлена в файле\n')

elif int(s[0]) == 0:

print('Многочлен задан одним числом без x')

Fout.write('Многочлен задан одним числом без x\n')

else:

SS = int(s[0])

Z = MN() # Переменная для второго многочлена

Z = OMG(Z, SS)

print('(', VV(M), ') + (', VV(Z), ')', sep='')

Fout.write('({}) + ({})\n'.format(VV(M),VV(Z)))

NT = Fun1(M, Z, NT)

print(VV(NT))

Fout.write('Результат операции: {}\n'.format(VV(NT)))

elif N == 2:

s = Fin.readline().split()

if not s:

print('Нет второго многочлена в файле')

Fout.write('Нет второго многочлена в файле\n')

elif int(s[0]) == 0:

print('Многочлен задан одним числом без x')

Fout.write('Многочлен задан одним числом без x\n')

else:

SS = int(s[0])

Z = MN() # Переменная для второго многочлена

Z = OMG(Z, SS)

print('(', VV(M), ') - (', VV(Z), ')', sep='')

Fout.write('({}) - ({})\n'.format(VV(M), VV(Z)))

NT = Fun2(M, Z, NT)

print(VV(NT))

Fout.write('Результат операции: {}\n'.format(VV(NT)))

elif N == 3:

s = Fin.readline().split()

if not s:

print('Нет второго многочлена в файле')

Fout.write('Нет второго многочлена в файле\n')

elif int(s[0]) == 0:

print('Многочлен задан одним числом без x')

Fout.write('Многочлен задан одним числом без x\n')

else:

SS = int(s[0])

Z = MN() # Переменная для второго многочлена

Z = OMG(Z, SS)

print('(', VV(M), ') \* (', VV(Z), ')', sep='')

Fout.write('({}) \* ({})\n'.format(VV(M), VV(Z)))

NT = Fun3(M, Z, NT)

print(VV(NT))

Fout.write('Результат операции: {}\n'.format(VV(NT)))

elif N == 4:

print(VV(M), sep='')

Fout.write('{}\n'.format(VV(M)))

Z = Fun4(M)

print(Z)

Fout.write('Результат операции: {}\n'.format(Z))

else:

print('Неправильно введены данные операции')

Fout.write('Неправильно введены данные операции\n')

Fin.close()

Fout.close()

Тесты:

| № | Исходные данные | Операция | Ожидаемый результат | Вывод программы |
| --- | --- | --- | --- | --- |
| 1 | [Файл](https://drive.google.com/file/d/1R3Xj0UfLk1zPIsFT6OIDfcS_M_V8x8oa/view?usp=sharing) | 1 | -x^(2)+9\*x-20 | -x^(2)+9\*x-20 |
| 2 | 2 | -x^(2)-x-12 | -x^(2)-x-12 |
| 3 | 3 | -5\*x^(3)+24\*x^(2)-96\*x+64 | -5\*x^(3)+24\*x^(2)-96\*x+64 |
| 4 | 4 (X = 1) | -13 | -13 |

Задание №5

Постановка задачи:

Инициализация МНОГОЧЛЕНА и вывод на экран и в текстовый файл.

(В файле в первой строке единственным числом написана старшая степень вводимого многочлена, затем второй строкой записана последовательность 1 число - степень, второе число - коэффициент (степени с коэффициентами записываются в порядке убывания степеней x)).

Код:

class MN(): # Класс Многочлен

step = 0

coef = []

def OMG(M, SS): # Инициализация

s = Fin.readline().split()

M.coef =[0]\*(SS+1)

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

M.step = int(s[(SS+1)-i])

M.coef[M.step] = int(s[(SS+1)-i+1])

return M

def VV(M): # Ввод многочлена в строку для вывода

s = ''

L = len(M.coef)

for i in range(L-1, -1, -1):

if M.coef[i]<0 and s == '':

s += '-'

if s != '' and s[len(s)-1] != '+' and s[len(s)-1] != '-':

if M.coef[i] > 0:

s += '+'

elif M.coef[i] < 0:

s += '-'

if i == 0 and M.coef[i] != 0:

s += str(abs(M.coef[i]))

else:

if M.coef[i] == 1 or M.coef[i] == -1:

if i == 1:

s += 'x'

else:

s += 'x^('+str(i)+')'

elif M.coef[i] != 0:

if i == 1:

s += str(abs(M.coef[i])) + '\*' + 'x'

else:

s += str(abs(M.coef[i])) + '\*' + 'x^(' + str(i) + ')'

return s

Fin = open('input.txt', "r")

Fout = open('output.txt', "w+")

s = Fin.readline().split()

if not s:

print('Файл пустой')

elif int(s[0]) == 0:

print('Многочлен задан одним числом без x')

else:

SS = int(s[0]) # Старшая степень

M = MN() # Переменная для многочлена

M = OMG(M, SS) # Занесение данных многочлена в переменную

s = VV(M) # Заносим запись многочлена в строку

print('Многочлен: ', s)

Fout.write("Многочлен: {}\n".format(s))

Fin.close()

Fout.close()

Тесты:

| № | Исходные данные | Ожидаемый результат | Вывод программы |
| --- | --- | --- | --- |
| 1 | [Файл](https://drive.google.com/file/d/1R3Xj0UfLk1zPIsFT6OIDfcS_M_V8x8oa/view?usp=sharing) | Многочлен: -x^(2)+4\*x-16 | Многочлен: -x^(2)+4\*x-16 |
| 2 |  |  |  |
| 3 |  |  |  |