Министерство образования Республики Беларусь

Белорусский Национальный Технический Университет

Факультет информационных технологий и робототехники

Кафедра «Программное обеспечение вычислительной техники

и автоматизированных систем»

**Отчёт**

по лабораторной работе № 9

по дисциплине ***«Языки программирования»***

тема:**« Кортежи, списки, словари и множества»**

|  |  |  |
| --- | --- | --- |
| Исполнитель: |  | студент группы 10701118  Воробей И.А. |
| Преподаватель: |  | Белова Светлана Владимировна |

2018-2019 учебный год

Лабораторная работа № 9

«**Кортежи, списки, словари и множества»**

**Цель работы**: приобрести навыки работы со стандартными встроенными высокоуровневыми типами данных в Python и закрепить их на примере разработки интерактивных приложений.

**Основное** **задание**

Разработать интерактивную программу, которая будет моделировать игру «Виселица» («Hangman»). Компьютер загадывает слово и выводит на консоль количество подчёркиваний, равное числу букв в загаданном слове. Пользователь начинает вводить буквы, чтобы отгадать слово. Если буква есть в слове, компьютер вписывает её на своё место в слово, а если нет, компьютер рисует в консоли один элемент импровизированной виселицы (к примеру: стойка, перекладина, верёвка, голова, туловище, две руки, две ноги). Дополнительно выводятся буквы, которые уже вводились.

Если игрок не успел угадать слово раньше, чем компьютер нарисовал полностью виселицу, то игрок считается проигравшим. Если игрок успевает угадать слово, то он выигрывает.

**Индивидуальное** **задание**

В соответствии с заданием своего варианта выполнить задания из Приложения «D» «Списки» и «E» «Матрицы».

**Дополнительное задание**

Модифицировать программу «Виселица» таким образом, чтобы в ней были уровни сложности и (или) подсказки.

**Требования к выполнению**

1. Каждое задание оформить в виде отдельной бизнес-функции. Все функции необходимо сгруппировать в одном модуле или разнести по отдельным модулям, согласно их логике их работы;

2. Размерность списка и матрицы задаётся пользователем;

3. Предусмотреть два способа инициализации последовательности и матрицы: с помощью генератора случайных чисел и пользовательского ввода. Оформить способы инициализации в виде отдельных функций, которые на вход принимают последовательность или матрицу для инициализации, и сгруппировать эти функции в отдельный модуль от основной функции программы;

4. В программах предусмотреть возможность повторного выполнения, и защиту от ввода некорректных пользовательских данных. Для этих целей рекомендуется разработать отдельные функции;

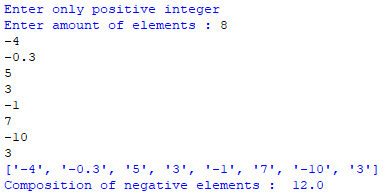
5. При разработке программ рекомендуется придерживаться принципа единственной ответственности;

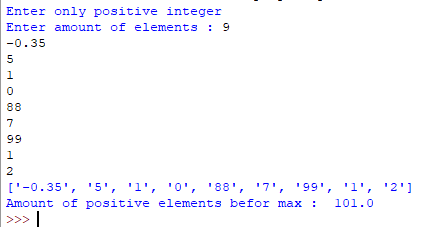
6. Программа должна обязательно быть снабжена комментариями, в которых необходимо указать краткое предназначение программы, номер лабораторной работы и её название, версию программы, ФИО разработчика и дату разработки;

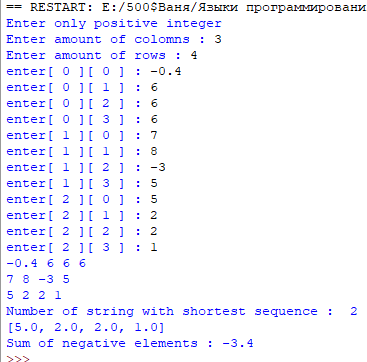
7. Программа должна быть снабжена дружелюбным и интуитивно понятным интерфейсом.

**Результаты выполнения основного задания**

**Результаты выполнения индивидуального задания**

****

****

****

**Выводы**

1. Янаучился делать операции над матрицами.
2. Научился находить максимальный и минимальный элемент в массиве.
3. Я рассмотрел функции для работы со списками.
4. Научился находить последовательности элементов в списках.

**Контрольные** **вопросы**

1. Привести примеры объявления каждого из высокоуровневых типов данных.

Список s=[]

Кортеж k=()

Словарь di={}

Множество m=set()

Пустое множество с помощью литерала создать нельзя.

2. Объяснить понятие «распаковка последовательности».

Разделение списка или кортежа в другие переменные.

3. Какое главное отличие списков от кортежей? Когда лучше использовать кортежи, а когда – списки? Списки изменяемы в отличие от кортежей, а так же кортежи занимают меньше памяти.

4. Что такое распределённые ссылки?

5. Как получить несколько ссылочных переменных на один и тот же список, а как получить полную копию списка?

list.copy()

L1=[1,2]

l2=[]

for I in range(len(l1))

l2[i]=l1[i]

6. Зачем нужна индексация элементов высокоуровневых типов? Как её использовать?

Для обращения к каждому элементу. Проход по списку.

7. Как получить доступ к элементам словаря? Можно ли использовать индексацию для словарей?

d={1:2,2:4,3:9}

d[1] /// 2

d[4]= 4\*\*2

d////{1:2,2:4,3:9, 4:16}

8. Зачем и как используются срезы?

s[START:STOP:STEP]-берет сре от номера START,до STOP не включая его с шагом STEP. По умолчанию шаг =1

a=[1,3,8,7]

a[:] ////[1,3,8,7]

a[1:] ////[3,8,7]

a[:3] //////[1,3,8]

a[::2] /////[1,8]

9. Какие операторы и встроенные функции используются для работы с кортежами, списками, словарями и множествами? Привести сравнительную таблицу.

Списки : list([]),len(s)

Словари :dict([])

Кортежи :tuple(obj)

Множества: set([]),frozenset([])

10. Какие методы есть у каждого из классов, отвечающих за каждый высокоуровневый тип данных (tuple, list, dict и set)? Описать наиболее востребованные.

Списки : list.append(x),list.extend(L),list.count(x),list.clear()

Словари :dict.clear(),dict.copy(),dict.items()//пары,dict.keys()

Кортежи : тоже что и для списков

Множества:set.copy(),set.add(),set.clear()

**Листинг основных программ:**

import random

HANGMANPICS = ['''

+---+

| |

|

|

|

|

=========''','''

+---+

| |

O |

|

|

|

=========''','''

+---+

| |

O |

| |

|

|

=========''','''

+---+

| |

O |

/| |

|

|

=========''','''

+---+

| |

O |

/|\ |

|

|

=========''','''

+---+

| |

O |

/|\ |

/ |

|

=========''','''

+---+

| |

O |

/|\ |

/ \ |

|

=========''']

def getRandomWord(wordList):

wordIndex = random.randint(0, len(wordList) - 1)

return wordList[wordIndex]

def getGuess(alreadyGuessed):

#Возвращает букву, которую ввел игрок. Эта функция проверяет, что введена

# буква, а не какой-то другой символ

while True:

print('Введите букву')

guess = input()

guess = guess.lower()

if len(guess) != 1:

print('Вводите по одной букве.')

elif guess in alreadyGuessed:

print ('Вы уже пробовали эту букву. Выберите другую')

elif guess not in 'ёйцукенгшщзхъфывапролджэячсмитьбю':

print('Пожалуйста, введите букву кириллицы')

else:

return guess

def displayBoard(HANGMANPICS, missedletters, correctLetters,secretWord):

print(HANGMANPICS[len(missedLetters)])

print()

print('Неправильные буквы:', end=' ')

for letter in missedLetters:

print(letter, end=' ')

print()

blanks = '\_'\*len(secretWord)

#Заменяем звездочки на правильно угаданные буквы

for i in range(len(secretWord)):

if secretWord[i] in correctLetters:

blanks = blanks[:i] + secretWord[i] + blanks[i+1:]

#Показываем загаданное слово с пробелами между букв

for letter in blanks:

print(letter, end=' ')

print()

def playAgain():

print('Хотите попробовать еще раз? ("Да" или "Нет")')

return input().lower().startswith('д')

easy = '''лук суп сок бок шум шов ток тук бук куш'''.split()

medium = '''слива забор город тема вечер змея указ'''.split()

hard = '''калейдоскоп компьютер аутентификация аутофагия'''.split()

#print(words)

words=''' '''

print('В И С Е Л И Ц А')

fl=0

while True:

fl=int(input("Введите уровень сложности: 1-easy, 2-medium, 3-HARD : "))

if fl==1:

words=easy

break

elif fl==2:

words=medium

break

elif fl==3:

words=hard

break

else:

print("Введите число от 1 до 3")

missedLetters = ''

correctLetters = ''

secretWord = getRandomWord(words)

print(secretWord)

gameIsDone = False

while True:

displayBoard(HANGMANPICS, missedLetters, correctLetters, secretWord)

guess = getGuess(missedLetters+correctLetters)

if guess in secretWord:

correctLetters = correctLetters + guess

#Проверка условия победы игрокая

foundAllLetters = True

for i in range(len(secretWord)):

if secretWord[i] not in correctLetters:

foundAllLetters = False

break

if foundAllLetters:

print('Превосходно! Было загадано слово "'+secretWord+'" Вы победили!')

#плюсы убирают пробелы

gameIsDone = True

else:

missedLetters = missedLetters+guess

#Проверка условия проигрыша игрока

if len(missedLetters) == len(HANGMANPICS) - 1:

displayBoard(HANGMANPICS, missedLetters, correctLetters, secretWord)

print('У вас не осталось попыток!\nПосле '+str(len(missedLetters))+' ошибок и '+str(len(correctLetters))+'угаданных букв. Загаданное слово:'+secretWord+'"')

gameIsDone = True

if gameIsDone:

if playAgain():

missedLetters = ''

correctLetters = ''

gameIsDone = False

secretWord = getRandomWord(words)

else:

break

#application "D" task1

#composition of negative elements

def enterl(sp):

sp=list()

while True:

print("Enter only positive integer")

k=int(input("Enter amount of elements : "))

if k>0:break

i=0

while True:

sp.append(input())

i+=1

if i==k:break

return sp,k

composition=1

s=()

s,n=enterl(s)

print(s)

s1=[float(item) for item in s]

i=0

for i in range(n):

if s1[i]<0:

flag=1

composition=composition\*s1[i]

if flag==1:

print("Composition of negative elements : ",composition)

else:

print("There are no negative itims in the list.")

#application "D" task2

#Amount of positive elements befor max

def enterl(sp):

sp=list()

while True:

print("Enter only positive integer")

k=int(input("Enter amount of elements : "))

if k>0:break

i=0

while True:

sp.append(input())

i+=1

if i==k:break

return sp,k

suma=0

s=()

s,n=enterl(s)

print(s)

s1=[float(item) for item in s]

i=0

i1=int

while True:

if s1[i]==max(s1):

i1=i

break

i+=1

for i in range(i1):

if s1[i]>0:

suma=suma+s1[i]

print("Amount of positive elements befor max : ",suma)

#application "E" task1,2 fanctions!

def input1(mx):

while True:

print("Enter only positive integer")

n=int(input("Enter amount of colomns : "))

m=int(input("Enter amount of rows : "))

if n>0 and m>0:break

for i in range(n):

mx.append([])

for j in range(m):

print("enter[",i,"][",j,"] :", end=' ')

mx[i].append(input())

return mx,n,m

def output(mtr,n,m):

for i in range(n):

for j in range(m):

print(mx[i][j], end=' ')

print()

#application "E" task1,2

#lab9 individual

mx=[]

mx, n,m=input1(mx)

output(mx,n,m)

for i in range(n):

mx[i]=[float(item) for item in mx[i]]

for i in range(n):

k=0

for j in range(m):

if mx[i][j]==mx[i][j-1] and j!=0:

k+=1

if k>0:

mi=k

n1=i

break

for i in range(n):

k=0

flag=0

for j in range(m):

if mx[i][j]==mx[i][j-1] and j!=0:

k+=1

flag=1

if k<mi and flag==1:

mi=k

n1=i

print("Number of string : ",n1)

print(mx[n1])

#sum of negative elements

sm=0

for i in range(n):

for j in range(m):

if mx[i][j]<0:

sm=sm+mx[i][j]

print("Sum of negative elements :", sm)