Федеральное государственное образовательное бюджетное

учреждение высшего образования

«**Финансовый университет**

**при Правительстве Российской Федерации»**

**(Финансовый университет)**

**Департамент анализа данных, принятия решений и финансовых технологий**

**Отчёт по семинару №2**

**Выполнил:**

Студент Малкеров Г.А.

Группы ЗБ-ПИ1-2

**Проверил:**

Петросов Д.А.

**Москва 2019**

**Задание:**

Проработать навыки работы со строками на языке Python и выполнить задание:

Реализовать одно алфавитный шифр Цезаря для шифрования и дешифрование строки любой длины и заданным ключем (сдвигом алфавита), используется кириллический алфавит, знаки препинания, цифры, верхний и нижний регистр.

Программный код на языке Python для тестовых заданий:

print("\nПример использования escape симоволов")

# Пример использования escape симоволов

# \n - перевод на новую строку (запомним как new)

# \t - табуляция горизонтальная (так и запомним Tабуляция)

# \v - табуляция вертикальная (так и запоминаем Veртикальная)

# \r - вернет корретку

s1 = "Привет!\nТы\t\vизучаешь\nPython"

print(s1)

print("\nОтключение использование escape симоволов")

# Отключение использование escape симоволов

s2\_1="C:\tath\vath"

s2\_2=r"C:\tath\vath"

print(s2\_1)

print(s2\_2)

print ("\nКонкатенация (сложение строк)")

# Конкатенация (сложение строк)

S3\_1 = 'Hello' # строка 1

S3\_2 = ' '# строка 2

S3\_3= 'world' # строка 3

S3\_4= S3\_1+S3\_2+S3\_3 # итоговая строка

print (S3\_4)# вывод строки на экран

#Повторение строки

print("\nПовторение строки")

s4\_1="Текст "

print(s4\_1)

s4\_1\*=3

print(s4\_1)

# Обращение по индексу в строке

print ("\n Обращение по индексу в строке")

s5\_1="Короче надоело просто копировать поэтому думаем"

for i in range(len(s5\_1)-1):

print(s5\_1[i],end="\t")

print("Измерим длину строки")

s6\_1="Короче надоело просто копировать поэтому думаем"

sizeOfStr=len(s6\_1)

print(sizeOfStr)

print("Строка ["+ s6\_1+ "] имееет длину ",sizeOfStr)

print("Так сказать replace")

s7\_1="измерим что-то"

print(s7\_1)

print(s7\_1.replace("р","н"))

print("Немного split")

s8\_1 = "Коровы овцы лошади собаки"

print (s8\_1.split(' '))

print (s8\_1.split('о'))

print (s8\_1.split('о', 2))

print (s8\_1.split('о', 4))

S1 = '34567o'

print(S1.isdigit())

S1 = '34567'

print(S1.isdigit())

print("удаление пробелов в строке")

S1 = ' 1 корова ! '

k1 = len(S1)

print('Начальная строка:',S1, 'всего символов - ', k1)

S2= S1.strip() # удадим пробелы в начале и конце строки

k2 = len(S2)

print ('Удалил пробелы в начале и конце строки результат:',S2,'осталось символов - ', k2)

S1 = ' !"1 корова " !'

k1 = len(S1)

print('Начальная строка:',S1, 'всего символов - ',k1)

S2 = S1.lstrip()

k2 = len(S2)

print ('Удалил пробелы в начале строки результат:',S2,'осталось символов - ', k2)

S1 = ' !"1 корова" '

k1 = len(S1)

print('Начальная строка:', S1, 'всего символов - ', k1)

S2 = S1.rstrip()

k2 = len(S2)

print ('Удалил пробелы в конце строки результат:', S2, 'осталось символов - ', k2)

S1 = ' ! !"1 корова" ! \* '

k1 = len(S1)

print('Начальная строка:', S1, 'всего символов - ', k1)

S2 = S1.replace(' ','')

k2 = len(S2)

print ('Удалил все пробелы результат:', S2, 'осталось символов - ', k2)

print("\nРабота с высотой букв")

# проверим состоит есть ли в строке неотображаемых символов

S1 = 'Привет мир'

print(S1.upper())

print(S1.lower())

print (S1)

print(S1.isspace())

S1 = ' '

print (S1)

print(S1.isspace())

S1 = '\n'

print (S1)

print(S1.isspace())

# проверим состоит ли строка из символов в нижнем регистре

S1 = 'привет мир'

print (S1)

print(S1.islower())

S1 = 'Привет мир'

print (S1)

print(S1.islower())

# проверим состоит ли строка из символов в верхнем регистре

S1 = 'Привет мир'

print (S1)

print(S1.isupper())

S1 = 'ПРИВЕТ МИР'

print (S1)

print(S1.isupper())

# проверим начинается ли строка с символа в верхнем регистре

S1 = 'привет мир' # все слова в строке с нижнего регистра результат False

print (S1)

print(S1.istitle())

S1 = 'Привет мир' # только первое слово в строке с верхнего регистра результат False

print (S1)

print(S1.istitle())

S1 = 'Привет Мир'

print (S1)

print(S1.istitle()) # оба слова с верхнего регистра результат True

S1 = '-'

posl = ('Быстрее','Выше','Сильнее')

print (S1.join(posl))

S1 = '0123456789'

print('Строка для среза: ',S1)

print ('<-------------------------->')

# выполнеим срез начиная с первого элемента по последний с шагом 2, т.е. каждый второйэлемент

# синтаксис "строка[начало среза:конец среза:шаг]". 0 - начало строки, 17 последний индекс

print ('Срез с 1 элемента по 6 с шагом 2', S1[1:6:2])

print ('Срез с первого по последний элемент с шагом 2', S1[::2])

# выплним срез начиная со второго элемента с шагом 2, не забываем, что индексация в строке начинается с 0,

# т.е. второй элемент имеет индекс 1

print ('Срез без указания шага во всей строке', S1[::])

print ('Срез без указания шага во всей строке', S1[-10:0:])

print ('Срез со второго по элемент с шагом 2',S1[1:3:2])

print (S1[1::2])

S1 = 'Пусть наступит новый год, он нам радость принесет!!!!'

print(S1[6:14:1])

print (S1[6:-39:1])

print (S1[-40:-48:-1])

print(S1[-40:5:-1])

print (S1[::3])

print(S1[::-2])

Программный код на языке Python для дополнительных заданий:

import traceback

al = " АБВГДЕЁЖЗИЙКЛМНОПРСТУФЧХЦШЩЪЫЬЭЮЯабвгдеёжзийклмнопрстуфхчцшщъыьэюя0123456789.,;?:!()-|\"«»\'"

def CheckValueAndReturnNormal(inputValue):

if inputValue in al:

return al.index(inputValue)+1

else:

aistr = al.join(",")

raise Exception("Индекс не находится в массиве для работы для элемента [{0}].\n Обратитесь к разрабам чтобы его добавить если это знак препинания или иной символ который должен тут быть по заданию.\n В настоящий момент имеются следующие данные\n [{1}]".format(

inputValue, aistr))

def mainEncryptSoulition(inputString, step):

resultStr = ""

for inpuInfo in inputString:

oldIndexNormal = CheckValueAndReturnNormal(inpuInfo)

newIndex = oldIndexNormal+step-1

if newIndex > len(al):

newIndex = newIndex-len(al)

resultStr += al[newIndex]

return resultStr

def mainDecryptSoulition(inputString, step):

resultStr = ""

for inpuInfo in inputString:

oldIndexNormal = CheckValueAndReturnNormal(inpuInfo)

newIndex = oldIndexNormal-step-1

if newIndex > len(al):

newIndex = newIndex-len(al)

resultStr += al[newIndex]

return resultStr

def checkStep(step):

if not step.isdigit():

raise Exception("Введенное значение не является числом", step)

step = int(step)

lenOfal = len(al)

if step > lenOfal:

print("Длина шага больше длины алфавита, поэтому просто передвигаем на определенное количество символов ", lenOfal, step)

step=step%26

print("Итоговый шаг ", step)

return int(step)

def main():

try:

step = checkStep(

input("Введите шаг (больше {0} не имеет смысла): ".format(len(al))))

typeOfWork = input(

"Тип работы:\nШифрование - 1\nДешифровка - 2 \nИ туда и сюда - 3\n")

print("Шаг для работы {0}".format(step))

if int(typeOfWork) == 1:

print(mainEncryptSoulition(

input("Введите данные для шифрования\n"), step))

elif int(typeOfWork) == 2:

print(mainDecryptSoulition(

input("Введите данные для дешифрованния\n"), step))

elif int(typeOfWork) == 3:

encriptStr = mainEncryptSoulition(

input("Введите данные для шифрования\n"), step)

decrStr = mainDecryptSoulition(encriptStr, step)

print("Зашифрованная строка\n{0}\nРасшифрованная строка\n{1}\n".format(

encriptStr, decrStr))

else:

print("Ничего не выбрано")

print("Конец работы программы")

except Exception:

print("Ошибка при работе программы ", traceback.format\_exc())

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Пример работы программного:**

Пример работы программного кода из примерных заданий представлен на рисунке 1-2.

Пример работы программного кода из задания представлен на рисунке 3.

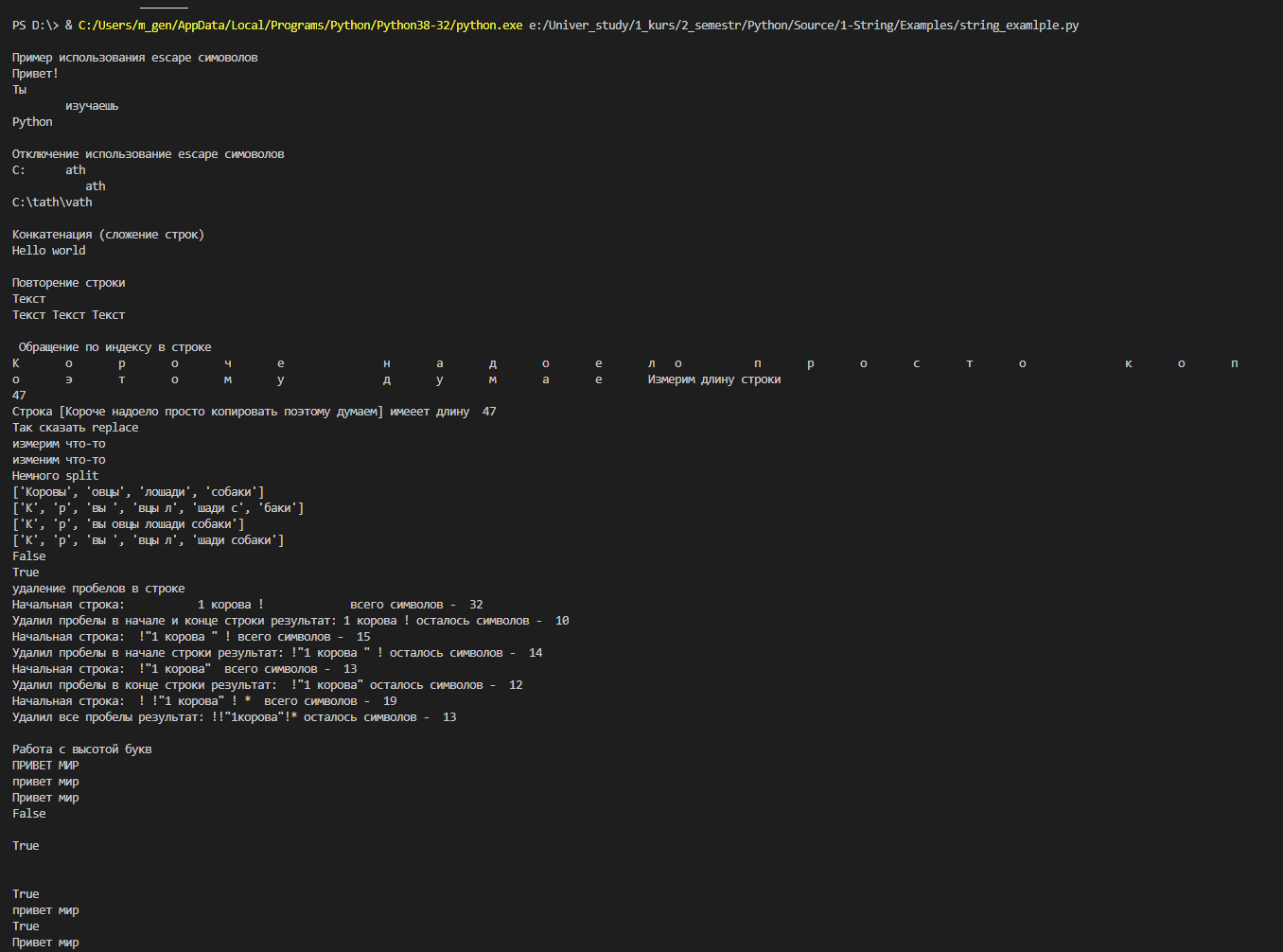


Рисунок 1 – Пример работы программного кода из тестовых примеров

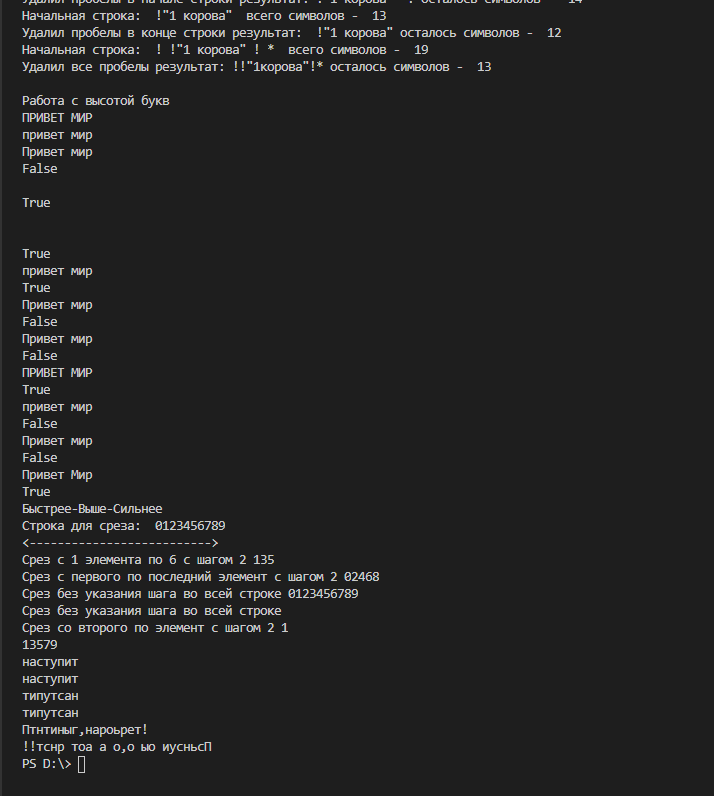


Рисунок 2– Пример работы программного кода из тестовых примеров

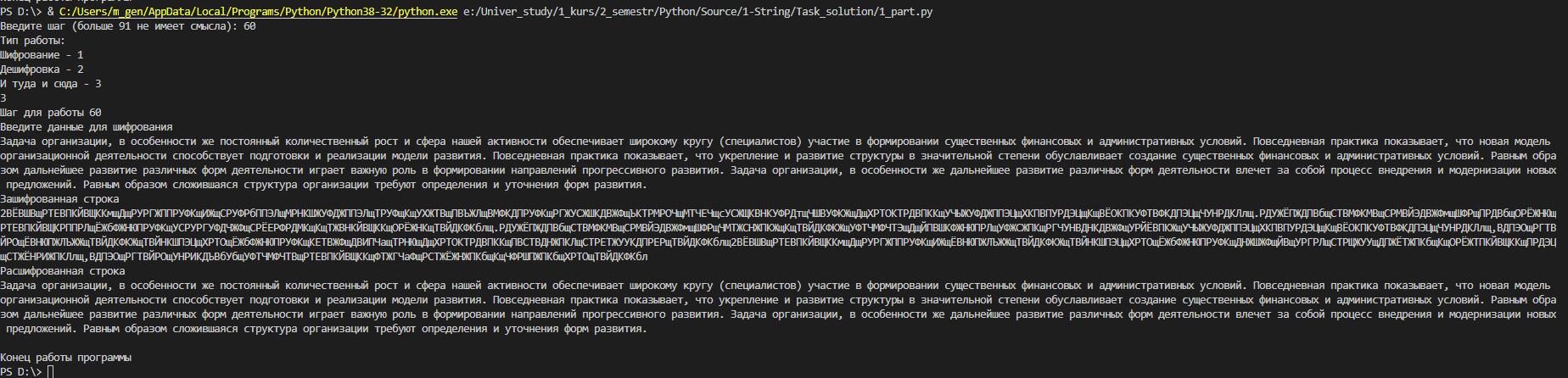


Рисунок 3 – Пример работы программного кода из основного задания

**Вывод:** В ходе выполнения данной практической работы были получены навыки работы со строками в языке программирования Python