Лабораторная работа №2
Работа со строками и строковыми методами

Цель работы: научиться разрабатывать алгоритмы и программы по обработке строк, находить оптимальные решения задач, содержащих строковые методы

Задание на лабораторную работу:

1. Изучить теоретические сведения.
2. Написать программу в соответствии с вариантом.

Теоретические сведения

Все строки, используемые в программах, являются объектами класса `str`. Чтобы посмотреть весь список методов, выполните `help(str)`.

Для работы со строками поддерживаются *операции выражений*, такие как конкатенация (объединение строк), выделение подстроки, выборка символов по индексам (по смещению от начала строки) и так далее. Помимо выражений язык Python предоставляет ряд *строковых методов*, которые реализуют обычные задачи работы со строками, а также *модули* для решения более сложных задач обработки текста, таких как поиск по шаблону.

Операция	**Интерпретация**
S = ‘’	Пустая строка
S = “spam’s”	Строка в кавычках
S = ‘s\np\ta\x00m’	Экранированные последовательности
S1 + S2	Конкатенация
S * 3	Повторение
S[i]	Обращение к символу по индексу
len(S)	Длина
“a %s parrot” % kind	Выражение форматирования строки
“a {0} parrot”.format(kind)	Строковый метод форматирования
S.find(‘pa’)	Вызов строкового метода: поиск
S.rstrip()	Удаление ведущих и конечных пробельных символов
S.rstrip(символ) S.lstrip(символ)	Удаление указанного символа справа в строке Удаление указанного символа слева в строке
S.replace(‘pa’, ‘xx’)	Замена
S.split(‘,’)	Разбиение по символу-разделителю
S.isdigit()	Проверка содержимого
S.lower()	Преобразование регистра символов
S.upper()	Преобразование регистра символов
S.isupper() S.islower()	Возвращает true, если в строке хотя бы одна буква в верхнем (или нижнем) регистре
S.endswith(‘spam’)	Проверка окончания строки
‘spam’.join(strlist)	Сборка строки из списка
S.encode(‘latin-1’)	Кодирование строк Юникода
S.count(T)	Возвращает число вхождений строки T внутри строки S

Примеры использования строковых методов (класса str):

```
name   =  'New string'     # Это  объект строки

# начинается ли строка с некоторой заданной подстроки?
if name.startswith('Ne'):
    print('Да,  строка  начинается на "Ne"')

# является ли некоторая строка частью данной строки?
```

```
if 'i'  in  name:
    print('Да,  строка  содержит букву "i"')

#метод find используется для определения позиции данной подстроки в строке;
#возвращает -1, если подстрока не обнаружена.
if name.find('ing')  != -1:
    print('Да,  она   содержит строку  "ing"')

# метод join - для объединения элементов последовательности с указанной
#строкой в качестве разделителя между элементами
delimiter =  '_*_'
mylist = ['Бразилия', 'Россия', 'Индия', 'Китай']
print(delimiter.join(mylist))

# еще один пример с join: склеить три строки
>>> names = ["John", "Paul", "Ringo", "George"]
>>> ", ".join(names)
'John, Paul, Ringo, George'

# метод replace заменяет подстроку
S = 'spammy'
S = S.replace('mm', 'xx')
print(S)                    # получим 'spaxxy'

# метод split преобразует строку в список подстрок, окружающих
строки-разделители
line = 'bob,hacker,40'
line.split(',')             # получим ['bob', 'hacker', '40'], разделитель
запятая
#Разделители могут содержать более одного символа:
line = "i'mSPAMaSPAMprogrammer"
line.split("SPAM")          # получим ["i'm", 'a', 'programmer']

>>> 'My name is Simon'.split()
['My', 'name', 'is', 'Simon']
```

Если необходимо заменить одну подстроку фиксированного размера, которая может появиться в любом месте, можно также выполнить операцию замены или отыскать подстроку с помощью метода find и затем воспользоваться операциями извлечения подстроки:

```
>>> S = 'xxxxSPAMxxxxSPAMxxxx'
>>> where = S.find('SPAM')       # Поиск позиции
>>> where                        # Подстрока найдена со смещением 4
4
>>> S = S[:where] + 'EGGS' + S[(where+4):]
>>> S
'xxxxEGGSxxxxSPAMxxxx'
```

Если нужно производить множество изменений длинных строк, вы можете повысить производительность программы, преобразовав строку в объект, который допускает внесение изменений:

```
>>> S = 'spammy'
>>> L = list(S)
>>> L
['s', 'p', 'a', 'm', 'm', 'y']
```

Встроенная функция list (или функция-конструктор объекта) создает новый список из элементов любой последовательности – в данном случае «разрывая» строку на символы и формируя из них список, элементы которого можно изменять:

```
>>> L[3] = 'x'              # Этот прием допустим для списков, но не для
строк
```

```
>>> L[4] = 'x'
>>> L
['s', 'p', 'a', 'x', 'x', 'y']
```

Если после внесения изменений необходимо выполнить обратное преобразование (чтобы, например, записать результат в файл), можно использовать метод join, который «собирает» список обратно в строку:

```
>>> S = ''.join(L)
>>> S
'spaxxy'
```

Пример обработки строк: с клавиатуры вводится строка, содержащая имя, отчество и фамилию человека, например: Василий Алибабаевич Хрюндиков. Каждые два слова разделены одним пробелом, в начале строки пробелов нет. В результате обработки должна получиться новая строка, содержащая фамилию и инициалы: Хрюндиков В.А.

```
print ( "Введите имя, отчество и фамилию:" )
s = input()
fio = s.split()
s = fio[2] + " " + fio[0][0] + "." + fio[1][0] + "."
print ( s )
```

Запись fio[0][0] обозначает «0-й символ из 0-го элемента списка fio», то есть, первая буква имени. Аналогично fio[1][0] – это первая буква отчества.

Строковые методы is удобно применять для проверки допустимости введенных пользователем данных:

- `isalpha()` возвращает значение `True`, если строка состоит только из букв и не является пустой.
- `isalnum()` возвращает значение `True`, если строка состоит только из буквенно-цифровых символов и не является пустой.
- `isdecimal()` возвращает значение `True`, если строка состоит только из цифровых символов и не является пустой.
- `isspace()` возвращает значение `True`, если строка состоит только из символов пробела, табуляции, новой строки и не является пустой.
- `istitle()` возвращает значение `True`, если строка состоит только из слов, которые начинаются с буквы в верхнем регистре, за которой следуют только буквы в нижнем регистре.

```
Пример: проверка пароля, содержащего только буквы и цифры
password = input('Введите новый пароль (только буквы и цифры)')
print(password.isalnum())
```

Смещения и срезы: положительные смещения отсчитываются от левого конца (первый элемент имеет смещение 0), а отрицательные отсчитываются от правого конца (последний элемент имеет смещение –1).

[начало:конец]
Индексы указывают позиции, в которых будут выполнены «разрезы»

По умолчанию используются начало и конец последовательности

Значение смещения слева от двоеточия обозначает левую границу (*включительно*), а справа – верхнюю границу (*она не входит в срез*).

Выражение	Что делает
S[1:3]	извлекает элементы со смещениями от 1 до 3 (не включая элемент со смещением 3)
S[1:]	извлекает элементы, начиная со смещения 1 и до конца (длина последовательности)
S[:3]	извлекает элементы, начиная со смещения 0 и до 3 (не включая его)
S[:-1]	извлекает элементы, начиная со смещения 0 и до последнего (не включая его)
S[:]	извлекает элементы, начиная со смещения 0 и до конца, – это эффективный способ создать поверхностную копию последовательности S (т.е. создается объект с тем же значением, но расположенный в другой области памяти)
S[1:10:2]	извлекает каждый второй элемент последовательности от 1 до 9 – то есть будут выбраны элементы со смещениями 1, 3, 5, 7 и 9 (2 – это шаг)
S[::2]	извлекает каждый второй элемент от начала и до конца последовательности

```
s = "0123456789"
s1 = s[2:5]       # "234"
s2 = s[:5]        # "01234"
s3 = s[2:]        # "23456789"
s4 = s[2::2]      # "2468"
s1 = s[-1]        # "9"
s2 = s[2:-1]      # "2345678"
s3 = s[-5:-2]     # "567"
s4 = s[-5::-2]    # "531"
```

Операция получения среза часто используется для удаления лишних символов из строк, считываемых из файлов.

Операции форматирования строк могут выполняться двумя способами:
*Выражения форматирования строк (*основан на модели функции printf из языка C).
!!!! Выражение форматирования всегда создает новую строку, а не изменяет строку, расположенную в левой части.

```
>>> pupil = "Ben"
>>> old = 16
>>> grade = 9.2
>>> print("It's %s, %d. Level: %f" % (pupil, old, grade))
It's Ben, 16. Level: 9.200000
```

Здесь вместо трех комбинаций символов %s, %d, %f подставляются значения переменных pupil, old, grade.

Обратите внимание: когда вставляется более одного значения, в правой части выражения их необходимо сгруппировать с помощью круглых скобок. Оператор форматирования % ожидает получить справа либо один объект, либо списо объектов.

Метод форматирования строк более уникальный для языка Python.

```
>>>age = 26
>>>name = 'Swaroop'
>>>print('Возраст {0} - {1} лет'.format(name, age))
>>>print('Возраст {} - {} лет'.format(name, age)) #без указания цифр

#с именами именованных аргументов (например, {food}).
>>> '{motto}, {0} and {food}'.format(42, motto=3.14, food=[1, 2])
'3.14, 42 and [1, 2]'
```

```
>>> '{0:.2f}'.format(1.2345)                # Строковый метод
'1.23'
>>> format(1.2345, '.2f')                   # Встроенная функция
'1.23'
>>> '%.2f' % 1.2345                         # Выражение форматирования
'1.23'
```

Задания

Задание 1

Пользователь вводит строку. Вывести результаты работы строковых методов, приведенных в таблице 1 (в теоретической части).

Задание 2

Пользователь вводит строку. Программа выводит:

1) В первой строке выведите второй символ этой строки.
2) Во второй строке выведите предпоследний символ этой строки.
3) В третьей строке выведите первые четыре символа этой строки.
4) В четвертой строке выведите всю строку, кроме последних двух символов.
5) В пятой строке выведите все символы с четными индексами (считая, что индексация начинается с 0).
6) В шестой строке выведите все символы с нечетными индексами.
7) В седьмой строке выведите все символы в обратном порядке.
8) В восьмой строке выведите все символы строки через один в обратном порядке, начиная с последнего.
9) В девятой строке выведите длину данной строки.

В каждой строке выведите соответствующее сообщение. Например,

Второй символ строки – t

Задание 3

В1. Пользователь вводит строку. Вывести все символы, чьи индексы кратны 4.
В2. Дана строка. Удалите k-ый символ с начала строки и с конца строки.
В3. Пользователь вводит строку и символ. Посчитать, сколько раз этот символ встречается в строке. Программа не должна зависеть от регистра вводимых символов.
В4. Дана строка, состоящая из слов, разделенных пробелами. Определите количество слов в строке.
В5. Пользователь вводит строку. Разделите ее на две равные части (если длина строки — четная, а если длина строки нечетная, то длина первой части должна быть на один символ больше). Переставьте эти две части местами, результат запишите в новую строку и выведите на экран.
В6. Пользователь вводит строку, состоящую ровно из двух слов, разделенных пробелом. Переставьте эти слова местами. Результат запишите в строку и выведите получившуюся строку.
В7. Пользователь вводит строку. Вывести два первых символа столько раз, какова длина строки.
В8. Дана строка. Определите общее количество символов '+' и '-' в ней.
В9. Дана строка. Вывести из строки символы, расположенные между символами «/*», «*/» включая границы.
В10. Дана строка, состоящая из нескольких слов. Найти длину первого слова.
В11. Написать программу, которая удаляет k символов с позиции номер n из строки S.
В12. Дан английский текст, содержащий запятые. Вывести текст до первой запятой и текст после последней запятой. Например: «hello, world, hi!», получим «hello hi!»
В13. Пользователь вводит строку, состоящую из трех слов, разделенных пробелами. Вывести второе слово.

Задание 4

В1-6: Пользователь вводит строку из 4 слов. Первое и третье слово вывести строчными буквами, второе и четвертое слово – прописными.
В7-13: Пользователь вводит текст, содержащий несколько пар скобок. Вывести текст в первых скобках в верхнем регистре и текст в последних скобках в нижнем регистре.

```
Ввели: Падал(Лишнее1) прошлогодний снег (лишнее2) вместо (лишнЕЕ3)
долгожданной весны
Получили: ЛИШНЕЕ1 лишнее3
```

Контрольные вопросы:

1. Что такое строка? Назовите строковый тип в Python.
2. Как с помощью среза из строки «Приветик» вывести «ик»?
3. Как с помощью среза заменить «п» на «П» в строке А=’программирование’?
4. Какой строковый метод преобразует все буквы в нижний регистр? в верхний регистр?
5. Что будет выведено в результате выполнения кода:
   s=’spam’
   s[0]=’x’
   print(s)
6. Какое значение будет иметь переменная S после выполнения кода:
   S=’123456’
   S=s[: 3] + S[4:]
7. Как получить все буквы слова в обратном порядке (Ира → арИ)?