9.8 Сравнение строк

Шаг 1

Тема урока: сравнение строк

1. Сравнение строк единичной длины
2. Сравнение строк не единичной длины

Аннотация. Урок посвящён сравнению строк: алгоритму и способам сравнения.

В Python мы можем сравнивать с помощью операторов `==`, `!=`, `<`, `<=`, `>` и `>=` не только числа, но и строки. В отличие от чисел, сравнение строк происходит на основе **лексикографического порядка** – в соответствии с кодами составляющих их символов в таблице Unicode.

Сравнение строк единичной длины

Начнем с примера сравнения строк, состоящих из одного символа. В Python это сравнение происходит путём сравнения кодов этих символов в таблице Unicode.

Приведённый ниже код:

```
print('a' > 'b')
print('a' < 'z')
```

выводит:

```
False
True
```

Действительно, код символа `a` в таблице Unicode равен числу 97, а символа `b` – числу 98. Число 97 меньше числа 98, поэтому и символ `a` меньше символа `b`. Аналогично символ `z` больше символа `a`, потому что код символа `z` (число 122) больше кода символа `a` (число 97).

Предыдущий код полностью эквивалентен следующему коду:

```
print(ord('a') > ord('b'))
print(ord('a') < ord('z'))
```

Обратите внимание, что буквы в нижнем регистре всегда больше своих аналогов в верхнем регистре. Для букв русского алфавита это правило также работает.

Приведённый ниже код:

```
print('d' > 'D')
print('ы' < 'ы')
```

выводит:

```
True
True
```

Ничего удивительного в этом нет, потому что буквы в нижнем регистре идут после букв в верхнем регистре в таблице Unicode.

Подробнее ознакомиться с таблицей символов Unicode можно по ссылке (https://symbl.cc/en/unicode-table/). Обычно на практике достаточно оперировать таблицей ASCII (https://www.asciitable.com/), которая является подмножеством таблицы Unicode. Первые 128 символов таблицы Unicode совпадают с таблицей ASCII.

Сравнение строк не единичной длины

Обычно мы работаем не с отдельными символами, а со строками, которые состоят из нескольких символов сразу.

Алгоритм сравнения строк:

1. Начинаем с первых символов каждой строки. Если символы равны, переходим к следующей паре символов
2. Когда находим первый отличающийся символ, строка с меньшим символом считается "меньше"
3. Если одна из строк заканчивается раньше, то более короткая считается "меньше"

Чтобы лучше разобраться с алгоритмом сравнения строк, рассмотрим несколько примеров.

Пример 1. Сравним строки `'hello'` и `'hell'`.

- Сравнение первых символов: `h` и `h` – оба символа равны, переходим к следующей паре символов
- Сравнение вторых символов: `e` и `e` – оба символа равны, переходим к следующей паре символов
- Сравнение третьих символов: `l` и `l` – оба символа равны, переходим к следующей паре символов
- Сравнение четвертых символов: `l` и `l` – оба символа равны, переходим к следующей паре символов
- Сравнение пятых символов: `o` (у первой строки) и отсутствующий символ (у второй строки) – вторая строка закончилась

Поскольку у второй строки закончились символы, а у первой строки они еще есть, считается, что первая строка больше второй. Поэтому `'hello' > 'hell'`.

h e l l o

h e l l

Пример 2. Сравним строки `'men'` и `'mya'`.

1) Сравнение первых символов: `m` и `m` – оба символа равны, переходим к следующей паре символов

2) Сравнение вторых символов: `e` (у первой строки) и `y` (у второй строки) – символ `e` (число 101) меньше символа `y` (число 121)

Так как строки начинают различаться со второго символа, то на основе этого символа и делается вывод о результатах сравнения строк. В данном случае символ `e` меньше символа `y`. Получаем, что `'men' < 'mya'`.

m e n

m y a

Обратите внимание, что в Python сравнение останавливается, как только находится первое различие между символами на соответствующих позициях. Дальнейшее сравнение символов не требуется.

Пример 3. Сравним строки `'Meeeeeooooooow'` и `'meow'`.

1) Сравнение первых символов: `M` (у первой строки) и `m` (у второй строки) – символ `M` (число 77) меньше символа `m` (число 109).

Строки различаются уже на первых символах, и первый символ у первой строки меньше первого символа второй строки. Поэтому `'Meeeeeooooooow' < 'meow'`.

Meeeeeooooooow
meow

Примечания

Примечание 1. Нельзя путать сравнение чисел и сравнение строк, содержащих эти числа.

Приведённый ниже код:

```
print(10 > 9)
print('10' > '9')
```

выводит:

```
True
False
```

Примечание 2. Мы можем сравнивать не только строки, состоящие из букв латинского алфавита, но и строки, состоящие из любых символов, которые входят в таблицу Unicode. Алгоритм сравнения строк при этом будет аналогичный – в соответствии с кодами символов в таблице Unicode.

Приведённый ниже код:

```
print('Тинькофф' == 'Т-банк')
print('¥' > '$€£')
print(max('🐷', '🦆', '🐔'))
```

выводит:

```
False
True
🦆
```

Примечание 3. В Python не поддерживается операция сравнения строк и чисел друг с другом.

Приведённый ниже код:

```
print('45' > 44)
```

приводит к возникновению ошибки:

```
TypeError: '>' not supported between instances of 'str' and 'int'
```

Python пытается выполнить лексикографическое сравнение для строк и числовое сравнение для чисел. Но эти операции несопоставимы, и поэтому возникает ошибка.

Примечание 4. В Python встроенные функции `min()` и `max()` могут принимать строки в качестве аргументов и сравнивают их лексикографически (используя порядок символов в кодировке Unicode). Как несложно догадаться, функция `min()` вернёт самую "маленькую" строку, а `max()` – самую "большую" строку.

Приведённый ниже код:

```
print(max('tree', 'try', 'true'))
print(min('cat', 'car', 'cape'))
```

выводит:

```
try
cape
```

Обратите внимание, что мы не можем **одновременно** передавать строки и числа в качестве аргументов в функции `min()` и `max()`. Это является следствием того, что мы не можем сравнивать строки с числами.

Приведённый ниже код:

```
print(min('2', 8, '45', 90))
```

приводит к возникновению ошибки:

```
TypeError: '<' not supported between instances of 'int' and 'str'
```

❤️ Happy Pythoning! 🐍

Шаг 2

Определите, какой из знаков (`>` , `<` , `==`) нужно поставить на место знака вопроса (`?`), чтобы выражение оказалось истинным.

Примечание. Гарантируется, что все буквенные символы в данной задаче являются буквами английского алфавита.

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/2

Шаг 3

Определите, какой из знаков (`>` , `<` , `==`) нужно поставить на место знака вопроса (`?`), чтобы выражение оказалось истинным.

Примечание. Гарантируется, что все буквенные символы в данной задаче являются буквами английского алфавита.

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/3

Шаг 4

Расположите строки сверху вниз в порядке возрастания.

Примечание 1. Самые "большие" строки должны быть в самом низу, а самые "маленькие" – в самом верху.

Примечание 2. Гарантируется, что все буквенные символы в данной задаче являются буквами английского алфавита.

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/4

Шаг 5

Расположите строки сверху вниз в порядке возрастания.

Примечание. Самые "большие" строки должны быть в самом низу, а самые "маленькие" – в самом верху.

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/5

Шаг 6

Расположите строки сверху вниз в порядке возрастания.

Примечание 1. Самые "большие" строки должны быть в самом низу, а самые "маленькие" – в самом верху.

Примечание 2. Гарантируется, что все буквенные символы в данной задаче являются буквами английского алфавита.

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/6

Шаг 7

Что покажет приведённый ниже код?

```
print(max(9, 10, 11))
print(max('9', '10', '11'))
```

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/7

Шаг 8

Что покажет приведённый ниже код?

```
print(min('9', '10', '11') + max('9', '10', '11'))
```

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/8

Шаг 9

Что покажет приведённый ниже код?

```
print(min(10, 5, 15) + max('10', '5', '15'))
```

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/9

Шаг 10

Что покажет приведённый ниже код?

```
print(min(10, 5, 15) + max(10, 5, '15'))
```

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/10

Шаг 11

Строковые минимум и максимум

На вход программе подаётся последовательность строк, каждая строка на отдельной строке. Концом последовательности является слово «**КОНЕЦ**» (без кавычек). При этом само слово «**КОНЕЦ**» не входит в последовательность, лишь символизируя ее окончание. Напишите программу, которая находит в данной последовательности **максимальную** и **минимальную** строки (в лексикографическом порядке) и выводит их в следующем формате:

```
Минимальная строка ⬇️: <минимальная строка>
Максимальная строка ⬆️: <максимальная строка>
```

Формат входных данных
На вход программе подаётся последовательность строк, каждая на отдельной строке.

Формат выходных данных
Программа должна вывести текст в соответствии с условием задачи.

Примечание. Не только у чисел мы можем находить максимум и минимум. 🤪

▶ Тестовые данные 🟢

Sample Input 1:

```
буря
мглою
небо
кроет
КОНЕЦ
```

Sample Output 1:

```
Минимальная строка ⬇️: буря
Максимальная строка ⬆️: небо
```

Sample Input 2:

```
вихри
снежные
крутя
КОНЕЦ
```

Sample Output 2:

```
Минимальная строка ⬇️: вихри
Максимальная строка ⬆️: снежные
```

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/11

Шаг 12

Волшебное число ✨

В некотором наборе слов Сэм находит "волшебное" число по следующему алгоритму: берет самую "маленькую" и самую "большую" строки, перемножает Unicode-коды последних символов этих строк и возводит полученное число в квадрат. Результатом и является "волшебное" число.

На вход программе подаются 4 слова. Найдите "волшебное" число в этом наборе слов.

Формат входных данных
На вход программе подаются 4 слова, каждое на отдельной строке.

Формат выходных данных
Программа должна вывести "волшебное" число в наборе слов.

▶ Тестовые данные

Sample Input 1:

```
I
will
be
back
```

Sample Output 1:

```
62157456
```

Sample Input 2:

```
all
dreams
come
true
```

Sample Output 2:

```
118984464
```

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/12

Шаг 13

Название класса 🏫 🌶

В школе BEEGEEK названия учебных классов необычные. Они имеют следующий формат:

```
<номер класса><буква класса>
```

где <номер класса> должен находиться в диапазоне от 0 (как и все у программистов) до 9 включительно, а буквой класса могут быть все буквы в диапазоне от «А» до «П» включительно.

Напишите программу, которая принимает натуральное число n, а далее n названий классов, каждое на новой строке. Для каждого названия класса ваша программа должна выводить на отдельной строке «YES» (без кавычек), если название класса корректное, или «NO» (без кавычек) в противном случае.

Формат входных данных
На вход программе подаются натуральное число n, а затем n названий классов, каждое на отдельной строке.

Формат выходных данных
Программа должна вывести на отдельной строке для каждого названия класса «YES» (без кавычек) или «NO» (без кавычек) в соответствии с условием задачи.

Примечание. Будем считать, что буквы `Ё` нет в русском алфавите, а значит, класс с такой буквой также будет считаться некорректным. 😳

▶ Тестовые данные 🟢

Sample Input 1:

```
5
9А
11Б
0К
5П
2У
```

Sample Output 1:

```
YES
NO
YES
YES
NO
```

Sample Input 2:

```
4
10А
00
5Д
2Я
```

Sample Output 2:

```
NO
YES
YES
NO
```

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/13

Шаг 14

Необычное сравнение 📊

На вход программе подаются 2 строки. Вам необходимо сравнить эти строки посимвольно, не учитывая регистр и игнорируя все небуквенные символы. Программа должна вывести «YES» (без кавычек), если строки окажутся равны в результате такой проверки, или «NO» (без кавычек) в противном случае.

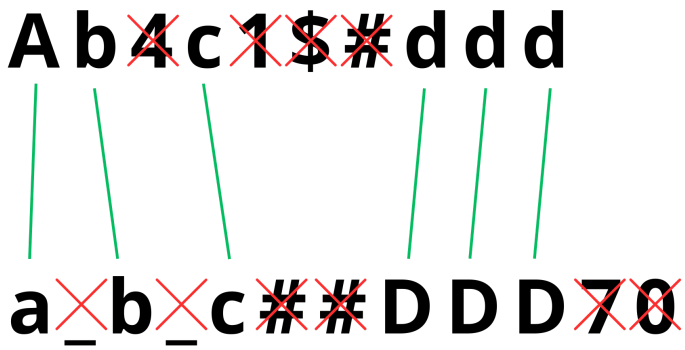
Формат входных данных

На вход программе подаются 2 строки, каждая на отдельной строке.

Формат выходных данных

Программа должна вывести «YES» (без кавычек) или «NO» (без кавычек) в соответствии с условием задачи.

Примечание. Разберём 1-й тест:



▶ Тестовые данные

Sample Input 1:

```
Ab4c1$#ddd
a_b_c##DDD70
```

Sample Output 1:

```
YES
```

Sample Input 2:

```
n5#e6vER
+NEV-er
```

Sample Output 2:

```
YES
```

Sample Input 3:

```
c-э-м
С-Э-Э-М
```

Sample Output 3:

```
NO
```

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/14

Шаг 15

Сортируем слова 📶

На вход программе подаются 3 **различных** слова. Вам необходимо отсортировать эти слова по возрастанию в лексикографическом порядке и вывести их на одной строке, разделяя символом пробела.

Формат входных данных

На вход программе подаются 3 слова, каждое на отдельной строке.

Формат выходных данных
Программа должна вывести 3 слова на одной строке, разделяя их символом пробела.

▶ Тестовые данные

Sample Input 1:

```
python
java
kotlin
```

Sample Output 1:

```
java kotlin python
```

Sample Input 2:

```
первое
второе
третье
```

Sample Output 2:

```
второе первое третье
```

Sample Input 3:

```
июнь
июль
август
```

Sample Output 3:

```
август июль июнь
```

Чтобы решить это задание откройте https://stepik.org/lesson/1402735/step/15

Шаг 16

Порядок книг 📚 🌶️

Все книги в домашней библиотеке Душнилы, друга Сэма, должны быть обязательно отсортированы **по возрастанию**: сначала по фамилиям авторов, а в случае совпадения фамилий – по названиям. Напишите программу, которая проверяет, верно ли отсортированы книги.

На вход вашей программе поступает число n, а затем – n строк, каждая строка представляет собой книгу в следующем формате:

```
<фамилия автора> <инициалы автора>, «<название книги>»
```

Программа должна вывести «YES» (без кавычек), если книги отсортированы в соответствии с пожеланиями Душнилы, или «NO» (без кавычек) в противном случае.

Формат входных данных
На вход программе подаются натуральное число n, а затем – n строк.

Формат выходных данных
Программа должна вывести «YES» (без кавычек) или «NO» (без кавычек) в соответствии с условием задачи.

Примечание 1. Обратите внимание, что Душнила **игнорирует** инициалы автора при сортировке книг.

Примечание 2. Гарантируется, что книги в наборе не повторяются.

Примечание 3. Гарантируется, что фамилия автора состоит из одного слова.

▶ Тестовые данные

Sample Input 1:

```
5
Гоголь Н.В., «Мертвые души»
Гончаров И.А., «Обломов»
Пушкин А.С., «Капитанская дочка»
Тургенев И.С., «Ася»
Тургенев И.С., «Первая любовь»
```

Sample Output 1:

```
YES
```

Sample Input 2:

```
3
Толстой А.Н., «Петр Первый»
Толстой А.Н., «Хмурое утро»
Толстой Л.Н., «Война и мир»
```

Sample Output 2:

```
NO
```