Занятие 12. Базовый синтаксис Python

План занятия

1. Логические блоки

В отличие от многих других языков программирования, в Python для обозначения отдельных блоков кода вместо скобок используются двоеточия и отступы. Отступ блока задается пробелами, количество которых для блока может быть произвольным (обычно задают четыре пробела на каждый блок).

2. Литералы

2.1 Понятие литерала

Литералы, иначе константы — это неизменяемые величины.

Существуют **строковые, целочисленные и действительные литералы**.

2.1.1 Строковые литералы

Строковые литералы бывают двух типов: String-литералы и Byte-литералы.

String-литералы могут содержать любой символ UTF-8. В UTF-8 применяется кодировка с переменной длиной, 1-4 байта на кодовую точку. Значения ASCII кодируются как ASCII с использованием 1 байта. Формально говоря, UTF-8 - это кодировка, т.е. алгоритм, который преобразует список чисел в двоичный файл, чтобы он мог быть сохранен на диске.

Byte-литералы содержат только символы unicode. Unicode не является кодировкой, хотя, к сожалению, многие документы неявно используют ее для обозначения какой-либо кодировки Юникода, которая конкретная система использует по умолчанию. В Windows и Java это часто означает UTF-16; во многих других местах это означает UTF-8, т.е. Unicode относится к набору абстрактных символов, а не к какой-либо конкретной кодировке.

Юникод – это набор символов, каждому из которых присвоен уникальный номерами (эти числа иногда называются "кодовыми точками"). Например, в набор символов Unicode, номер для *A* равен 41.

Byte-литералы должны начинаться с префикса "B" или "b". Литералы обоих типов должны заключаться в двойные ("…") или одинарные кавычки ('…').

Примеры:

```
line = 'test'            # Это строка в UTF-8

line2 = u'тест'          # Это строка в Unicode
```

Для длинных строковых литералов используются тройные кавычки """ или '''. В пределах тройных кавычек можно использовать двойные или одинарные кавычки, использовать перевод на новую строку.

Выражения в Python, как правило, заканчиваются новой строкой. Однако, слишком длинную строку можно разбить на несколько строк, используя специальный символ переноса строки (\).

Строковые литералы, ограниченные одинарными ('), или двойными (") кавычками представляются одной строкой (линией).

Строковые литералы, ограниченные тройными (""") кавычками представляются в нескольких строках (линиях).

2.1.2 Объединение нескольких физических строк

Две (или более) физические строки могут быть объединены в одну логическую строку с использованием обратного слэша ("\"), а именно: когда физическая строка заканчивается символом "\" и он не является частью строки (т. е. не находится в пределах " ") или комментария, он присоединяет следующую физическую строку, образуя одну логическую строку. После символа "\" должен следовать символ признака конца строки (т. е. физическая строка должна заканчиваться).

2.1.3 Неявное объединение физических строк

Выражения в скобках, в квадратных скобках, либо в фигурных скобках можно разделить на несколько физических строк без использования обратной косой черты.

2.1.4 Целочисленные литералы

В Python поддерживаются целочисленные литералы четырех типов:

1. decimalinteger — десятичные числа. Префикс не используется.
2. octinteger — восьмеричные числа. Префикс "0O" или "0o".
3. hexinteger — 16-ричные числа. Префикс "0x" или "0X".
4. bininteger — двоичные числа. Префикс "0b" или "0B".

2.1.5 Действительные (вещественные) литералы

Действительные (вещественные) литералы представляются числами с фиксированной точкой (5.7, .001, 35., 0.0) и числами с плавающей точкой (экспоненциальный) (0.2E6, .11e-3, 5E10, 2.e-10).

Примечание. Для представления порядка и мантиссы используются числа по основанию 10. Числовые литералы не включают в себя знак ("-" или "+").

2.2 Escape-последовательности

Escape-последовательности (или управляющие последовательности) используются для описания определённых специальных символов внутри строковых литералов, то есть внутри ограничителей "". Вот некоторые из них:

\' — одинарная кавычка

\" — двойная кавычка

\? — вопросительный знак

\\ — обратный слеш

\n — новая строка

\t — горизонтальная табуляция

\v — вертикальная табуляция

\0 — нулевой символ

2.3 Комментарии

Все символы после # и до конца физической линии являются частью комментария и интерпретатор Python игнорирует их.

Для сохранения информации в файл или передачи по сети. рекомендуется использовать кодировку символов в utf8, поэтому в начале программы Python рекомендуется указать строку:

```
# -*- coding: utf-8 -*-
```