

Модуль 2. Урок 3.
СПИСКИ

алгоритмика

Повторим

Сегодня на занятии:

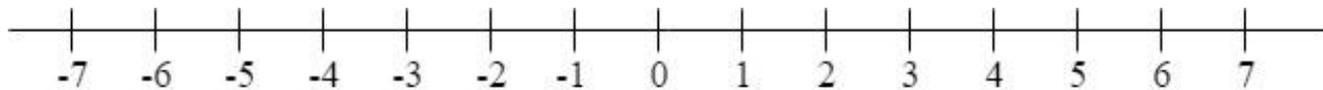


- Покорение новых высот программирования;
- Работа со списками;
- Связь строк и списков — похожи ли?
- Big Data — раскрытие тайн больших данных!
- Познание новой магии — магии списков!

Школьный журнал — пример списка

	Школьный журнал		
1	Агаева И.А.		
2	Виноградов А.С.		
3	Григорян Е.Е.		
4	Григорян С.П.		
5	Дятлов А.В.		
6	Ежов В.А.		
7	Железнов С.А.		
8	Жилова А.П.		
9	Зиновьев А.Д.		
10	Исаева М.П.		
11	Константинов А.Р.		

Ещё примеры СПИСКОВ



1

2

3

4

5

6

7



Анонс

алгоритмика

Что же такое список?

Список —

это последовательность элементов любого типа, например, строк или чисел.

Демонстрация

(пустой список)

Заполнение списка

```
checklist = ["one" , "two" , "three"]
```

Два вида списков

```
checklist = ["one" , "two" , "three"]
```

```
checklist2 = [1 , 2 , 3]
```

алгоритмика

**Как измерить длину
списка?**

Функция len()—

функция, считающая количество элементов в списке.

Демонстрация

(считаем длину списка)

Демонстрация

(считаем длину списков в начальном примере)

Таблица индексов элементов списка

Элементы первого списка	Индекс (с 0)
one	0
two	1
three	2
Элементы второго списка	Индекс (с 0)
1	0
2	1
3	2

Демонстрация

(вывод списков и их элементов)

Вывод элементов списка по индексам

имя переменной, которой присвоено значение списка [индекс элемента]

Пример:

```
checklist = [1, 2, 3]
```

```
print(checklist[2])
```

Вывод:

3

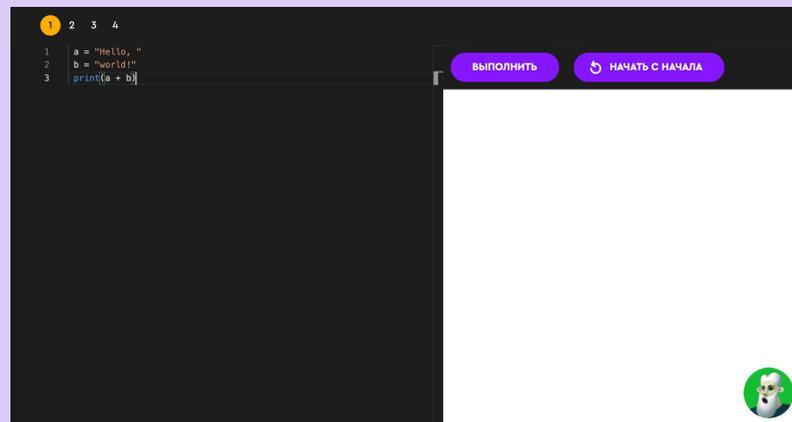
Демонстрация

(дополнение списка)

Заходим на платформу



Списки



The image shows a code editor window with a dark theme. The code is as follows:

```
1 a = "Hello, "  
2 b = "world!"  
3 print(a + b)
```

At the top of the editor, there are four numbered tabs: 1 (highlighted in yellow), 2, 3, and 4. On the right side of the editor, there are two buttons: a purple button labeled "ВЫПОЛНИТЬ" (Execute) and a purple button labeled "НАЧАТЬ С НАЧАЛА" (Start from the beginning). In the bottom right corner of the editor, there is a small circular icon of a white robot head on a green background.

Задание на платформе

Итоги первой половины урока



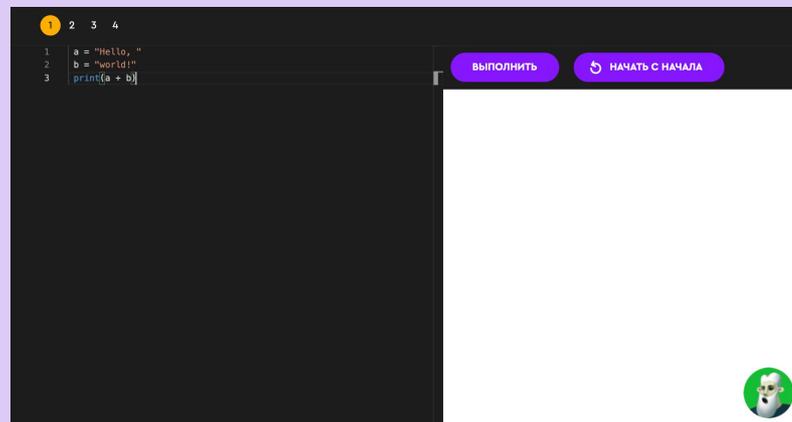
алгоритмика

Давайте отдохнём!

Заходим на платформу



Списки



The image shows a code editor window with a dark theme. The code is as follows:

```
1 a = "Hello, "  
2 b = "world!"  
3 print(a + b)
```

At the top of the editor, there are four numbered tabs: 1 (highlighted in yellow), 2, 3, and 4. On the right side of the editor, there are two buttons: a purple button labeled "ВЫПОЛНИТЬ" (Execute) and a purple button labeled "НАЧАТЬ С НАЧАЛА" (Start from the beginning). In the bottom right corner of the editor, there is a small circular icon of a white robot head on a green background.

Задание на платформе

алгоритмика

Big Data

алгоритмика

**Что такое база
данных?**

База данных —

это набор данных для информационных сетей и пользователей. Большой объём информации, которую нужно хранить.



ВИТАЛИЙ ЖУКОВ
КАК ОБРАБАТЫВАЮТСЯ
БОЛЬШИЕ ДАННЫЕ?



Основные принципы работы с большими данными

- Горизонтальная масштабируемость
- Отказоустойчивость
- Локальность данных

Подходы к анализу данных

- Данные обрабатываются аналитиками, которые решают задачу с помощью известных им моделей и личных предположений
- Data-scientists создают алгоритм искусственного интеллекта, который сам учится решать поставленные задачи

Игра «Го»



Компания «Caterpillar»



Проверь себя

- Что такое список?
- Для чего он используется?
- С помощью какого метода можно дополнять список после запуска программы?
- Как посчитать длину списка?
- Как обратиться к элементу списка по индексу?
- Какие методы работы со списками вы узнали?
- Что такое база данных?
- Как обрабатываются большие данные?

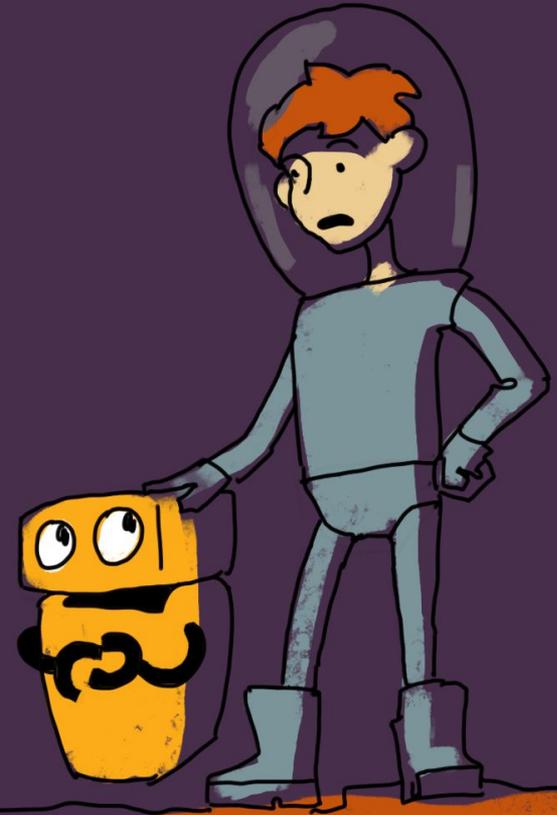
алгоритмика

Как прошло занятие?

На следующем занятии:

- Новый цикл — цикл for.
- Развитие мозга — прокачаемся по полной!

До встречи!



Работа со списками метод `split()`

Используя метод `split()`, можно преобразовать введённую пользователем строку в список. Он разрежет исходную строку из элементов на части по пробелам.

Пример:

```
checklist = input("Введите элементы  
через пробел: ")
```

```
checklist = checklist.split()
```

```
print(checklist)
```

Ввод:

```
1 2 3 4 5 6 7 8
```

Вывод:

```
[1, 2, 3, 4, 5, 6, 7, 8]
```

Работа со списками метод `split()` для чисел

Если пользователь хочет ввести, в качестве элементов списка, числа, тогда, помимо функции `input()` и метода `split()`, необходимо использовать ещё функцию `map()`. Она принимает в качестве аргумента функцию, например `int`, и применяет её к каждому элементу списка, то есть делает каждую строку целым числом.

Пример без применения `map()`:

```
checklist = input("Введите элементы через  
пробел:").split()  
print(checklist)
```

Ввод:

1 2 3

Вывод:

```
['1', '2', '3']
```

Пример с применением `map()`:

```
checklist = input("Введите элементы через  
пробел:").split()  
checklist2 = map(int, checklist)  
print(checklist2)
```

Ввод:

1 2 3

Вывод:

```
[1, 2, 3]
```

Работа со списками метод `join()`

Если все элементы списка — строки, то, при выводе на экран, они отделяются друг от друга запятыми. Чтобы изменить это, нужно применить метод `join()`.

Он работает следующим образом: в качестве аргумента принимается список строк, который затем объединяется в одну строку. На места, где раньше стояли пробелы и запятые, вставляется разделитель, равный той строке, к которой применялся метод.

Пример:

```
checklist = ["one", "two", "three"]  
print("%".join(checklist))
```

Вывод без метода `join()`:
['one', 'two', 'three']

Вывод с применением метода `join()`:
one%two%three

Работа со списками

метод `join()` (для чисел)

Если же элементы списка — это числа, то вместе с методом `join()` нужно также применить функцию `map()`. В качестве её аргументов первым указывается функция `str`, поскольку метод `join()` возвращает одну строку, а затем, через запятую имя переменной, которой присвоен список.

Пример:

```
checklist = [1, 2, 3]
print("%".join(map(str,checklist)))
```

Вывод без метода `join()`:

```
[1, 2, 3]
```

Вывод с применением метода

`join()`:

```
1%2%3
```

Работа со списками

конкатенация и дублирование

Строка — это частный случай списка, в качестве элементов в ней выступает последовательность символов. Поэтому, такие операции, как конкатенация и дублирование применимы ко всем спискам. Применяются они точно так же, как со строками: при сложении двух списков все их элементы объединяются в один, а при умножении списка на число, все элементы дублируются такое количество раз, какому равно это число.

Пример:

```
checklist = input("Введите  
элементы через пробел:").split()  
print(checklist + checklist)  
print(checklist * 3)
```

Ввод:

Привет

Вывод:

```
['привет', 'привет']
```

```
['привет', 'привет', 'привет']
```