

Стать IT-шником может каждый



Python Summary 1



Преподаватель



ИМЯ ФАМИЛИЯ

Текущая должность

Количество лет опыта

Какой у Вас опыт - ключевые кейсы

Самые яркие проекты

Дополнительная информация по вашему усмотрению

[Корпоративный e-mail](#)

[Социальные сети \(по желанию\)](#)

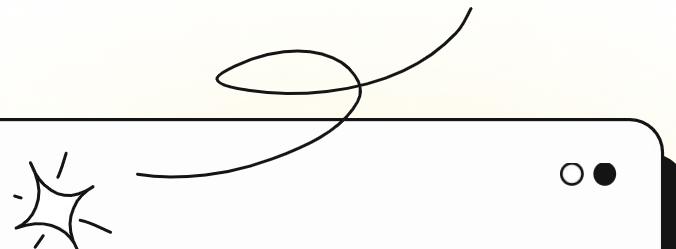


ВАЖНО!

- Камера должна быть включена на протяжении всего занятия.
- Если у Вас возник вопрос в процессе занятия, пожалуйста, поднимите руку и дождитесь, пока преподаватель закончит мысль и спросит Вас, также можно задать вопрос в чате или когда преподаватель скажет, что начался блок вопросов.
- Организационные вопросы по обучению решаются с кураторами, а не на тематических занятиях.
- Вести себя уважительно и этично по отношению к остальным участникам занятия.
- Во время занятия будут интерактивные задания, будьте готовы включить микрофон или демонстрацию экрана по просьбе преподавателя.

ПЛАН ЗАНЯТИЯ

- Повторение изученного за неделю
- Вопросы по повторению
- Разбор домашних заданий
- Вопросы по разбору заданий
- Задание для закрепления
- Оставшиеся вопросы



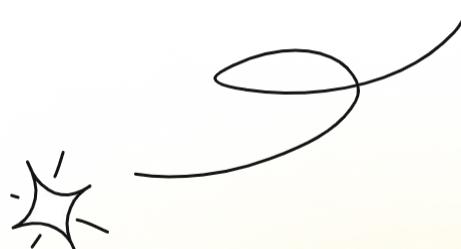
Стать IT-шником может каждый



ПОВТОРЕНИЕ ИЗУЧЕННОГО



Экспресс-опрос



Что такое Python?

Что помните об истории создания языка?

Стать IT-шником может каждый



ПОЛЕЗНО ЗНАТЬ

Python

Это язык программирования, созданный Гвидо ван Россумом в конце 1980-х годов.



ПОЛЕЗНО ЗНАТЬ

Python

Назван Python в честь любимого комедийного шоу Гвидо ван Россума "Монти Пайтон и священный Грааль".



Стать IT-шником может каждый



История Python

1989

2000

2008

...

2022

Начало разработки

Python2

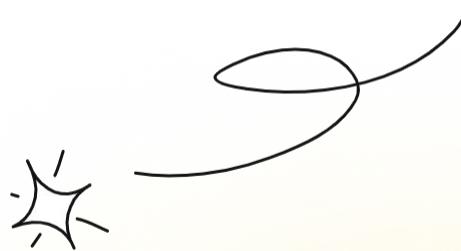
Python3

Версия 3.10.6
(02.08.2022), 3.7.6,
3.9.4

Стать IT-шником может каждый



Экспресс-опрос



Что такое интерпретатор и зачем он нужен?





Интерпретатор —

это [программа](#), которая выполняет код Python. Он считывает и интерпретирует инструкции построчно, преобразуя их в машинный код или байт-код, который может быть непосредственно выполнен компьютером.





CPython

Наиболее распространенный
интерпретатор Python

- CPython является референсной реализацией языка.
- CPython написан на языке С и обеспечивает высокую производительность и надежность.

Стать IT-шником может каждый



Другие интерпретаторы Python:

Jython
(реализация на Java)

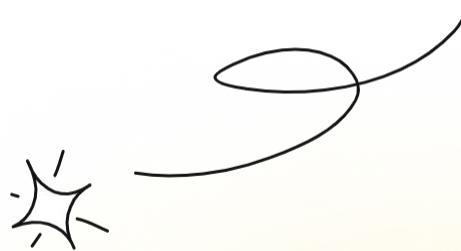
IronPython
(реализация на платформе .NET)

PyPy
(Just-in-Time компилятор)

Стать IT-шником может каждый



Экспресс-опрос



Что такое Jupyter Notebook и зачем он нужен?



Jupyter Notebook —

это интерактивная среда разработки и обмена документами, которая позволяет создавать и выполнять код Python в виде ячеек.



Стать IT-шником может каждый

Jupyter Notebook

Веб-интерфейс

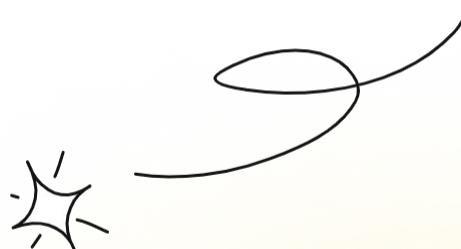
Позволяет создавать и редактировать ноутбуки

Встроенный интерпретатор Python

Выполняет код в каждой ячейке ноутбука



Экспресс-опрос



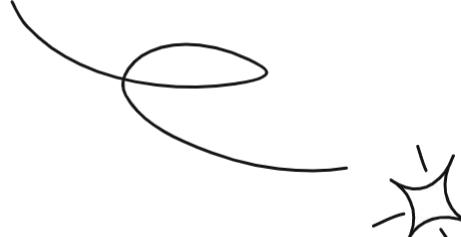
Что такое переменная в Python? Что помните про переменные?

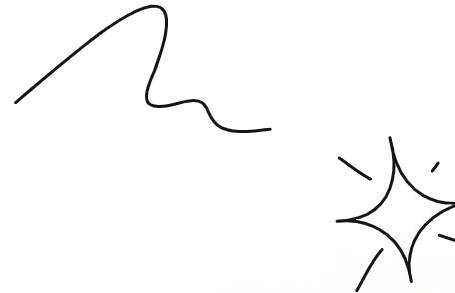




Переменная —

это **символическое имя**, которое используется для хранения и обращения к данным в памяти компьютера. Другими словами, переменная — хранилище, в которое можно положить разного рода данные.





Переменная



- Переменные позволяют сохранять значения и использовать их в различных частях программы.
- В Python для создания переменной достаточно указать имя переменной и присвоить ей значение.
- “Программа – есть перекладывание одной переменной в другую хитрым способом”
- Имя переменной – последовательность букв, цифр и знак _, которая начинается не с цифры.

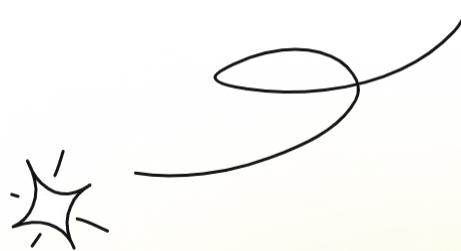
Пример:

`x = 5` создаст переменную `x` и присвоит ей значение 5.

Стать IT-шником может каждый



Экспресс-опрос



За что отвечают команды print и input?



print —

это команда, которая используется для вывода информации на экран. Она позволяет нам отображать значения переменных, текстовые сообщения и результаты вычислений.

```
print("Hello, world!")
```





input —

это команда, которая используется для ввода данных пользователем. Она позволяет нам запрашивать данные у пользователя и сохранять их в переменных.

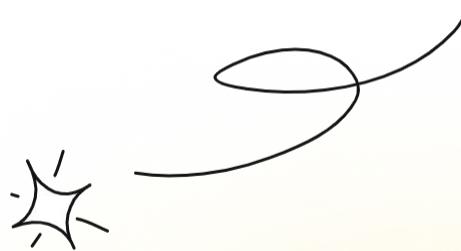
```
name = input("Введите ваше имя: ")
```



Стать IT-шником может каждый



Экспресс-опрос



Зачем используют комментарии при написании кода?



Комментарии —

это текст в коде программы, который не влияет на функциональность и добавляется программистами для себя и своих коллег.





Комментарии

- С помощью комментариев добавляют пояснения, как работает код, какие ошибки нужно исправить или не забыть что-то добавить позже:

```
# Удалить строку ниже после реализации задачи по регистрации
print(10)
```

- Комментарии в Python начинаются со знака # и могут появляться в любом месте программы.

```
# For Winterfell!
# For Lanisters!
```

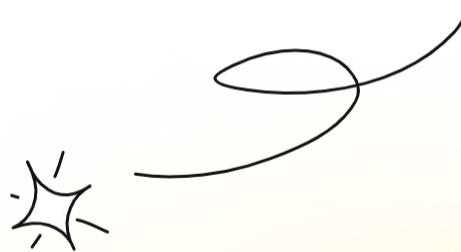
- Комментарий может находиться на строке после кода.

```
print('I am the King') # For Lannisters!
```

Стать IT-шником может каждый



Экспресс-опрос



Что такое синтаксис и какие правила помните?



Синтаксис —

это набор правил, описывающих комбинации символов алфавита, считающиеся правильно структурированной программой (документом) или её фрагментом.





Синтаксис



- Конец строки является концом инструкции (точка с запятой не требуется).
- Вложенные инструкции объединяются в блоки по величине отступов.
- Отступ может быть любым, в пределах одного вложенного блока отступ одинаков.



Синтаксис



- Вложенные инструкции: основная инструкция завершается двоеточием, за которым располагается вложенный блок кода, обычно с отступом под строкой основной инструкции.

Основная инструкция:

 Вложенный блок инструкций

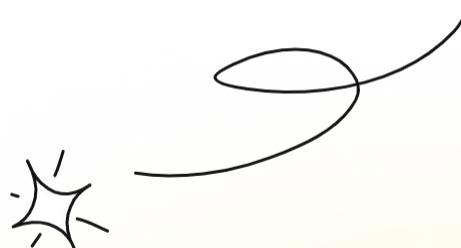
- Иногда возможно записать несколько инструкций в одной строке, разделяя их точкой с запятой.

a = 1; b = 2; print(a, b)

Стать IT-шником может каждый



Экспресс-опрос



Что такое синтаксические ошибки?





Синтаксические ошибки —

это **нарушение** грамматических правил языка
программирования.





SyntaxError (ошибка парсинга) —

это тип ошибок в Python, возникающих при наличии синтаксических ошибок в коде.





Синтаксические ошибки



- Если программа написана синтаксически некорректно, то интерпретатор выводит на экран соответствующее сообщение.
- Интерпретатор указывает на файл и строку, где произошла ошибка.
- Синтаксическая ошибка возникает, когда код записали с нарушением грамматических правил.

Стать IT-шником может каждый



Синтаксические ошибки



Пример кода с синтаксической ошибкой:

```
print('Hodor')
```

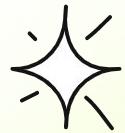
При запуске этого кода:

```
$ python index.py
  File "index.py", line 1
    print('Hodor')
               ^
SyntaxError: EOL while scanning string literal
```

Стать IT-шником может каждый



ВОПРОСЫ



Стать IT-шником может каждый



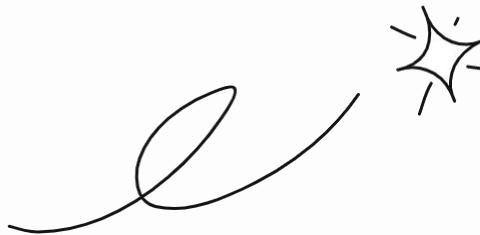
ЗАДАНИЕ ДЛЯ ЗАКРЕПЛЕНИЯ



Ответь в чат:

Какой код вы введете, чтобы отобразить строку в консоли?

1. `print (my message)`
2. `println ("my message")`
3. `print ("my message")`

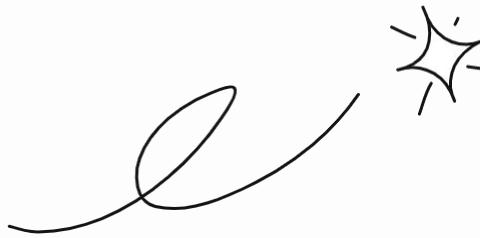




Ответь в чат:

Что произойдет, если выполнить инструкцию "1" + 2?

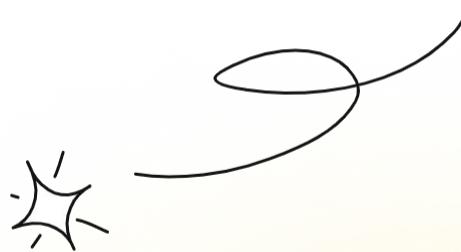
1. Будет вычислено значение 3.
2. Поступит сообщение об ошибке.
3. Поступит ответ "12".



Стать IT-шником может каждый

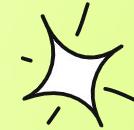


Экспресс-опрос



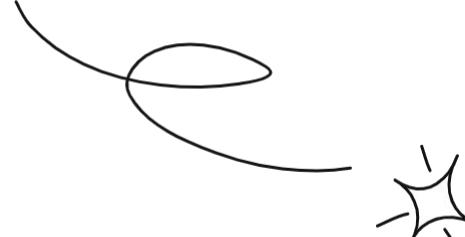
Что такое стандартные потоки ввода и вывода?





Стандартные потоки ввода и вывода в Python —

это способы обмена данными между программой и пользователем.



Стать IT-шником может каждый



Стандартные потоки ввода и вывода в Python



- Стандартный поток ввода (**stdin**) используется для чтения данных из внешних источников, таких как клавиатура.
- Стандартный поток вывода (**stdout**) используется для вывода данных на экран.

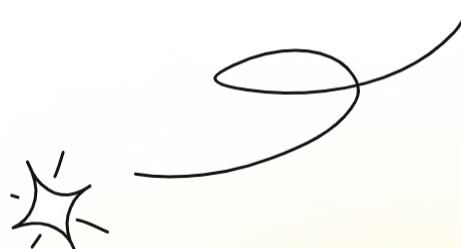
Стандартные потоки ввода и вывода в Python

- **input** используют для чтения данных из стандартного потока ввода
- **print** используют для вывода данных в стандартный поток вывода

Стать IT-шником может каждый



Экспресс-опрос



За что отвечают типы переменных?



Тип переменных —

это зарезервированные ячейки памяти для хранения значений. Это означает, что при создании переменной вы резервируете некоторое место в памяти.



Тип переменных

- Тип переменной определяет, какие значения и операции могут быть применены к переменной.
- Посмотреть тип можно с помощью команды **type**.

Типы данных

Строки (str)

Строки представляют собой последовательность символов, заключенных в кавычки (одинарные или двойные).

Пример:
"Hello" и 'World' - это строки

Числа (int, float)

Числа могут быть целыми (int) или вещественными (float).

Пример:
5 и 3.14 - это числа

Стать IT-шником может каждый



Знакомство со строками и числами

- К одному числу мы можем прибавить второе. Или поделить на число (если это не ноль!)

Знакомство со строками и числами

- К одному числу мы можем прибавить второе. Или поделить на число (если это не ноль!)
- Строки нельзя делить друг на друга.

Знакомство со строками и числами

- К одному числу мы можем прибавить второе. Или поделить на число (если это не ноль!)
- Строки нельзя делить друг на друга.
- Строки заключаются с двух сторон в одинарные кавычки, или двойные кавычки (один символ), или же обрамляются тремя одинарными кавычками или тремя двойными (случай многострочной строки).

Пример:
"Hello" и 'World' - это строки

Экспресс-опрос



Что такое системы счисления?

Как перевести из любой системы в двоичную?

Системы счисления —
это способ записи (представления) чисел.



Стать IT-шником может каждый



СИСТЕМЫ СЧИСЛЕНИЯ

Непозиционные

Позиционные

Двоичная
Восьмеричная
Десятичная
Шестнадцатеричная
...



Представление данных



- Входные и выходные данные представляются в форме, удобной для человека.
- Числа люди привыкли изображать в десятичной системе счисления (0..9).

Пример:

1, 2, 45, 1003...

Представление данных

Для компьютера удобнее двоичная система:

- Любые данные в компьютере представляются в виде последовательностей из нулей и единиц:
1, 10001, 0001010...
- Обработка данных внутри компьютера = преобразование слов из нулей и единиц по правилам, зафиксированным в микросхемах процессора.

Бит —

элемент **последовательности** из нулей и единиц
(член такой последовательности).



Кодирование —

отображение внешней информации во внутреннее представление.



Код (франц. code, от лат.
codex — свод законов)

это способ отображения внешней информации во
внутреннее представление, и множество слов
(кодовых комбинаций), используемых при
кодировании.



Полезно знать



ЭВМ —
электронно-вычислительная машина.





Битовые наборы —

это последовательности нулей и единиц фиксированной длины.



Стать IT-шником может каждый



Битовые наборы

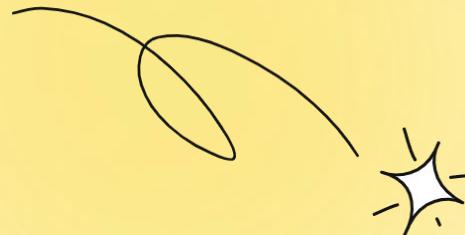
- используются для представления чисел в ЭВМ.
- организовать обработку наборов фиксированной длины технически легче, чем наборов переменной длины.

Полезно знать



Разряд —
это позиция в битовом наборе.





Разряд

В ЭВМ разрядом называют также часть регистра (или ячейки памяти), хранящую один бит.

Тысячи	Сотни	Десятки	Единицы
2	0	9	8



Перевод чисел из одной системы счисления в другую —

это процесс представления числа в различных системах счисления.



Десятичная система счисления

- Основана на использовании десяти цифр (0-9)
- Каждая позиция числа в десятичной системе имеет вес, который определяется позицией цифры от правого к левому концу числа.

Пример:

Число 1234 в десятичной системе имеет значение:

$$(1 * 10^3) + (2 * 10^2) + (3 * 10^1) + (4 * 10^0)$$

[^] - обозначение степени

Бинарная (двоичная) система счисления

- Использует две цифры (0 и 1).
- Каждая позиция числа в бинарной системе имеет вес, который определяется позицией цифры от правого к левому концу числа.

Пример:

Число 101 в бинарной системе имеет значение:

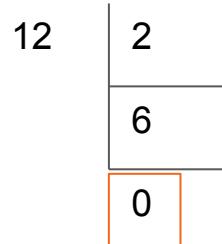
$$(1 * 2^2) + (0 * 2^1) + (1 * 2^0) = 5 \text{ в десятичной системе.}$$



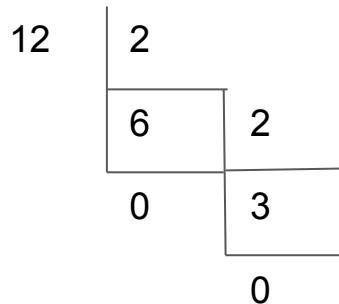
Перевод из десятичной системы в другие системы

1. Выполнить последовательное деление числа на основание новой системы счисления и сохранять остатки.
2. Остатки записывать в обратном порядке.
3. Получить искомое число в новой системе счисления.

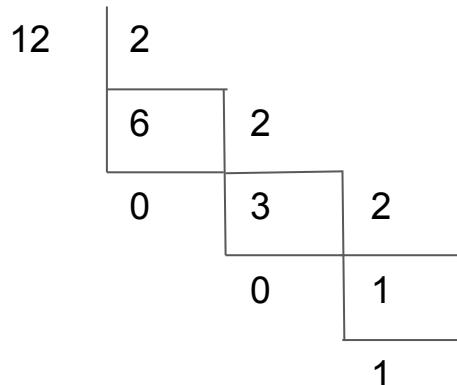
Перевод из десятичной системы в другие системы



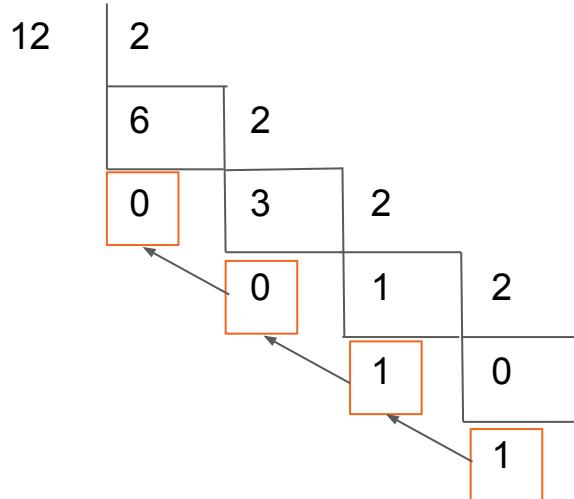
Перевод из десятичной системы в другие системы



Перевод из десятичной системы в другие системы



Перевод из десятичной системы в другие системы



$$12_{10} = 1100_2$$



Перевод из других систем счисления в десятичную систему

1. Умножить каждую цифру числа на соответствующую степень основания системы счисления.
2. Сложить все полученные произведения.

Шестнадцатеричная система счисления

- Использует 16 цифр (0-9 и A-F).
- Шестнадцатеричные числа могут быть записаны с помощью "0x" или "0X".

Пример:

Число 1A в шестнадцатеричной системе имеет значение:

$$(1 * 16^1) + (10 * 16^0) = 26 \text{ в десятичной системе.}$$



Обратный перевод —

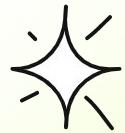
это процесс перевода **числа** из другой системы счисления обратно в десятичную систему. После этого число можно преобразовать в другую систему счисления, если необходимо.



Стать IT-шником может каждый



ВОПРОСЫ



Стать IT-шником может каждый



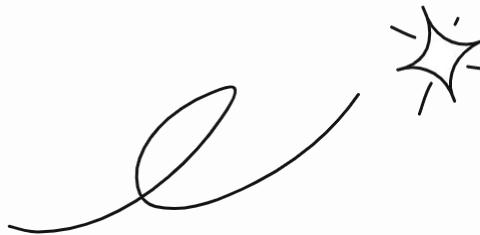
ЗАДАНИЕ ДЛЯ ЗАКРЕПЛЕНИЯ



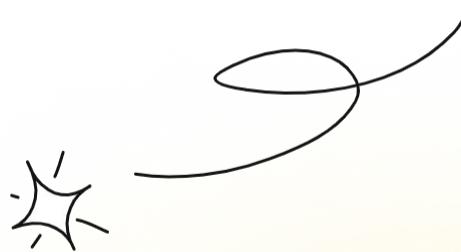


Ответь в чат:

1. Десятичная система счисления — это позиционная система счисления или непозиционная?
 - a. Непозиционная.
 - b. Позиционная.
2. В какой системе счисления «12» записывается как «С»?
 - a. В шестнадцатеричной
 - b. В восьмеричной
 - c. В двоичной
3. Число 11_2 больше числа 11_{10} ?
 - a. Да
 - b. Нет.



Экспресс-опрос



Приведите примеры арифметических операций в Python?

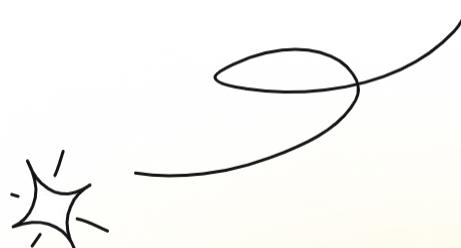
Арифметические операции в Python

- Сложение (+)
- Вычитание (-)
- Умножение (*)
- Деление (/)
- Целочисленное деление (//)
 - возвращает целую часть от деления одного числа на другое.
- Взятие остатка от деления (%)
 - возвращает остаток от деления одного числа на другое.
- Возведение в степень (**)

Стать IT-шником может каждый



Экспресс-опрос



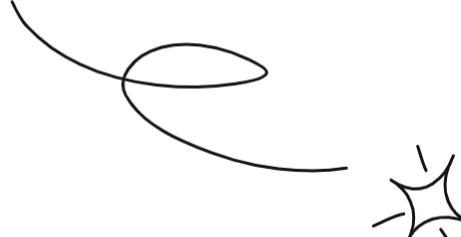
Что такое вещественные числа?





Вещественные числа —

множество рациональных и множества иррациональных чисел. Вещественные числа могут быть как положительными, так и отрицательными, а также нулем.



Операции с вещественными числами

- Вещественные числа хранятся в памяти компьютера с некоторой погрешностью.
- Сравнивать на равенство вещественные числа с помощью знака == не совсем корректно.

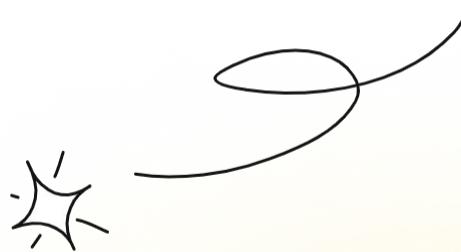
ПРИМЕР

```
x = 0.1 + 0.1 + 0.1
print(x) # Выводит 0.3000000000000004
```

Стать IT-шником может каждый



Экспресс-опрос



Что такое битовые операции?





Битовые операции

Побитовое И (&)

Побитовое ИЛИ (|)

Побитовое исключающее ИЛИ (^)

Побитовый сдвиг влево (<<)

Побитовый сдвиг вправо (>>)

Побитовое отрицание (~)

Выполняет побитовую конъюнкцию двух чисел

Выполняет побитовую дизъюнкцию двух чисел

Выполняет побитовое исключающее ИЛИ двух чисел

Сдвигает биты числа влево на указанное количество позиций

Сдвигает биты числа вправо на указанное количество позиций

Инвертирует все биты числа



ПРИМЕР

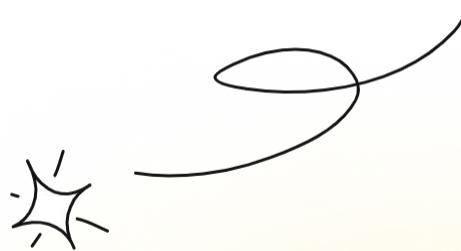
```
x = 10 # 1010 в двоичной системе
y = 5 # 0101 в двоичной системе

print(x & y) # Выводит 0
print(x | y) # Выводит 15
print(x ^ y) # Выводит 15
print(x << 1) # Выводит 20
print(x >> 1) # Выводит 5
```

Стать IT-шником может каждый



Экспресс-опрос



Какие вы помните встроенные математические функции?

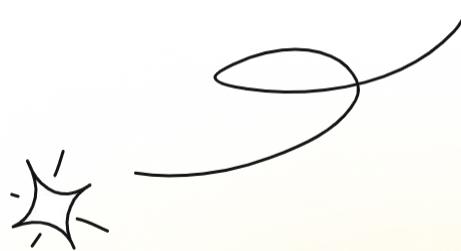
Встроенные математические функции

- `abs()`
 - возвращает абсолютное значение числа.
- `pow()`
 - возводит число в указанную степень.
- `round()`
 - округляет число до указанного количества десятичных знаков.
- `max()`
 - возвращает наибольшее значение из указанных.
- `min()`
 - возвращает наименьшее значение из указанных.
- `sum()`
 - возвращает сумму всех элементов в списке или кортеже чисел.

Стать IT-шником может каждый



Экспресс-опрос



Что такое приоритет операций?





Приоритет операций —

в Python операции имеют определенный приоритет, который определяет порядок их выполнения. Например, умножение и деление имеют более высокий приоритет, чем сложение и вычитание. Если нужно изменить порядок выполнения операций, можно использовать скобки.



Приоритет операций

Операторы	Применение
{ } ()	Скобки (объединение)
f(args...)	Вызов функции
x[index:index]	Срез
x[index]	Получение по индексу
x.attribute	Ссылка на атрибут
**	Возведение в степень
~x	Побитовое нет
+x, -x	Положительное, отрицательное число
*, /, %	Умножение, деление, остаток
+, -	Сложение, вычитание
<<, >>	Сдвиг влево/вправо
&	Побитовое И
^	Побитовое ИЛИ НЕ
	Побитовое ИЛИ
in, not in, is, is not, <, <=, >, >=, <>, !=, ==	Сравнение, принадлежность, тождественность
not x	Булево НЕ
and	Булево И
or	Булево ИЛИ
lambda	Лямбда-выражение



ПРИМЕР

```
x = 2 + 3 * 4 # Умножение выполнится первым,  
затем сложение  
print(x) # Выводит 14
```

```
y = (2 + 3) * 4 # Сложение выполнится первым,  
затем умножение  
print(y) # Выводит 20
```

Стать IT-шником может каждый



Деление на ноль

- На ноль делить нельзя!
- При делении на ноль в Python возникает ошибка ZeroDivisionError



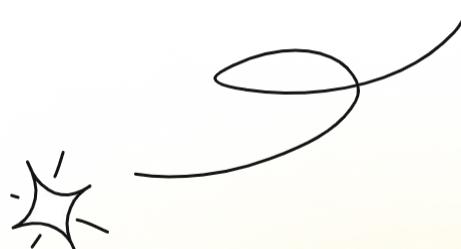
ПРИМЕР

```
x = 10
y = 0
print(x / y) # Вызовет ошибку ZeroDivisionError
```

Стать IT-шником может каждый



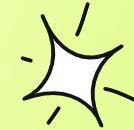
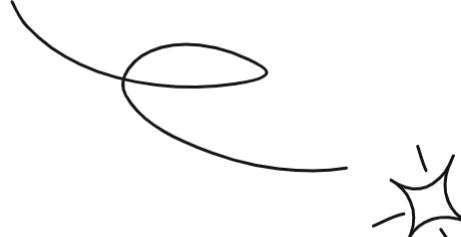
Экспресс-опрос



Что делают операции приращения?



Операторы приращения —
[модифицируют](#) исходную переменную.



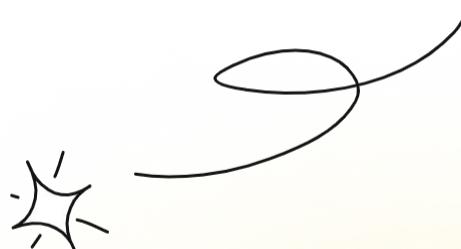
Операторы присваивания

- Сложение (`+=`)
- Вычитание (`-=`)
- Умножение (`*=`)
- Деление (`/=`)
- Целочисленное деление (`//=`)
- Взятие остатка от деления (`%=`)
- Возведение в степень (`**=`)

ПРИМЕР

```
x = 5
x += 2 # Эквивалентно x = x + 2
print(x) # Выводит 7
```

Экспресс-опрос



Как в Python называется логический тип данных?

Логический тип данных

В Python логический тип данных (**bool**) представляет собой тип, который принимает одно из двух возможных значений:

True (истина)

False (ложь)

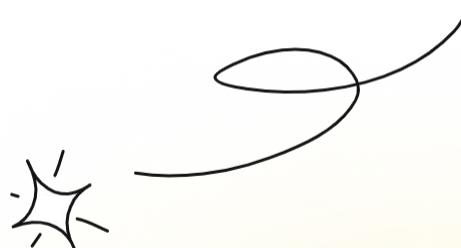
ПРИМЕР

```
x = True
y = False
print(x) # Выводит True
print(y) # Выводит False
```

Стать IT-шником может каждый



Экспресс-опрос



Какие операторы сравнения вы помните?





Операторы сравнения

`==`
(равно)

`!=`
(не равно)

`>`
(больше)

`<`
(меньше)

`>=`
**(больше или
равно)**

`<=`
**(меньше или
равно)**

Проверяет, равны ли два значения

Проверяет, не равны ли два значения

Проверяет, является ли первое значение большим, чем второе

Проверяет, является ли первое значение меньшим, чем второе

Проверяет, является ли первое значение большим или равным второму

Проверяет, является ли первое значение меньшим или равным второму

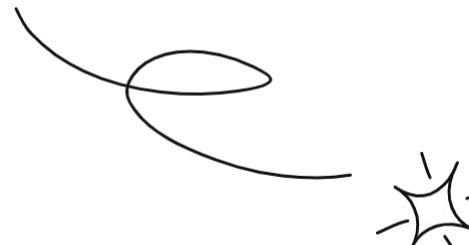


ПРИМЕР

```
x = 5
y = 10
print(x == y) # Выводит False
print(x < y) # Выводит True
```

None —

это специальное значение в Python, которое означает отсутствие значения. Оно часто используется, чтобы указать на то, что переменная не имеет никакого значения.



Проверка на None

Для проверки на None нужно использовать специальный оператор `is`.

`a = None`

`a is None`

`a is not None`

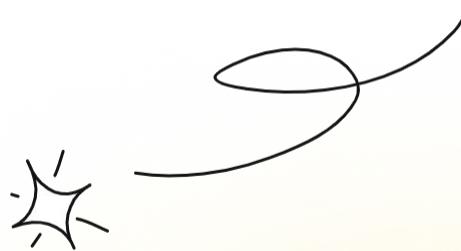
ПРИМЕР

```
x = None
if x is None:
    print("Переменная x не имеет значения")
```

Стать IT-шником может каждый



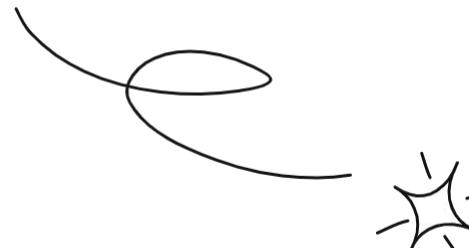
Экспресс-опрос



Что вы помните про булеву логику?

Булева логика —

это система логических операций, которые работают с логическими значениями.



& (and): Логический оператор "и"

- Возвращает True, если оба операнда являются истинными
- False в противном случае

True & True = True

False & True = False

True & False = False

False & False = False



| (or): Логический оператор "или"

- Возвращает True, если хотя бы один из operandов истинен
- False, если оба операнда ложны

○ ●

True | True = True

False | True = True

True | False = True

False | False = False

! (not): Логический оператор "не"

Инвертирует булевое значение:

- если operand истинен, то результат будет False
- если operand ложен, то результат будет True

$! \text{True} = \text{False}$

$! \text{False} = \text{True}$



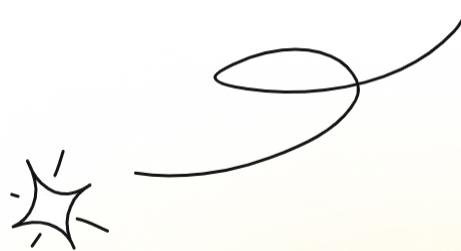
ПРИМЕР

```
x = True
y = False
print(x and y)  # Выводит False
print(x or y)   # Выводит True
print(not x)    # Выводит False
```

Стать IT-шником может каждый



Экспресс-опрос



Для чего нужны Законы де Моргана?



Законы де Моргана —

это набор правил, которые позволяют упростить и переписать сложные логические выражения.



Законы де Моргана

$$\begin{aligned} \text{not } (\text{A and B}) &== \text{not A or not B} \\ \text{not } (\text{A or B}) &== \text{not A and not B} \end{aligned}$$



Таблица истинности —

это таблица, которая показывает результаты логических операций для всех возможных комбинаций значений.

Таблица истинности позволяет легко понять результаты операций and, or, not и их комбинаций.



Таблицы истинности: конъюнкция

Конъюнкция		
A	B	$A \wedge B$
1	1	1
1	0	0
0	1	0
0	0	0

- Логическое умножение
- И
- Обозначается $A \wedge B$ или $A \& B$
- Читается
A и B

Таблицы истинности: дизъюнкция

Дизъюнкция		
A	B	$A \vee B$
1	1	1
1	0	1
0	1	1
0	0	0

- Логическое сложение
- ИЛИ
- Обозначается $A \vee B$
- Читается
A или B

Таблицы истинности: инверсия

Инверсия	
A	-A
0	1
1	0

- Логическое отрицание
- НЕ
- Обозначается **-A или A**
- Читается
не A

Таблицы истинности: ИМПЛИКАЦИЯ

Импликация		
A	B	$A \rightarrow B$
1	1	1
1	0	0
0	1	1
0	0	1

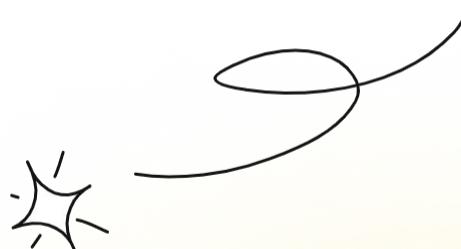
- Логическое следование
- ЕСЛИ...,ТО...
- Обозначается $A \rightarrow B$
- Читается
если A,то B
или
из A следует B

Таблицы истинности: эквивалентность

Эквивалентность		
A	B	$A \leftrightarrow B$
1	1	1
1	0	0
0	1	0
0	0	1

- Логическая равнозначность
- ТОГДА И ТОЛЬКО ТОГДА
- Обозначается $A \leftrightarrow B$
- Читается:
 A тогда и только тогда, когда B или
 A эквивалентно B

Экспресс-опрос



Какая функция возвращает уникальный идентификатор объекта?



ФУНКЦИЯ id —

это [функция](#), которая возвращает уникальный идентификатор объекта. Может использоваться для сравнения и проверки идентичности объектов.





Уникальный идентификатор —

это **числовое значение**, которое однозначно идентифицирует каждый объект в памяти компьютера.



ПРИМЕР

```
x = 5
print(id(x)) # Выводит адрес объекта в памяти
```



ПРИМЕР

```
a = 4
b = 4
c = 5
id(a) == id(b)    # True
id(a) == id(c)    # False
a += 1
id(a) == id(b)    # False
id(a) == id(c)    # True
```



Двойные неравенства



- Двойные неравенства используют для проверки, что значение находится в определенном диапазоне.
- Двойные неравенства позволяют объединять несколько условий в одном выражении.

Стать IT-шником может каждый



ПРИМЕР

```
x < y < z  
x >= y >= z
```



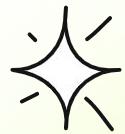
ПРИМЕР

```
x = 10
if 5 < x < 15:
    print("Значение x находится в диапазоне от 5 до 15")
```

Стать IT-шником может каждый



ВОПРОСЫ



Стать IT-шником может каждый



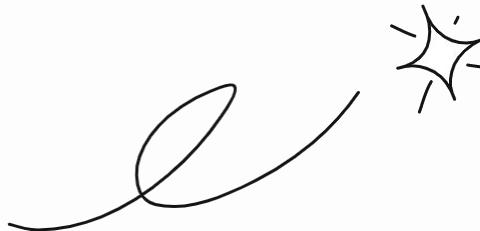
ЗАДАНИЕ ДЛЯ ЗАКРЕПЛЕНИЯ



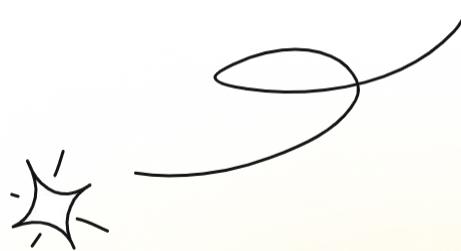


Отправь скриншот в чат:

1. Напишите программу, которая посчитает разность между числами 6 и -81 и выведет ответ на экран.
2. Напишите программу, которая считает и выводит на экран последовательно (по одному значению в каждой строчке) значения следующих математических выражений: «3 в степени 5» и «-8 разделить на -4».



Экспресс-опрос



Что такое условие в программировании?

Какой условный оператор позволяет выполнить определенный блок кода,
если условие истинно, и выполнить другой блок кода, если условие ложно?



Условие в программировании —

это выражение или логическое выражение, которое оценивается как истина или ложь. Условия используются для принятия решений в программе и выполнения определенных действий в зависимости от их истинности или ложности.





Условный оператор if-else —

позволяет выполнить определенный блок кода, если условие истинно, и выполнить другой блок кода, если условие ложно.





ПРИМЕР

```
if условие:  
    # выполняемые действия, если условие истинно  
else:  
    # выполняемые действия, если условие ложно
```



ПРИМЕР

```
x = 5
if x > 10:
    print("x больше 10")
else:
    print("x меньше или равно 10")
```

Экспресс-опрос



Какое ключевое слово используется в условном операторе для проверки дополнительных условий после оператора if?



Ключевое слово elif —

используется в условном операторе для проверки дополнительных условий после оператора if. Оно позволяет проверить несколько условий последовательно и выполнить соответствующий блок кода при первом истинном условии.





ПРИМЕР

```
if условие1:  
    # выполняемые действия, если условие1 истинно  
elif условие2:  
    # выполняемые действия, если условие2 истинно  
else:  
    # выполняемые действия, если все условия ложны
```

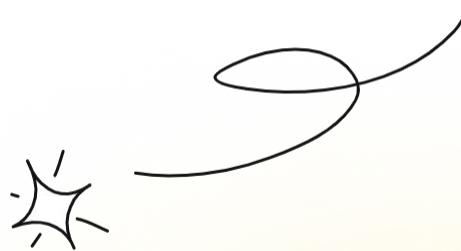
ПРИМЕР

```
x = 5
if x > 10:
    print("x больше 10")
elif x > 5:
    print("x больше 5, но не больше 10")
else:
    print("x меньше или равно 5")
```

Стать IT-шником может каждый



Экспресс-опрос



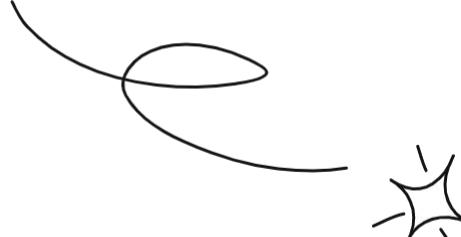
Для чего нужны вложенные условия?





Вложенные условия —

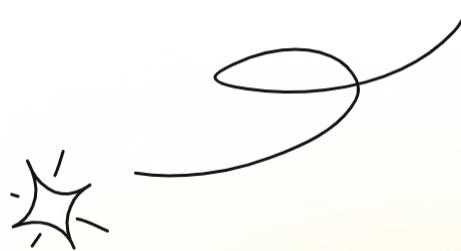
это [условные операторы](#), которые находятся внутри других условных операторов. Они позволяют создавать более сложные проверки и выполнять различные блоки кода в зависимости от их истинности.



ПРИМЕР

```
x = 5
if x > 0:
    if x < 10:
        print("x находится в диапазоне от 0 до
10")
    else:
        print("x больше или равно 10")
else:
    print("x меньше или равно 0")
```

Экспресс-опрос



К какому логическому типу могут быть приведены все значения?



Приведение типов к логическому



- В Python все значения могут быть приведены к логическому типу `bool`.
- Любое ненулевое значение или непустой объект рассматриваются как истинное, а ноль или пустой объект - как ложное.



ПРИМЕР

```
if условие:  
    # выполняемые действия, если условие истинно  
else:  
    # выполняемые действия, если условие ложно
```

ПРИМЕР

```
x = 0
if x:
    print("x истинно")
else:
    print("x ложно")
```

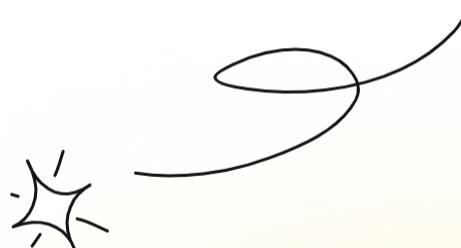
Важность использования табов/пробелов

- Используются 4 пробела для каждого уровня отступа.
- Использование табов и пробелов вместе в одном скрипте может привести к ошибкам.

Стать IT-шником может каждый



Экспресс-опрос



Чем полезен тернарный оператор





Тернарный оператор —

позволяет сократить условное выражение до одной строки.



ПРИМЕР

```
выражение1 if условие else выражение2
```



ПРИМЕР

```
x = 5
message = "x больше 10" if x > 10 else "x меньше или равно
10"
print(message)
```

Стать IT-шником может каждый

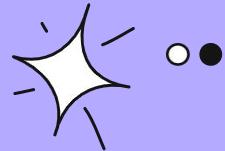


РАЗБОР ДОМАШНЕГО ЗАДАНИЯ

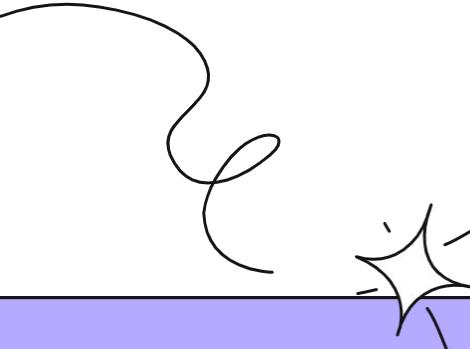


Домашнее задание 1

1. Создать Jupyter Notebook, в ячейке написать $1+1$, выполнить ячейку. Убедиться, что кроме числа 2 не вывелось ни ошибок, ни предупреждений. Закрыть Jupyter Notebook.
1. Вывести на экран названия предметов, которые у будут на этой неделе на курсе.



Домашнее задание 2

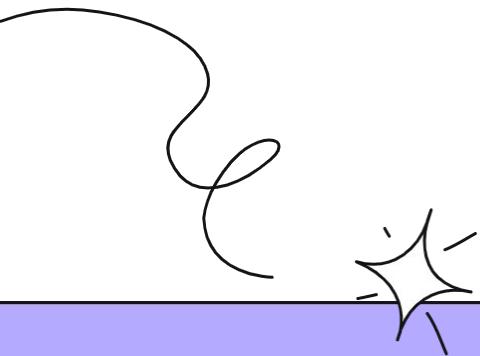


1. Напишите программу, которая запрашивает у пользователя его имя и возраст, а затем выводит на экран приветствие в формате: "Привет, [имя]! Тебе [возраст] лет."
Используйте команду `input` для получения данных от пользователя и команду `print` для вывода приветствия.

Пример вывода:

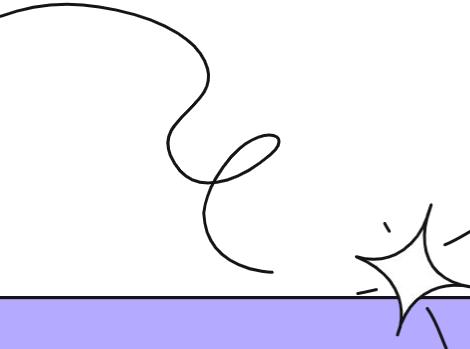
```
Введите ваше имя: Алексей
Введите ваш возраст: 25
Привет, Алексей! Тебе 25 лет.
```

Домашнее задание 2

- 
- 2. Переведите из двоичной системы в десятичную число 1100110 и напишите только ответ.
 - 3. Переведите из десятичной системы в двоичную число 127 и напишите только ответ.



Домашнее задание 3



1. Давайте создадим простой калькулятор. Напишите программу, которая запрашивает у пользователя два целых числа и выполняет следующие действия:



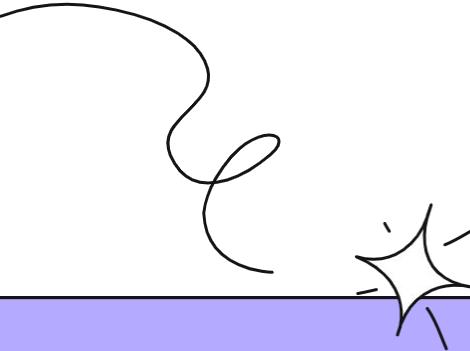
- Вычисляет сумму и выводит результат.
- Вычисляет разность и выводит результат (первое число минус второе число).
- Вычисляет произведение и выводит результат.
- Вычисляет частное (результат деления первого числа на второе) и выводит результат.
- Вычисляет остаток от деления первого числа на второе и выводит результат.
- Возводит первое число в степень второго числа и выводит результат.

Домашнее задание 3

```
a=int(input("Введите первое число: "))
b=int(input("Введите второе число: "))
print("Сумма:",a+b)
print("Разность:",a-b)
print("Произведение:",a*b)
print("Частное:",a/b)
print("Целая часть от деления:",a//b)
print("Остаток от деления:",a%b)
print("Степень:",a**b)
```

Введите первое число: 78
Введите второе число: 5
Сумма: 83
Разность: 73
Произведение: 390
Частное: 15.6
Целая часть от деления: 15
Остаток от деления: 3
Степень: 2887174368

Домашнее задание 3



2. Есть круглая арена цирка, нам известен ее радиус, нужно подобрать покрытие для этой арены, чтобы это сделать необходимо узнать длину и площадь окружности. Напишите программу, которая запрашивает у пользователя радиус окружности и выполняет следующие действия:
- Вычисляет длину окружности по формуле $2 * \pi * r$, где π – математическая константа (примерно равна 3.14159), а r – радиус окружности. Выводит результат.
 - Вычисляет площадь окружности по формуле $\pi * r^2$, где π – математическая константа, а r – радиус окружности. Выводит результат.

Домашнее задание 3

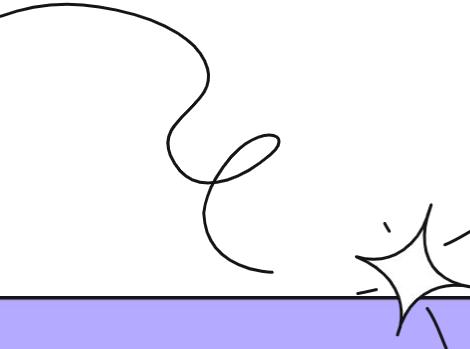
```
pi=3.14159
r=float(input("Введите число"))
print("Длина окружности",2*pi*r)
print("Площадь окружности",pi*r**2)
```

Введите число 4.5

Длина окружности 28.27431

Площадь окружности 63.61719749999996

Домашнее задание 3



3. Географ хочет находить расстояние между двумя точками на карте. Напишите для него программу, которая запрашивает у пользователя координаты двух точек в двумерном пространстве (x_1, y_1) и (x_2, y_2) , а затем вычисляет и выводит расстояние между этими точками по формуле:

```
distance = sqrt((x2 - x1)^2 + (y2 - y1)^2)
, где sqrt - функция извлечения квадратного корня.
```

- Не используйте встроенную математическую функцию `sqrt` для вычисления корня.
- Не забывайте, что `sqrt(x)==x**0.5`.

Результат должен быть выведен с помощью команды `print`.

Домашнее задание 3

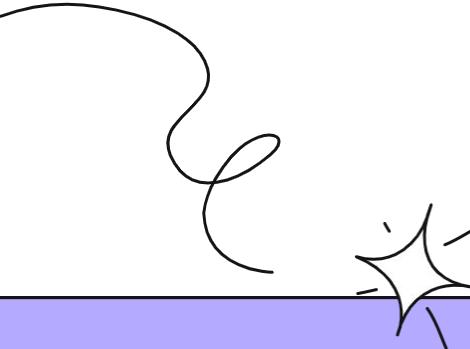
```
x1, y1=input("Введите координаты первой точки:").split()
x2, y2=input("Введите координаты второй точки:").split()
x1=int(x1)
x2=int(x2)
y1=int(y1)
y2=int(y2)
print("Расстояние между этими точками равно", ((x2-x1)**2+(y2-y1)**2)**0.5)
```

Введите координаты первой точки: 1 3

Введите координаты второй точки: 4 5

Расстояние между этими точками равно 3.605551275463989

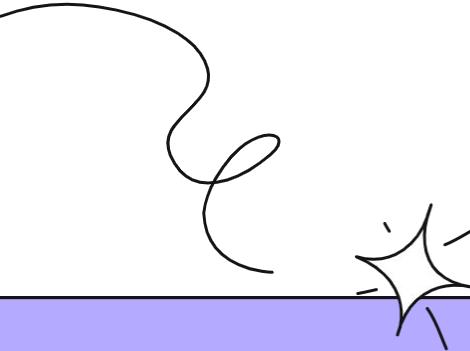
Домашнее задание 4



1. Даны формулы: $\neg((A \vee B) \wedge (C \vee D))$ и $((\neg A \wedge \neg B) \vee (\neg C \wedge \neg D))$.
Используя закон Де Моргана, докажите, что эти формулы эквивалентны.



Домашнее задание 4



Формулы: $\neg((A \vee B) \wedge (C \vee D))$ и $((\neg A \wedge \neg B) \vee (\neg C \wedge \neg D))$

Применяем закон Де Моргана к первой формуле:

$$\begin{aligned}\neg((A \vee B) \wedge (C \vee D)) &\equiv \neg(A \vee B) \vee \neg(C \vee D) \\ &\equiv ((\neg A \wedge \neg B) \vee (\neg C \wedge \neg D))\end{aligned}$$

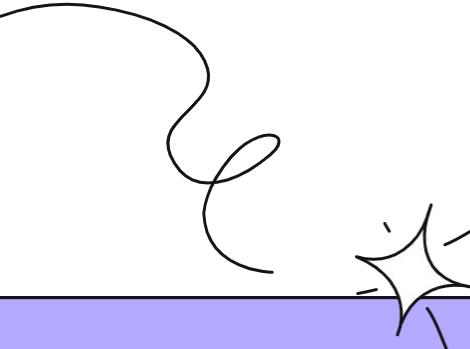
Получаем:

$$((\neg A \wedge \neg B) \vee (\neg C \wedge \neg D)) \text{ и } ((\neg A \wedge \neg B) \vee (\neg C \wedge \neg D))$$

Обратно применяем закон Де Моргана ко второй формуле:

$$((\neg A \wedge \neg B) \vee (\neg C \wedge \neg D)) \equiv ((\neg A \wedge \neg B) \vee (\neg C \wedge \neg D))$$

Домашнее задание 4



2. Напишите программу, которая запрашивает у пользователя два логических значения (True или False) и выводит результаты следующих логических операций:

○ ●

- Логическое И (and) между двумя значениями.
- Логическое ИЛИ (or) между двумя значениями.
- Логическое НЕ (not) для каждого значения.
- Результат сравнения двух значений на равенство.
- Результат сравнения двух значений на неравенство.

Результаты должны быть выведены с помощью команды print.

Домашнее задание 4

```
a=input("Введите значение а (True или False):")
b=input("Введите значение b (True или False):")
if a=="True": #не забываем, что input принимает строку, а не логическое
    значение!
```

```
    a=True
```

```
if a=="False":
```

```
    a=False
```

```
if b=="True":
```

```
    b=True
```

```
if b=="False":
```

```
    b=False
```

```
print("a and b =", a and b)
```

```
print("a or b =", a or b)
```

```
print("not a =", not a)
```

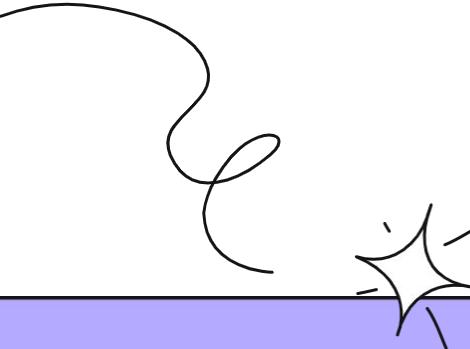
```
print("not b =", not b)
```

```
print("a == b =", a==b)
```

```
print("a != b =", a!=b)
```



Домашнее задание 5



1. Напишите программу, которая запрашивает у пользователя три числа и выводит их в порядке возрастания, разделенные запятой.
Используйте условные операторы и вложенные условия для решения задачи.
Предполагается, что все три числа различны.



Пример вывода:

Введите первое число: 5

Введите второе число: 2

Введите третье число: 7

Числа в порядке возрастания: 2, 5, 7



Домашнее задание 5

```
a=int(input())
b=int(input())
c=int(input())

if a>=b>=c:
    print("Числа в порядке возрастания: ", c,"", b,"", a)
elif b>=a>=c:
    print("Числа в порядке возрастания: ", c,"", a,"", b)
elif c>=b>=a:
    print("Числа в порядке возрастания: ", a,"", b,"", c)
elif a>=c>=b:
    print("Числа в порядке возрастания: ", b,"", c,"", a)
elif c>=a>=b:
    print("Числа в порядке возрастания: ", b,"", a,"", c)
else:
    print("Числа в порядке возрастания: ", a,"", c,"", b)
```



Домашнее задание 5

2. Напишите программу, которая запрашивает у пользователя год и проверяет, является ли он високосным.



Год является високосным, если он делится на 4 без остатка, но не делится на 100 без остатка, за исключением годов, которые делятся на 400 без остатка.

Выведите соответствующее сообщение на экран с помощью команды print.

Пример вывода:

Введите год: 2024

Год является високосным.



Домашнее задание 5



```
year = int(input("Введите год: "))
if (year % 4 == 0 and year % 100 != 0) or year % 400 == 0:
    print("Високосный")
else:
    print("Невисокосный")
```

Стать IT-шником может каждый



ОСТАВШИЕСЯ ВОПРОСЫ



Стать IT-шником может каждый



ЗАКЛЮЧЕНИЕ

