IT CAREER HUB

Python

Базовые типы данных, работа с функциями

Преподаватель

Портрет

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
- В течение занятия вопросы задавать в чате или когда преподаватель спрашивает, есть ли у Вас вопросы
- Вести себя уважительно и этично по отношению к остальным участникам занятия
- Организационные вопросы по обучению решаются с кураторами, а не на тематических занятиях
- Во время занятия будут интерактивные задания, будьте готовы включить камеру или демонстрацию экрана по просьбе преподавателя

Повторение

- Особенности языка Python
- Интерпретатор и компилятор
- Переменная и оператор присваивания
- Правила именования переменных
- Синтаксис, ошибки синтаксиса, комментарии и PEP 8
- Функция, вызов функции
- Функция print и функция input

План занятия

- Тип данных, объект, динамическая типизация
- Примитивные типы данных
- Возвращаемое значение
- Процедуры в Python, передача аргументов в функцию
- Функция type
- Строковые операции
- Работа с кавычками в строках
- Экранирование символов
- Параметры sep и end

ICH IT CAREER HUB

ОСНОВНОЙ БЛОК

ICH IT CAREER HUB

Тип данных, объект, динамическая типизация

Тип данных

Определяет, какие значения может хранить объект и какие операции можно выполнять с этими значениями.

Объект

Объект в Python — это любое значение, с которым можно работать в программе.

Полезно знать

Объект

Всё в Python — это объект: числа, строки, списки и т.д.

Хранение

Каждый объект хранит данные и имеет определённый **тип**, который определяет, что это за данные и какие операции можно с ними выполнять.

Динамическая типизация

Python является динамически типизированным языком, что означает, что тип переменной определяется автоматически в момент присваивания значения, и он может меняться в процессе выполнения программы.

Полезно знать

Объяснение

Тип данных присваивается не переменной, а объекту, на который она ссылается. Это позволяет одной и той же переменной в ходе программы ссылаться на объекты разных типов данных.

Пример

```
price = 10  # Тип автоматически определен как int

has_passed = 9.99 # Тип автоматически
# переопределен как float
```

IT CAREER HUB

ВОПРОСЫ

ICH IT CAREER HUB

Примитивные типы данных

Полезно знать

ICH
IT CAREER HUB

Примитивные типы данных

Используются для хранения простых значений.

Целые числа (int)

Определение

Целые числа представляют собой значения без дробной части. Они могут быть положительными, отрицательными или равны нулю.

Пример

```
x = 5    # Положительное целое число
y = -3   # Отрицательное целое число
z = 0    # Ноль
```

Числа с плавающей запятой (float)

Определение

Числа с плавающей запятой представляют собой значения с дробной частью.

Пример

```
pi = 3.14           # Число с плавающей запятой
temperature = -5.2  # Отрицательное число с
# плавающей запятой
```

Экспресс-опрос

? Чем целое число отличается от числа с плавающей точкой?

Строки (str)

Определение

Строки представляют собой упорядоченные последовательности символов, заключённых в одинарные или двойные кавычки, а также в тройные кавычки (как одинарные, так и двойные).

Пример

```
# Одинарные кавычки
single_quote_string = 'Это строка в одинарных кавычках.'

# Двойные кавычки
double_quote_string = "Это строка в двойных кавычках."
```

Строки (str)

Определение

Тройные кавычки позволяют создавать многострочные строки, сохраняющие исходное форматирование, включая переносы строк.

Пример

```
# Одинарные тройные кавычки (многострочная
# строка)
triple_quote_string = '''Это строка в
тройных кавычках, которая
может занимать несколько строк.'''

# Альтернативный вариант с двойными
# тройными кавычками
another_triple_quote_string = """Это ещё
одна многострочная строка, но с
использованием двойных тройных кавычек."""
```

Экспресс-опрос

? Зачем в строковом типе используется разное число кавычек?

Логические значения (bool)

Определение

Логические значения представляют два возможных состояния: True (истина) и False (ложь). Они записываются текстом с большой буквы без кавычек.

Пример

```
is_student = True
# Логическое значение True
has_passed = False
# Логическое значение False
```

Ответьте используя булеву логику

?

Идёт ли сейчас урок?

Отсутствие значения (NoneType)

Определение

Тип NoneType представляет отсутствие значения.

Единственное значение для этого типа: None

Пример

```
value = None
# Переменная, не содержит никакого значения
```

Предположите

?

Зачем может применяться отсутствие значения (NoneType)?

Полезно знать

ICH
IT CAREER HUB

Отсутствие значения

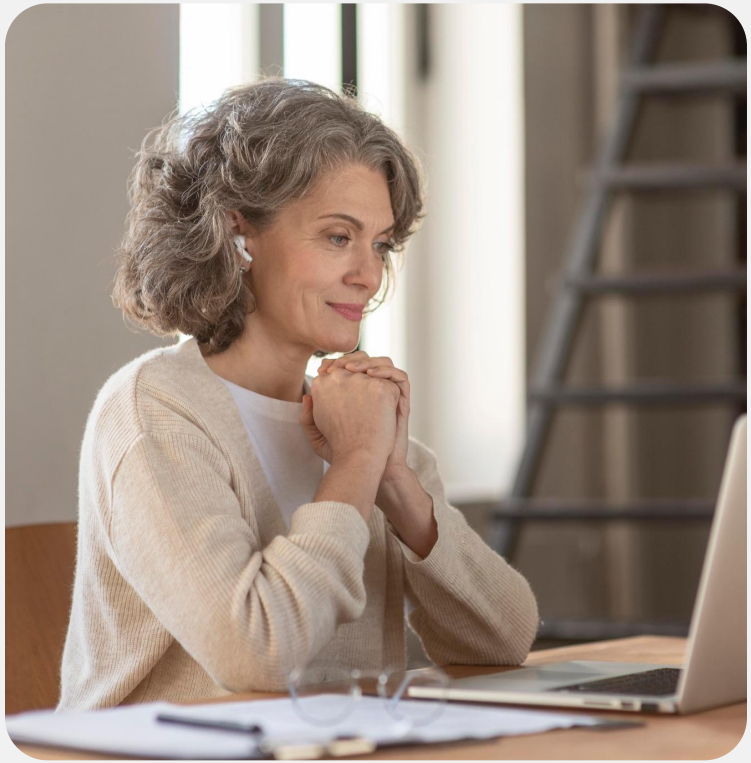
Значение `None` часто используется для инициализации переменных, когда значение ещё не известно, или для обозначения того, что функция ничего не возвращает.

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

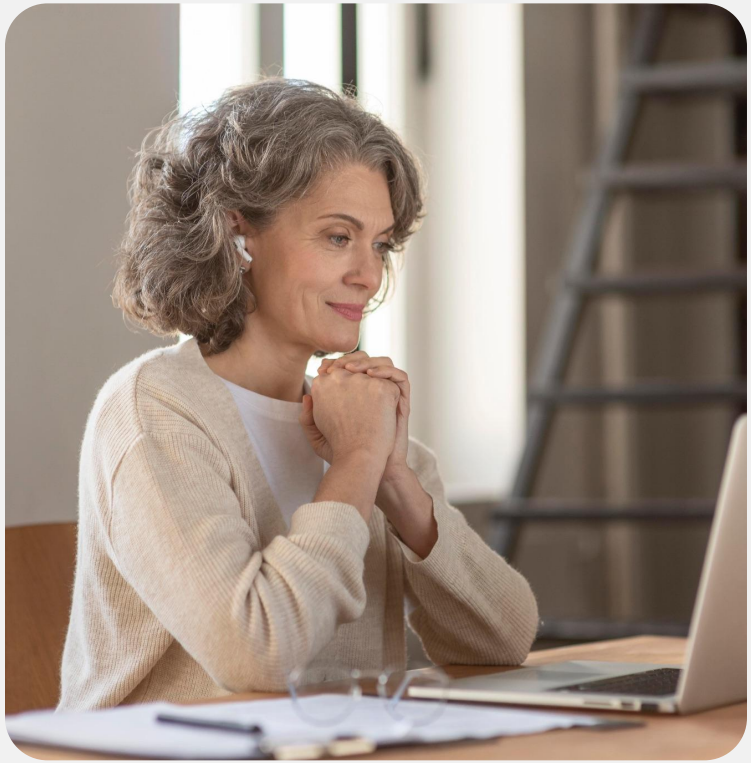
ЗАДАНИЕ



Выберите правильный вариант ответа

Какой тип данных будет у переменной x, если присвоить ей значение x = 5.0?

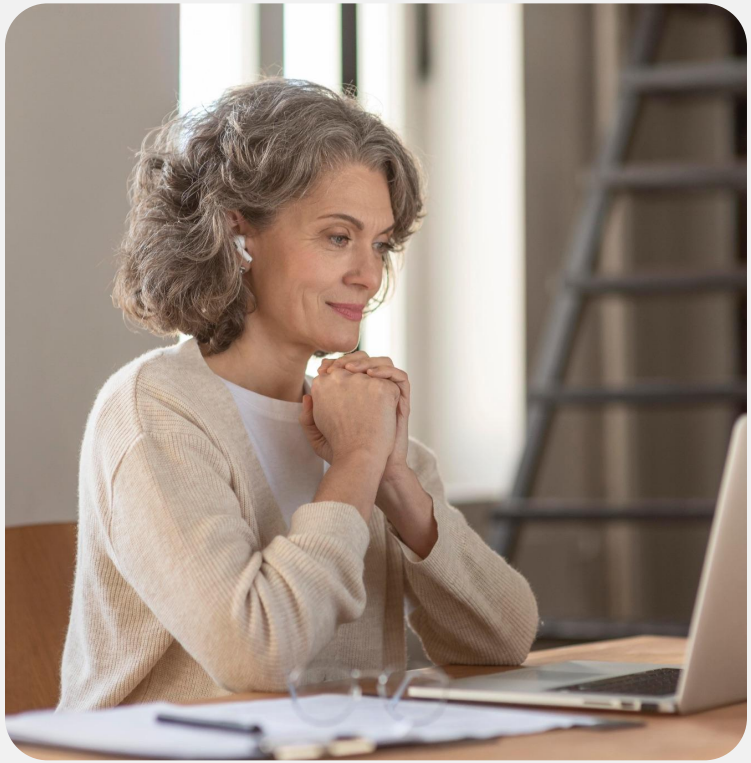
a. str
b. bool
c. int
d. float



Выберите правильный вариант ответа

Какой тип данных будет у переменной x, если присвоить ей значение x = 5.0?

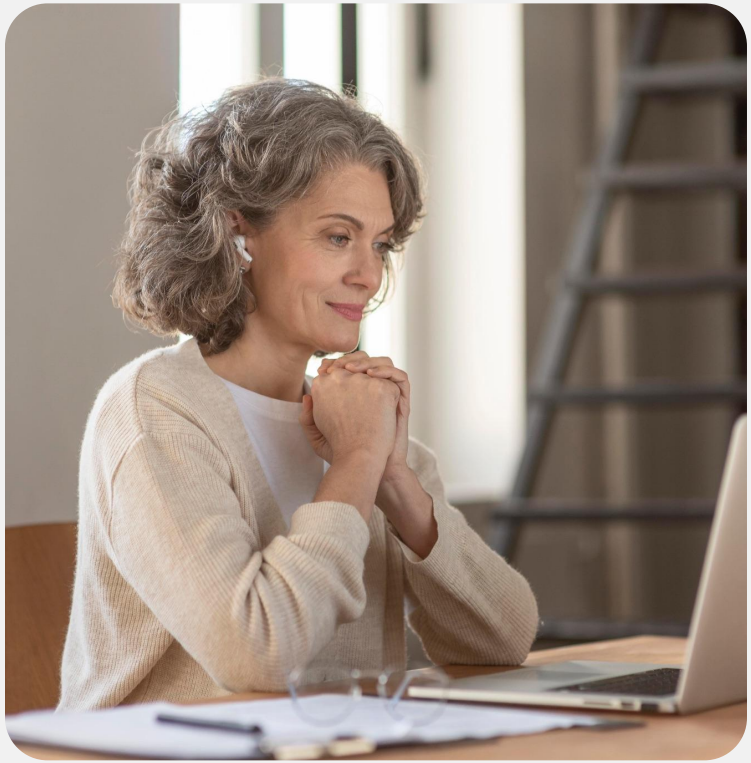
a. str
b. bool
c. int
d. float



Выберите правильный вариант ответа

Что из перечисленного является логическим значением?

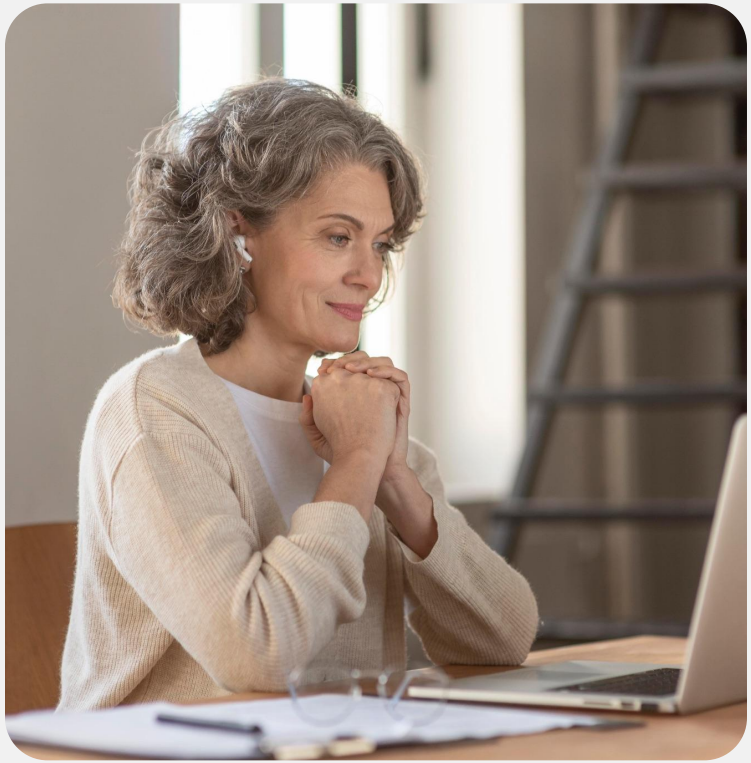
a. "True"
b. 0
c. True
d. None



Выберите правильный вариант ответа

Что из перечисленного является логическим значением?

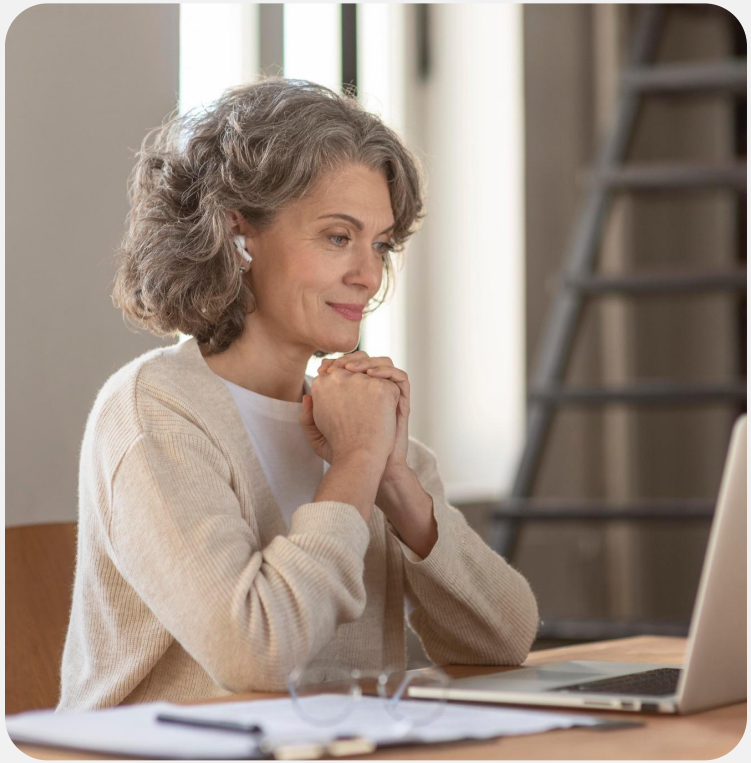
a. "True"
b. 0
c. True
d. None



Выберите правильный вариант ответа

Что произойдет, если переменной сначала присвоить целое число, а затем строку?

a. Возникнет ошибка
b. Тип переменной изменится автоматически
c. Программа завершится
d. Значение переменной будет преобразовано в int



Выберите правильный вариант ответа

Что произойдет, если переменной сначала присвоить целое число, а затем строку?

a. Возникнет ошибка
b. Тип переменной изменится автоматически
c. Программа завершится
d. Значение переменной будет преобразовано в int

IT CAREER HUB

Возвращаемое значение

Полезно знать

ICH
IT CAREER HUB

Возвращаемое значение

Это результат, который функция передаёт обратно в место своего вызова.

Процедуры в Python

Важная информация

Термин процедура часто используется для описания функций, которые ничего не возвращают (или возвращают None) и выполняют определённые действия, такие как вывод на экран или модификация данных.

Пример

```
print("Привет, мир!")  # Выводим сообщение, но
# print не возвращает результат
```

Результат None

Важная информация

Если попытаться вывести результат работы процедуры, результат всегда будет None.

Пример

```
value = print("Привет, мир!")  # Выводит
# сообщение "Привет, мир!", но print не
# возвращает результат
print(value)                   # Выводит None
```

IT CAREER HUB

Передача аргументов в функцию

Полезно знать

IT CAREER HUB

Передача аргументов в функцию

В функцию можно передавать аргументы разными способами.

Передача аргументов в функцию

Способ

Значения (литералы), например, числа или строки

Пример

```
print(5)
print("Привет, мир!")
```

Передача аргументов в функцию

Способ

Переменные, из которых будут получены значения

Пример

```
text = "Привет, мир!"
print(text)
```

Передача аргументов в функцию

Способ

Результат выполнения математических операций

Пример

```
print(2 + 5)
```

Передача аргументов в функцию

Способ

Результаты выполнения других функций

Пример

```
print(input("Введите имя: "))
```

ICH IT CAREER HUB

Функция type

Функция type

Определение

Функция `type` используется для определения типа объекта. Она возвращает класс типа объекта.

Пример

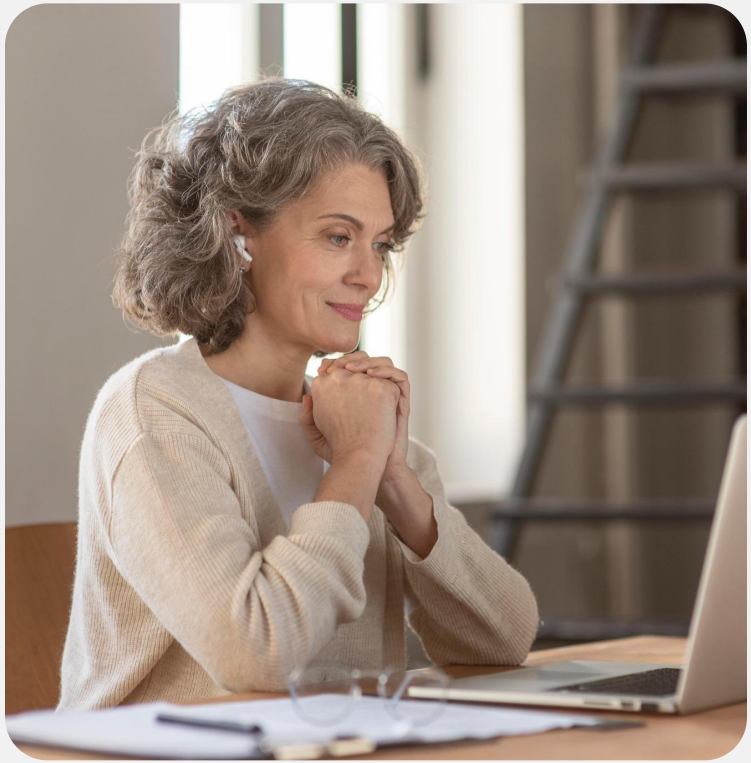
```
x = 10
print(type(x))  # <class 'int'>
x = "Привет"
print(type(x))  # <class 'str'>
```

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

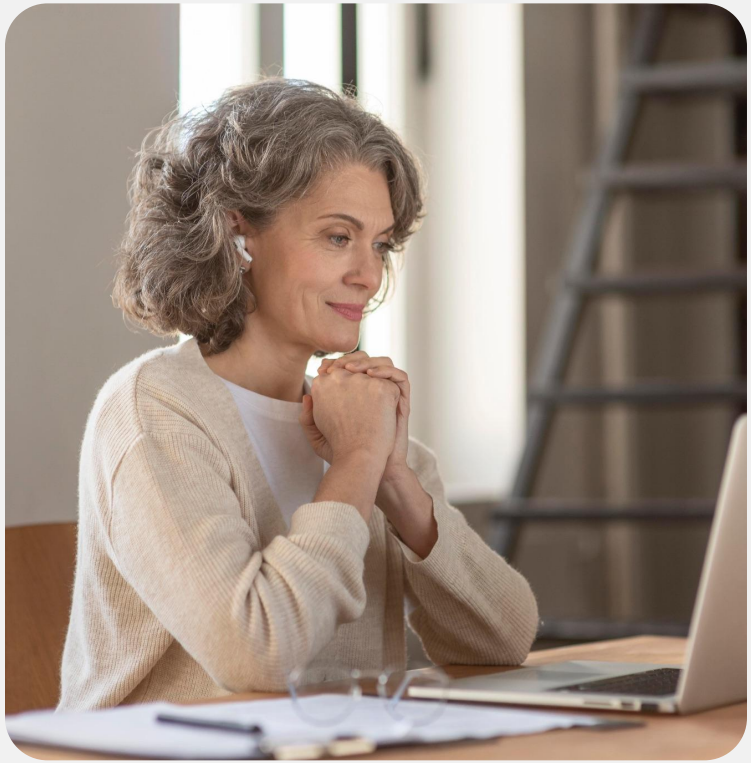
ЗАДАНИЕ



Выберите правильный вариант ответа

Что произойдёт, если передать переменную в функцию `print()`?

a. Будет выведено имя переменной
b. Будет выведено значение переменной
c. Возникнет ошибка
d. Будет выведен тип переменной



Выберите правильный вариант ответа

Что произойдёт, если передать переменную в функцию print()?

a. Будет выведено имя переменной
b. Будет выведено значение переменной
c. Возникнет ошибка
d. Будет выведен тип переменной

IT CAREER HUB

Строковые операции

Конкатенация

Определение

Конкатенация — это процесс объединения двух или более строк в одну строку. Для этого используется оператор +.

Пример

```
str1 = "Привет"
str2 = "мир"
result = str1 + " " + str2  # Объединение строк с
# пробелом между ними
print(result)
```

Умножение строк

Определение

Умножение строк позволяет создать новую строку, состоящую из повторений исходной строки.

Пример

```
str1 = "Hi! "
result = str1 * 3  # Создание строки из трёх
# повторений исходной строки
print(result)
```

Экспресс-опрос

?

В какой ситуации может быть полезно умножение строк?

IT CAREER HUB

Работа с кавычками в строках

Полезно знать

IT CAREER HUB

Кавычки в Python

В Python строки можно заключать в одинарные (') или двойные кавычки ("), что позволяет гибко работать с текстом.

Одинарные и двойные кавычки

Рекомендация

Если внутри строки необходимо использовать кавычки, то целесообразно выбирать противоположные по типу кавычки для обрамления строки.

Пример

```
# Одинарные кавычки внутри строки
quote1 = "Она сказала: 'Привет!'"

# Двойные кавычки внутри строки
quote2 = 'Он ответил: "Привет!"'
```

Многострочные строки

Рекомендация

Для многострочных строк и строк, содержащих как одинарные, так и двойные кавычки, удобно использовать тройные кавычки (''' или """).

Пример

```
multi_line_string = """Это многострочная строка, в которой есть 'одинарные' и "двойные" кавычки."""
```

ICH IT CAREER HUB

Экранирование символов

Полезно знать

ICH
IT CAREER HUB

Экранирование

Это способ обработки специальных символов внутри строк.

Экранирование

В Python используется обратный слэш (\) для обозначения специальных символов и создания escape-последовательностей.

Экранирование кавычек

Рекомендация

Если строка обрамлена в одинарные кавычки, а внутри нужно использовать одинарную кавычку, её нужно экранировать. То же правило действует и для двойных кавычек.

Пример

```
# Одинарные кавычки внутри строки
string1 = 'В строке есть \'одинарные\' и "двойные" кавычки.'
print(string1)

# Двойные кавычки внутри строки
string2 = "В строке есть 'одинарные' и \"двойные\" кавычки."
print(string2)
```

Специальные символы

Рекомендация

Экранирование также используется для вставки специальных символов, таких как:

`\n` — новая строка

`\t` — табуляция

Пример

```
# Новая строка
text = "Первая строка\nВторая строка"
print(text)
# Табуляция
text_with_tab = "Первая строка\tВторая строка"
print(text_with_tab)
```

Экспресс-опрос

? Для чего используется специальный символ `\t`?

Экранирование обратного слэша

Рекомендация

Если в строке необходимо вставить обратный слэш, его тоже нужно экранировать, так как сам обратный слэш является специальным символом.

Пример

```
path = "C:\\Users\\Username\\Documents"
```

IT CAREER HUB

Параметры sep и end

Полезно знать

ICH
IT CAREER HUB

Параметры sep и end

Функция `print` имеет несколько необязательных параметров, которые позволяют настроить ее поведение.

Параметры sep и end

sep (разделитель)

Определяет строку, которая будет вставлена между аргументами. По умолчанию используется пробел.

```
print("one", "two", "three")              # one two three
print("one", "two", "three", sep="---")
# one---two---three
```

end (конец)

Определяет строку, которая будет добавлена в конец вывода. По умолчанию используется символ новой строки.

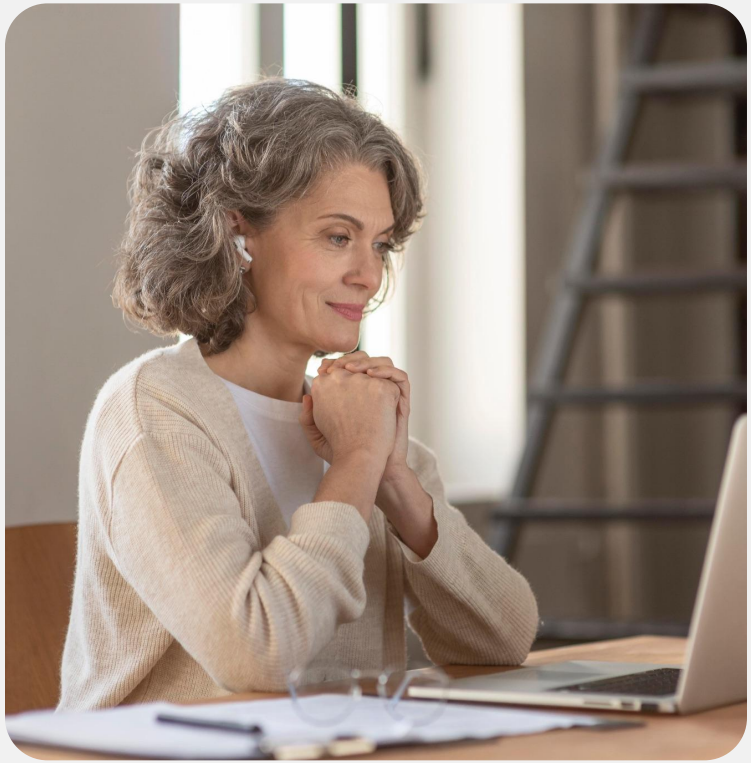
```
print("Привет,", "мир", end="!!")
print("Привет")
```

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

ЗАДАНИЕ

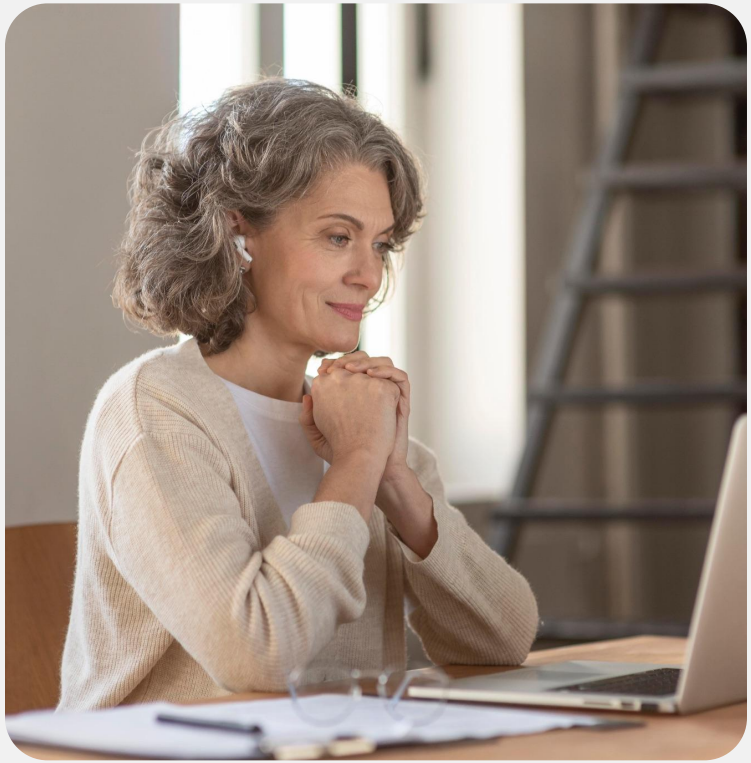


Выберите правильный вариант ответа

Какой результат выведет следующий код?

```
str1 = "Привет"
str2 = "мир"
result = str1 + str2
print(result)
```

a. Привет мир
b. Приветмир
c. Привет, мир
d. Ошибка

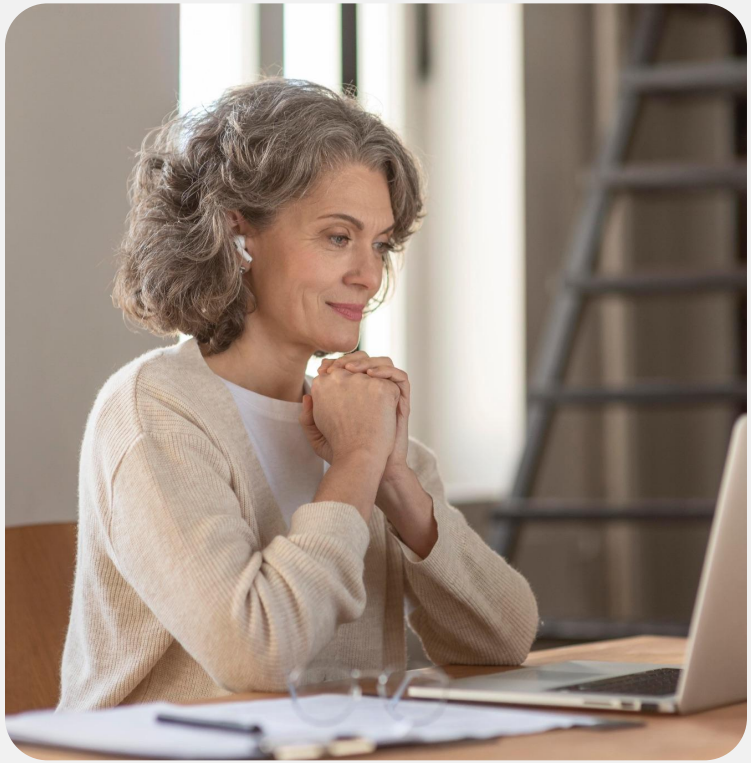


Выберите правильный вариант ответа

Какой результат выведет следующий код?

```
str1 = "Привет"
str2 = "мир"
result = str1 + str2
print(result)
```

a. Привет мир
b. Приветмир
c. Привет, мир
d. Ошибка

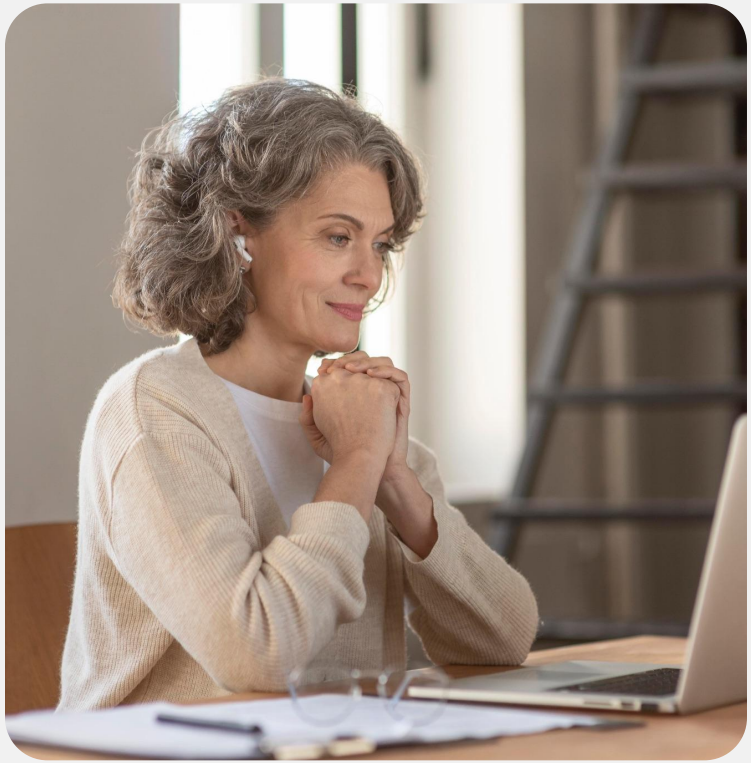


Выберите правильный вариант ответа

Какой результат выведет следующий код?

```
str1 = "Hi! "
result = str1 * 3
print(result)
```

a. Hi! Hi! Hi!
b. HiHiHi
c. Ошибка
d. Hi!



Выберите правильный вариант ответа

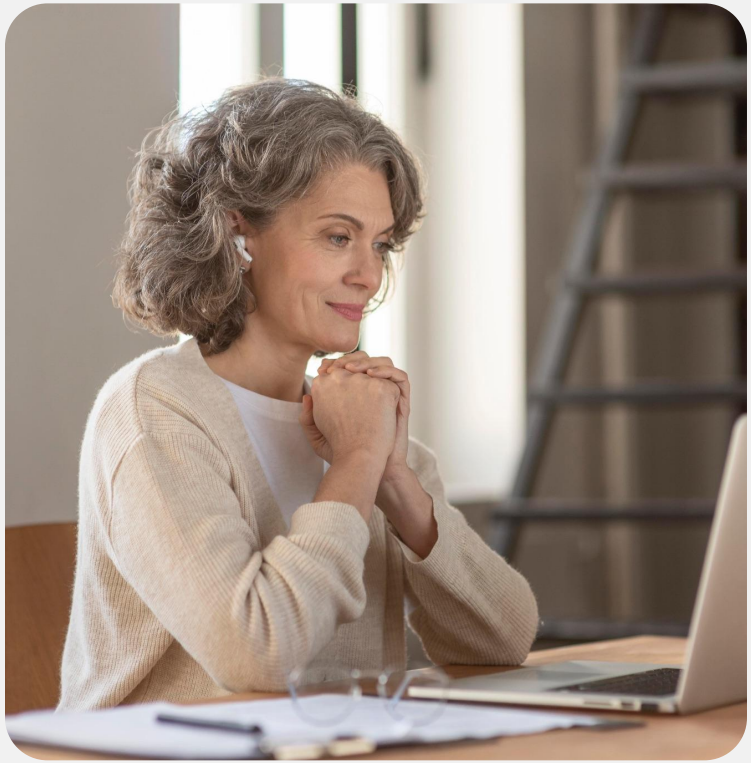
Какой результат выведет следующий код?

```
str1 = "Hi! "
result = str1 * 3
print(result)
```

a. **Hi! Hi! Hi!**
b. HiHiHi
c. Ошибка
d. Hi!

IT CAREER HUB

Практическая работа

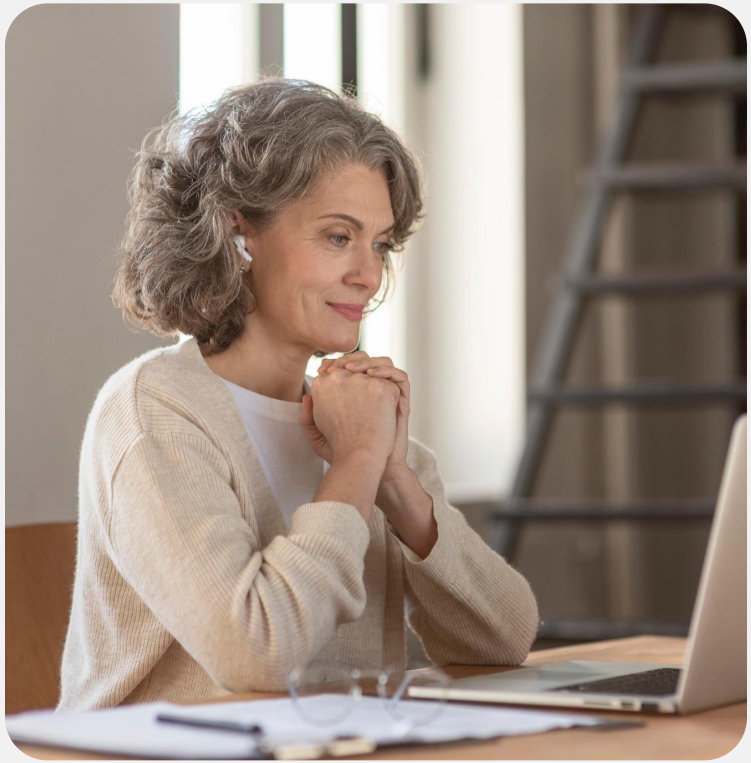


Приветствие с восклицанием

Напишите программу, которая выведет строку "Привет, мир!" с использованием параметра end, чтобы вывод закончился на восклицательном знаке вместо новой строки.

Пример вывода:

```
Привет, мир!
```



Цитата

Создайте программу, которая выведет строку: She said: "It's amazing!" двумя способами (одинаковый результат, но разный код).

Пример вывода:

```
She said: "It's amazing!"
```

```
She said: "It's amazing!"
```

IT CAREER HUB

Домашнее задание

Домашнее задание

1. **Числовая цепочка**

   Напишите программу, которая выводит числа от 1 до 5 включительно, разделяя их символами “---”. Используйте параметр sep.

   Пример вывода:

   1---2---3---4---5

Домашнее задание

2. Пробельные символы

Напишите программу, которая выведет две строки: первая строка с табуляцией, вторая — на новой строке с помощью одного вызова `print()`.

Пример вывода:

```
	Первая строка с табуляцией

Вторая строка на новой строке
```

Домашнее задание

3. Текст в кавычках

Напишите программу, которая выводит строку `"Это файл "example.txt""`.

Пример вывода:

```
Это файл "example.txt"
```

IT CAREER HUB

ВОПРОСЫ

Заключение