SKILLFACTORY

Итоговая аттестация по программе «Профессия "Веб-разработчик на Python"»

Создайте копию этого шаблона у себя на диске. После того, как вы ответите на все задания, скачайте документ в формате .pdf и приложите его в Форму для сдачи ответа на учебной платформе.

Приступим к тесту

Задание 1. Что выведется на экран?

```
quote = "Do not give up, the beginning is always the hardest."
print(quote [8])
```

i

Задание 2. Дайте определение понятиям «Список» и «Кортеж». Опишите, чем они отличаются.

Список и кортеж - это два различных типа данных во многих языках программирования, включая Python. Вот их определения и основные различия:

Список (List) представляет собой упорядоченную изменяемую коллекцию элементов любого типа данных. Для создания списка в Python используются квадратные скобки: my_list = [1, 2, 3].

Кортеж (Tuple) представляет собой упорядоченную неизменяемую коллекцию элементов любого типа данных (после создания кортежа, вы не можете изменить его элементы, добавить новые или удалить старые). Для создания кортежа в Python используются круглые скобки: my_tuple = (1, 2, 3).

Основное отличие между списком и кортежем заключается в их изменяемости. Список может быть изменен, кортеж - нет. Из-за этой разницы, выбор между ними зависит от конкретной задачи. Если вам нужна коллекция данных, которую вы можете изменять, используйте список. Если вам нужна коллекция данных, которую нельзя изменять после создания (например, для представления набора данных, который должен оставаться неизменным), то используйте кортеж.

Задание 3. Что такое миграции в Django и какие команды для работы с ними существуют в стандартном инструментарии Django?

SKILLFACTORY

Миграции в Django - это механизм, который позволяет автоматически обновлять схему базы данных на основе изменений в определении моделей Django. Это очень полезно при разработке веб-приложений, так как позволяет управлять структурой базы данных без необходимости ручного создания и обновления таблиц.

Основные команды:

migrate - применяет миграции и обновляет структуру базы данных на основе файлов миграций;

makemigrations - создает миграции на основе изменений в моделях Django, автоматически анализирует изменения в файлах моделей и создает файлы миграций, которые описывают, как нужно изменить схему базы данных, чтобы отразить эти изменения; й

sqlmigrate - генерирует SQL-запросы, которые будут выполнены при применении конкретной миграции;

showmigrations - выводит список всех миграций в проекте и показывает, были ли они уже применены к базе данных или нет (статус).

Задание 4. Есть два списка разной длины. В первом содержатся ключи, а во втором — значения. Напишите функцию make_dict, которая создаёт из этих ключей и значений словарь.

Если ключу не хватило значения, в словаре должно быть значение None. Значения, которым не хватило ключей, нужно игнорировать.

Обратите внимание! Функция должна возвращать созданный словарь.

Приложите код решения.

```
def make_dict(keys, values):
    result_dict = {}

    min_length = min(len(keys), len(values))

    for i in range(min_length):
        result_dict[keys[i]] = values[i]

    for i in range(min_length, len(keys)):
```

```
        result_dict[keys[i]] = None


    return result_dict


# Пример использования:
keys = ["a", "b", "c", "d"]
values = [4, 5, 6]
result = make_dict(keys, values)
print(result)
```

Задание 5. Напишите простой калькулятор. У вас должна быть функция calculate, которая принимает три аргумента: первые два — числа, третий — операция, которая должна быть произведена над ними.

Доступные операции: сложение, вычитание, умножение и деление.

Функция возвращает результат операции. В остальных случаях функция должна вернуть строку «Неизвестная операция».

```
def calculate(num1, num2, operation):
    if operation == 'сложение':
        return num1 + num2
    elif operation == 'вычитание':
        return num1 - num2
    elif operation == 'умножение':
        return num1 * num2
    elif operation == 'деление':
        if num2 != 0:
            return num1 / num2
        else:
```

```
        return 'Деление на ноль невозможно'
    else:
        return 'Неизвестная операция'


# Пример использования
num1 = float(input("Введите первое число: "))
num2 = float(input("Введите второе число: "))
operation = input("Введите операцию (сложение, вычитание, умножение, деление): ")

result = calculate(num1, num2, operation)
print(f"Результат операции: {result}")
```