**### Техническое задание (ТЗ) для функции add**

1. Название функции: add

2. Описание: Функция предназначена для добавления информации о товаре в список items.

3. Параметры функции:

- items: Словарь, в который добавляется информация о товаре.

- title: Название товара, которое является ключом в словаре items.

- good\_amount: Количество товара.

- expiration\_date (по умолчанию None): Дата истечения срока годности товара (если она есть).

4. Подробное описание шагов:

- Шаг 1: Проверка наличия названия товара в словаре items.

- Если название товара не существует как ключ в словаре items, то создать пустой список для этого товара.

- Шаг 2: Преобразование строки expiration\_date в объект date:

- Если expiration\_date предоставлено, преобразовать его в объект date с помощью datetime.strptime(), используя формат даты из переменной DATE\_FORMAT.

- Если expiration\_date не предоставлено (равно None), оставить expiration\_date без изменений.

- Шаг 3: Добавление информации о товаре в список items[title]:

- Добавить в список items[title] словарь, содержащий информацию о количестве товара good\_amount и дате истечения expiration\_date, если он был предоставлен.

5. Пояснения:

- Функция add предполагает, что переменная DATE\_FORMAT предварительно определена и содержит формат даты, используемый для преобразования строки в объект date.

- Предполагается, что словарь items уже существует и содержит информацию о товарах.

Функция add предназначена для добавления информации о товаре в список items. Она выполняет следующие шаги:

1. Проверяет, содержит ли список items название товара title. Если нет, то создает пустой список для этого товара.

2. Преобразует строковое представление даты expiration\_date в объект date с помощью datetime.strptime(), при условии, что expiration\_date предоставлен, иначе оставляет expiration\_date без изменений.

3. Добавляет в список items[title] словарь, содержащий информацию о количестве товара good\_amount и дате истечения expiration\_date, если она была предоставлена.

Теперь, когда у нас есть Техническое Задание и реализация функции add, мы можем продолжить с созданием Технического Задания для функции add\_by\_note.

**Техническое задание (ТЗ) для функции add\_by\_note**

1. Название функции: add\_by\_note

2. Описание: Функция должна принимать список items и строку note, которая содержит информацию о товарах. Функция должна обрабатывать каждую строку в note, извлекать информацию о товаре (название, количество, дата истечения) и добавлять эту информацию в список items.

3. Подробное описание шагов:

- Шаг 1: Разбиение строки note на отдельные строки по символу перевода строки (\n).

- Шаг 2: Итерация по каждой строке, полученной на шаге 1.

- Шаг 3: Разделение строки на отдельные части (название товара, количество, дата истечения, если она есть) по пробелам.

- Шаг 4: Проверка последней части строки:

- Если она представляет дату в формате "год-месяц-день", то она считается датой и сохраняется в переменную expiration\_date, предпоследняя часть строки считается количеством товара и сохраняется в переменную good\_amount, а все остальные части объединяются в название товара и сохраняются в переменную title.

- Если последняя часть не является датой, то она считается количеством товара и сохраняется в переменную good\_amount, все остальные части объединяются в название товара и сохраняются в переменную title, а переменная expiration\_date устанавливается в None.

- Шаг 5: Вызов функции add с извлеченными данными (название товара, количество, дата истечения) для добавления их в список items.

4. Пояснения:

- Предполагается, что функция add уже существует и добавляет новый товар в список items.

**\*\*Техническое задание (ТЗ) для функции поиска подстроки в списке элементов:\*\***

1. \*\*Цель:\*\*

- Создать функцию `find`, которая будет искать подстроку в элементах списка и возвращать список элементов, содержащих данную подстроку.

2. \*\*Входные данные:\*\*

- `items`: Список элементов, в котором будет производиться поиск подстроки.

- `needle`: Строка, подстроку которой требуется найти в элементах списка.

3. \*\*Действия:\*\*

\*\*Шаг 1: Инициализация списка результатов:\*\*

- Создать пустой список `result`, в котором будут храниться элементы списка, содержащие искомую подстроку.

\*\*Шаг 2: Поиск подстроки:\*\*

- Проход по каждому элементу `title` в списке `items`.

- Для каждого элемента `title` проверять, содержит ли он искомую подстроку `needle` (без учета регистра).

- Если подстрока присутствует в элементе списка, добавить данный элемент в список `result`.

\*\*Шаг 3: Возврат результата:\*\*

- Вернуть список `result`, содержащий элементы списка `items`, в которых найдена искомая подстрока.

4. \*\*Формат возврата:\*\*

- Функция `find` должна возвращать список элементов, содержащих подстроку `needle`.

**\*\*Техническое задание (ТЗ) для функции подсчета общей суммы продуктов с определенной подстрокой в названии:\*\***

1. \*\*Цель:\*\*

- Разработать функцию `amount`, которая будет находить продукты с заданной подстрокой в названии, суммировать их количество и возвращать общую сумму.

2. \*\*Входные данные:\*\*

- `items`: Словарь, содержащий названия продуктов в качестве ключей и информацию о каждом продукте в виде списка словарей.

- `needle`: Строка, подстрока, которую необходимо обнаружить в названиях продуктов.

3. \*\*Действия:\*\*

\*\*Шаг 1: Инициализация суммы продуктов:\*\*

- Создать переменную `product\_amount` типа Decimal со значением `0`, в которой будут накапливаться суммы продуктов.

\*\*Шаг 2: Поиск и суммирование продуктов:\*\*

- Итерация по каждому элементу `title` из результата функции `find(items, needle)`, которая возвращает список продуктов с подстрокой в названии.

- Для каждого найденного продукта, итерация по его частям в `items[title]`.

- Суммирование значений ключа `'amount'` из словарей частей продукта к переменной `product\_amount`.

\*\*Шаг 3: Возврат результата:\*\*

- Вернуть общую сумму продуктов с подстрокой `needle` в названии.

4. \*\*Формат возврата:\*\*

- Функция `amount` должна возвращать общую сумму продуктов типа Decimal.

**\*\*Техническое задание (ТЗ) для функции проверки истечения срока годности продуктов:\*\***

1. \*\*Цель:\*\*

- Разработать функцию `expire`, которая будет проверять продукты на истечение срока годности с учетом заданного числа дней вперед и возвращать список продуктов с остаточным количеством.

2. \*\*Входные данные:\*\*

- `items`: Словарь, содержащий названия продуктов в качестве ключей и информацию о каждом продукте в виде списка словарей.

- `in\_advance\_days` (необязательный): Количество дней, на которое нужно учитывать истечение срока годности (по умолчанию 0).

3. \*\*Действия:\*\*

\*\*Шаг 1: Инициализация данных и результата:\*\*

- Создать пустой список `result`, который будет содержать названия продуктов и остаточное количество.

- Получить текущую дату `today` с помощью `dt.date.today()`.

\*\*Шаг 2: Проверка срока годности продуктов:\*\*

- Итерация по каждой паре `title, parts` из элементов словаря `items`.

- Инициализация переменной `amount` типа Decimal со значением `0` для накопления остаточного количества.

- Для каждой части продукта в `parts`, проверить, истек ли срок годности с учетом `in\_advance\_days`:

- Если `expiration\_date` есть и срок годности истекает в или до `today + dt.timedelta(days=in\_advance\_days)`, увеличить `amount` на значение части продукта.

\*\*Шаг 3: Формирование результата:\*\*

- Если `amount` больше `0`, добавить кортеж `(title, amount)` в список `result`.

\*\*Шаг 4: Возврат результата:\*\*

- Вернуть список `result`, содержащий названия продуктов и остаточное количество, у которых истекает срок годности.

4. \*\*Формат возврата:\*\*

- Функция `expire` должна возвращать список кортежей (название продукта, остаточное количество) типа Decimal.

**В данной функции expire происходит следующее:**

1. Инициализируется пустой список result для хранения результатов.

2. Получается текущая дата с помощью dt.date.today() и сохраняется в переменной today.

3. Для каждой пары ключ-значение в словаре items происходит следующее:

- Инициализируется переменная amount как десятичное число 0.

- Для каждой части в parts:

- Если у части есть expiration\_date и если сегодняшняя дата плюс количество дней вперед больше или равно expiration\_date:

- Увеличивается значение amount на значение amount этой части.

- Если amount больше 0, то кортеж (title, amount) добавляется в список result с помощью функции list.append(result, (title, amount)).

4. Функция возвращает список result с результатами.

Эта функция сканирует элементы в словаре items, проверяет их даты и вычисляет сумму для частей, удовлетворяющих условиям. Результатом будет список кортежей (title, amount), представляющих состояние элементов, удовлетворяющих условиям.