ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ МОЛДОВЫ

ФАКУЛЬТЕТ МАТЕМАТИКИ И ИНФОРМАТИКИ

ДЕПАРТАМЕНТ ИНФОРМАТИКИ

Ciobanu Stanislav

Отчет

по дисциплине „ПРОГРАММИРОВАНИЕ В PYTHON”

Руководитель: _____ Плешка Наталья, лектор
(подпись)

Автор: _____
(подпись)

Кишинев, 2024

Содержание:

(1) Задача:

Вариант 1

Необходимо создать приложение для взаимодействия со словарём авторов. В этом словаре необходимо представить имя автора, как ключ, а книги, как значения по ключу.

Необходимо создать меню, в котором пользователь сможет вызывать функции для взаимодействия со словарём авторов.

Функции должны быть импортированы из отдельного файла.

Программа должна реагировать на неправильные действия пользователя.

(2) Описание алгоритмов:

Для представления меню используется чикл while, который будет выполняться до тех пор, пока значение переменной IsExecuting не станет 0. Внутри цикла каждый раз при помощи функции input считывается ввод пользователя и операторами if, elif, else выбирается необходимое действие.

```
authors = {}

IsExecuting = 1

while(IsExecuting) :

    print("\n1.Print list of authors")
    print("2.Count books for each author")
    print("3.Add an author")
    print("4.Add a book to author")
    print("5.Delete an author")
    print("6.Exit")

    inpt = input("\nEnter your choice: ")

    if inpt == '1':
        menu.PrintAuthorsData(authors)

    elif inpt == '2':
        menu.CountBooksForEachAuthor(authors)
```

```
elif inpt == '6':
    IsExecuting = 0
    print("Thank you for using my program!")

else :
    print("You've entered an not allowed option. Please try again")
```

Внутри цикла в зависимости от выбора пользователя будут вызываться функции.

(3) Взаимодействие с пользователем:

Тут же происходит начальная проверка дынных, введённых пользователем. Если пользователю необходимо ввести имя автора или название книги, а он вводит пустую строку, программа сообщит ему, что он ввёл неправильные данные.

Аналогично если пользователь выберет несуществующую опцию в меню, он получит об этом сообщение.

```
elif inpt == '4':
    print("Enter author please")
    authorName = input()

    print("Enter book please")
    book = input()

    if book != "" and authorName != "":
        menu.AddBook(authors, authorName, book)
    else:
        print("You've entered an incorrect book or author names")
```

В самих функциях так-же происходит проверка данных. Если пользователь ввёл несуществующего автора, то функция выведет на экран сообщение, что такого автора нет.

```
7   def AddBook(authorsDict, authorName, book):
8       if authorName in authorsDict:
9           authorsDict[authorName].append(book)
10          print("Book was successfully added")
11      else:
12          print("This author does not exist")
13
```

(4) Описание структур данных:

Данные хранятся в словаре. Ключ – это автор, значение – список книг. Изначально, задавая автора мы создаём ему пустой список. В последствии, чтобы добавить книгу, мы будем пользоваться методом append, который будет добавлять элемент в конец списка.

```
1 usage
def AddAuthor(authorsDict, authorName):
    if authorName not in authorsDict:
        authorsDict[authorName] = []
    else:
        print("This author does not exist")

1 usage
def AddBook(authorsDict, authorName, book):
    if authorName in authorsDict:
        authorsDict[authorName].append(book)
        print("Book was successfully added")
    else:
        print("This author does not exist")
```

(5) Описание функционала:

В приложении представлено 5 функций, которые позволяют взаимодействовать с авторами.

Функция AddAuthor принимает внутрь список авторов и имя автора. Она задаёт значение по ключу (имени автора) в виде пустого списка.

```
1 usage
def AddAuthor(authorsDict, authorName):
    if authorName not in authorsDict:
        authorsDict[authorName] = []
    else:
        print("This author does not exist")

```

Функция AddBook принимает внутрь список авторов, имя автора и название книги. Она добавляет название книги в конец списка книг по ключу.

```
1 usage
def AddBook(authorsDict, authorName, book):
    if authorName in authorsDict:
        authorsDict[authorName].append(book)
        print("Book was successfully added")
    else:
        print("This author does not exist")

```

Функция PrintAuthorsData принимает внутрь только список авторов и выводит на экран имя автора и все его книги. Если книг нет, она так и напишет, что книги не были найдены.

```
1 usage
def PrintAuthorsData(authorsDict):
    if authorsDict != {}:
        for author in authorsDict:
            if authorsDict[author] == []:
                print(f"{author} : No books found")
            else:
                books = ", ".join(authorsDict[author])  # Чтобы не было [] при выводе списка на экран
                print(f"{author} : {books}")
    else:
        print("No authors found")
```

Функция CountBooksForEachAuthor так-же принимает внутрь только список авторов и выводит на экран имя автора и количество элементов внутри списка его книг.

```
1 usage
def CountBooksForEachAuthor(authorsDict):
    if authorsDict != {}:
        for author in authorsDict:
            print(f"{author} : {len(authorsDict[author])}")
    else:
        print("No authors found")

```

Функция DeleteAuthor принимает внутрь список авторов и имя автора и по ключу удаляет элемент словаря.

```
1 usage
def DeleteAuthor(authorsDict, authorName):
    if authorName in authorsDict:
        del authorsDict[authorName]
        print("Author was successfully deleted")
    else:
        print("This author does not exist")
```

(6) Выводы:

Python обладает удобными инструментами для работы с различными массивами данных. В языках C/C++/C# и др. тот же функционал пришлось бы реализовывать при помощи гораздо более массивного и плохо-читаемого кода.

Однако в Python, языке со статической типизацией, тем не менее функции могут принимать любые типы данных, что требует постоянной проверки корректности передаваемых данных. В других языках для передачи двух разных типов в одну функцию необходима была бы перегрузка функции, в Python – не обязательно. Из-за этого могут возникать непредвиденные результаты.