Документация к программе, реализующей

функционал телефонной книги

Файлы:

* **Interface.py**

Главный модуль, реализующий общение с пользователем, большую часть интерфейса программы, из него запускается чтение и запись в файл Phone\_Book.txt, хранение текущего состояния телефонной книги

* **PhoneBookModule.py**

Здесь реализованы два класса: PhoneBook – объект этого класса представляет собой запись в телефонной книге; NumbersTypes – объект этого класса представляет собой поле номера телефона, имеющий расширенный функционал

* **PhoneBookEditor.py**

Здесь реализованы все функции для работы с телефонной книгой (часть из них относятся также к интерфейсу)

* **Phone\_Book.txt**

Исходный файл, из которого берется первичная информация и куда записывается обновленная (после корректного завершения работы программы – если программа не была прервана внешними обстоятельствами)

В программе используются модули из стандартной библиотеки Python:

* datetime
* os
* time
* re
* calendar

Рекомендуется запуск в командной строке Windows, так как для нее реализована очистка экрана после некоторого количества пройденных этапов работы с программой. Также используется декоративная функция для смены цвета текста в консоли, поэтому внешнее прерывание (например, с помощью сочетаний клавиш Ctrl + C) может привести к тому, что в консоли цвет текста останется отличным от ваших настроек.

Программа имеет только консольный интерфейс.

Обзор по каждому модулю

1.PhoneBookModule.py

Класс PhoneBook

Объект этого класса представляет собой запись телефонной книги.

Имеет поля:

* **name** – имя, string. Поле не может быть незаполненным, т.к. является часть уникального идентификатора
* **surname** – фамилия, string. Поле не может быть незаполненным, т.к. является часть уникального идентификатора
* **birthdate** – дата рождения, string Поле может быть незаполненным
* **phNumbers** – телефонные номера, объект класса NumberTypes. Поле не может быть незаполненным

Методы:

* **\_\_str\_\_(self)** – возвращает полностью отформатированную строку, состоящую из полей класса, удобную для чтения пользователем

Формированием строки из поля phNumbers занимается класс NumberTypes

* **countAge(self, inStr)** – подсчет и возвращение возраста, основанный на сегодняшней дате и дате рождения, указанной в записи  
  Параметр inStr используется для определения типа возвращаемого значения – True или False соответственно string или int  
  Если поле birthdate не заполнено, возвращается False или сообщение об ошибке в соответствии с параметром inStr  
  Заменяет поле, которое могло бы хранить в себе возраст, т.к. его в любом случае приходилось бы пересчитывать при запуске в новый день либо при изменении даты рождения

Класс NumberTypes

Объект этого класса реализует поле phNumbers для класса PhoneBook, поддерживает категоризацию номеров телефона.

Поля:

* mobiles – номера телефонов, соответствующие категории «мобильные», set
* workNums – номера телефонов, соответствующие категории «рабочие», set
* homeNums – номера телефонов, соответствующие категории «домашние», set

Хотя бы одно поле всегда должно иметь в себе хотя бы один номер телефона. Поля имеют тип «множество», чтобы избежать повторение одного и того же номера телефона в одной категории, но наличие экземпляров этого номера допускается в других категориях так же одноразово.

Методы:

* **\_\_str\_\_(self)** – возвращает полностью отформатированную строку, состоящую из названий категорий и содержащихся в них номеров (если в какой-либо категории отсутствуют номера, ее название не включается в строку) и удобную для чтения пользователем
* **writeByTypes(self, phoneNums, settings)** – производит распределение номеров телефонов по категориям в соответствии с переданными параметрами  
  phoneNums – список, состоящий из номеров, которые требуется записать в поля  
  settings – «настройки». Список идущих в порядке, соответствующем ожиданию категории, в которую будет записан номер из списка phoneNums (находящийся на той же позиции), строковых литералов ‘m’, ‘w’, ‘h’

Дополнительное замечание о переменных whiteTheme = “\033[37m” и lightBlueTheme = “\033[36m”.

Они встречаются во всех модулях как средства изменения цвета текста в консоли для декорирования выводимых данных.

2.Interface.py

Данный модуль реализует меню-интерфейс программы для пользователя, хранит в себе текущее состояние телефонной книги (в виде списка объектов класса PhoneBook) и управляет ее изменением (собственными средствами либо с помощью функций, реализованных в PhoneBookEditor).

Меню, встречающее пользователя:

--Welcome to the Top-Level Phone Book Manager--

1. View all records in current book

2. Add a new record to the book

3. Edit a record

4. Delete a record

5. Quick browse in records

6. Show the age of a person

7. Show the records with the nearest birthdays

8. Find by age

9. Quit

>>>

>>> - приглашение к любому вводу от пользователя

При выполнении любых действий с книгой программа выводит сообщение, которое просит пользователя следовать всем инструкциям, написанным ниже, а также информирует о том, что пользователь может в любое время прервать действие, если при любом запросе на ввод введет символ ‘\*’, при этом все изменения не будут сохранены.

Почти для любого из предложенных действий предоставлен поиск по записям в телефонный книге, который может вестись по любому сочетанию полей

1. View all records in current book

При вводе пользователем «1» программа выводит все существующие записи в текущей телефонной книге либо сообщение, что записей в книге пока нет

Используемые функции: нет

2. Add a new record to the book

Добавление новой записи в телефонную книгу. Предоставляет собой многоэтапный процесс, включающий в себя: ввод данных для всех полей (в т.ч. проверку на уникальный идентификатор имя+фамилия), возможность корректировки каждого из заполненных полей перед окончательным добавлением в книгу.

Если веденный уникальный идентификатор уже существует, пользователю предлагается либо повторить ввод имени-фамилии, либо выйти в главное меню, либо показать запись с совпадающим уникальным идентификатором.

Редактирование такой записи на месте не предоставляется, т.к. пользователь может уйти в бесконечный цикл совпадений имени+фамилии в уже существующих записях, что запутает его при создании новой записи.

Используемые функции: editNames, browseRecords, editDate, createNumbers, editsCommandCenter, parseToClassObj

3. Edit a record

Поиск нужной записи и ее редактирование, повторный запрос на подтверждение изменений, вывод сообщения в случае прерывания действия.

Используемые функции: printNChooseRes, editsCommandCenter, mainMenuYN

4. Delete a record

Поиск нужной записи и ее удаление, повторный запрос на подтверждение удаления, вывод сообщения в случае прерывания действия.

Используемые функции: printNChooseRes, mainMenuYN

5. Quick browse in records

Поиск нужной записи(ей) в телефонной книге с поддержкой совпадений по нескольким полям сразу, вывод найденных записей или сообщение об отсутствии совпадений

Используемые функции: whileInterface, mainMenuYN

6. Show the age of a person

Поиск интересующей записи для определения возраста, вывод сообщения о невозможности определения возраста для текущей записи, вывод сообщения о прерывании действия.

Используемые функции: printNChooseRes, mainMenuYN

7. Show the records with the nearest birthdays

Вывод записей, даты которых указывают на ближайшие дни рождения или вывод сообщения, что в ближайшие 30 дней нет дней рождений

Используемые функции: nearestBDs

8. Find by age

По введенному возрасту (проверка на его корректность) и критерию (старше, младше, ровно N лет) ищет подходящие записи и выводит их вместе с возрастом, если таких нет – сообщение об отсутствии нужных записей

Используемые функции: showAge

9. Quit

Выход из программы. Завершает работу программы записью данных в файл, очищает консоль

При любом некорректном вводе команды от пользователя выдается сообщение об ошибке и предлагается повторный ввод

3.PhoneBookEditor.py

Модуль со всеми функциями, участвующими во внесении изменений к телефонную книгу (в т.ч. и функции, реализующие интерфейс)

Примечание: любой параметр с именем message/question нужен только для декоративных целей

parseToClassObj(toParse, index, currentPhoneBook)-> index (int)

Превращает строковые данные toParse в объект класса PhoneBook и добавляет его в список текущей телефонной книги currentPhoneBook. Параметр index для определения последней занятой ячейки в списке currentPhoneBook.

Возвращает новый index (последняя занятая ячейка в currentPhoneBook)

Используемые функции: нет

writeToFile(currentPhoneBook,file) -> нет возвращаемого значения

Обратная функция – для записи сформированных из списка currentPhoneBook данных в исходный файл, указанный параметром file.

Используемые функции: нет

editNames(message) -> newName (str)

Функция, реализующая проверку корректности введенного имени/фамилии для создания, редактирования и поиска по данному значению. Допускает наличие в имени любых букв и цифр (не в начале) и пробелов, автоматическое выравнивание регистра.

Возвращает корректную форму имени

Используемые функции: нет

editDate(question) -> newBirthdate (str)

Функция, реализующая проверку корректности введенной даты для создания, редактирования и поиска по данному значению. Допускает только числовой формат даты вида DD.MM.YYYY.

Возвращает корректную форму даты

Используемые функции: checkData

checkData(date) -> (boolean)

Проверяет корректность формата даты в особом режиме, чтобы избежать ошибок в программе при неправильных данных

Используемые функции: нет

createNumbers(edit) -> newNumbers, settings (list, list)

newNumbers[0] (string)

Функция, реализующая проверку корректности введенного номера телефона для создания, редактирования и поиска по данному значению. Допускает только полностью цифровой формат номера, авто замена «+7» на «8», всего 11 цифр.

Параметр edit служит командой для выполнения только корректировки одного введенного номера, если он = false, то функция позволяет создать списки номеров и категорий, к которым они относятся, так же функция поддерживает немедленное прерывание действия.

Возвращает: списки с номерами и категориями/только строку с единственным номером/пустые списки, если действие было прервано

Используемые функции: нет

editClassObjNums(nums) -> newNumCatalogue (NumberTypes)

(boolean)

Функция, реализующая изменение объекта класса NumberTypes. Поддерживает редактирование, удаление и добавление в него номеров. Не позволяет оставить все поля объекта пустыми. Выводит текущее состояние объекта после каждого изменения. Поддерживает прерывание действия.

Возвращает новый объект класса с измененными данными или false в случае прерывания действия

Используемые функции: chooseNumber

chooseNumber(i, message, mlen, wlen, hlen) -> index (int), (str)

(str)

Высчитывает нужный индекс и нужную категорию по выбранному пользователем номеру телефона из предложенных в списке вывода в функции editClassObjNums. Параметр i – выбранный пользователем индекс, mlen, wlen, hlen – количество номеров в каждой категории.

Эти же данный и возвращает, в случае прерывания действия – символ ‘\*’.

Используемые функции: нет

editsCommandCenter(name, surname, birthdate, phNums)

-> name(str), surname(str), birthdate(str), newNums(NumberTypes)

(boolean) (boolean) (boolean) (boolean)

Функция, реализующая элемент интерфейса, служащий для управления редактированием любого поля записи. Поддерживает прерывание действия.

Возвращает измененные параметры + новый объект для номеров телефона, чтобы по ошибке пользователя не изменить предыдущий, если это не нужно. При принудительном прерывании возвращает ложные параметры

Используемые функции: editNames, editDate, editClassObjNums

browseRecords(toSearch, field, currentPhoneBook, indexesForList, toSearch=””) -> indexes (list)

i (int) / (int)

Функция, реализующая поиск по записям в телефонной книге по заданным параметрам. toSearch – что искать, field – в каком поле искать, currentPhoneBook – текущая телефонная книга, indexesForList – параметр для реализации поиска по нескольким полям (позволяет проводить поиск по определенным записям в книге), toSearch2 – особый параметр для поиска уникального идентификатора.

Возвращает список индексов, по которым были найдены записи с совпадениями / индекс, где произошло совпадение / -1 как особое значение совпадений не найдено

Используемые функции: нет

whileInterface(currentPhoneBook, lenOfBook) -> indexes (list)

(string)

Функция, реализующая элемент интерфейса, предоставляющий возможности настройки поиска. Запись можно вести по любому сочетанию из 4х полей телефонной книги, но каждое участвует не более одного раза (максимум: одно имя, одна фамилия, одна дата, один номер). По фрагментам возможно искать только имя и фамилию, дата и номер требуются полные.

Возвращает список индексов, по которым содержатся записи, совпавшие со всеми критериями поиска / символ ‘\*’ при прерывании действия.

Используемые функции: editNames, editDate, createNumbers, browseRecords, mainMenuYN

printNChoose(currentPhoneBook, message, lenOfBook) -> indexes[i] (int)

(string)

Функция, реализующая элемент интерфейса, предлагающий пользователю выбрать из найденных результатов запись, с которой он будет дальше работать. Запрос на повторное подтверждение выбора, поддержка прерывания действия.

Возвращает индекс записи в списке или символ ‘\*’.

Используемые функции: whileInterface, mainMenuYN

mainMenuYN(message, message2) -> (boolean)

Функция, реализующая элемент интерфейса, в котором пользователю задается вопрос о выходе в меню по разным причинам.

Возвращает согласие / несогласие пользователя

Используемые функции: нет

nearestBDs(currentPhoneBook) -> nearBDs(list)

По текущей дате и полю birthdate в записях текущей телефонной книги currentPhoneBook определяет дни рождения в районе 30 дней (считая от текущего дня, т.е. если сегодня 1 декабря, до ближайшие дни рождения ищутся в промежутке между 1м и 30м декабря включительно).

Возвращает список записей из книги

Используемые функции: нет

showAge(age, statement, currentPhoneBook) -> (list), (string)

По заданным параметрам ищет записи с подходящим возрастом.

Возвращает список подходящих записей и сообщение.

Используемые функции: нет