МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)

Факультет: Энергетический

Кафедра:Информатики, вычислительной техникии прикладнойматематики

КУРСОВАЯ РАБОТА

По дисциплине: «Программирование»

На тему: БД подписчиков журнала

Отлично

Выполнил студент группы
ИВТз-20 Жерлов Григорий Владимирович

Руководитель работы:
Старший преподаватель кафедры информатики вычислительной техники и прикладной математики (ИВТиПМ) Ветров Сергей Владимирович

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)

Факультет: Энергетический

Кафедра: Информатики, вычислительной техники и прикладной математики

ЗАДАНИЕ

на курсовую работу

по курсу “Программирование”

Студенту: Жерлову Григорию Владимировичу

1.Тема работы “БД подписчиков журнала”

2.Исходные данные, используемые в программе: Создать программу – базу данных подписчиков журнала.

3. Данные о подписчиках, используемые в программе: Имя, Фамилия, возраст, название журнала, сумма оплаты, срок подписки.

Дата выдачи задания 02.02.2021

Руководитель: Ветров Сергей Владимирович

Дата представления студентом законченной работы

Задание принял к исполнению 02.02.2021

Подпись студента _____

Содержание

ВВЕДЕНИЕ

База данных, должна содержать сведения о подписчиках журнала: имя, фамилия, возраст, название журнала, сумма оплаты и срок подписки. Реализовать возможности: создание нового/случайного подписчика и его сохранение в файл, загрузка списка подписчиков из файла, вывод на экран всех данных о подписчиках, выборочное удаление подписчиков по присвоенному номеру, редактирование данных о выбранном подписчике, сохранение редактированной базы данных в файл, поиск по базе данных, сортировка всех подписчиков по названию журнала.

Программа в виде консольного приложения будет состоять из модулей, содержащих функции для работы с базой данных, класса, описывающего информацию о подписчике, текстовый файл, где хранятся данные, функции, которая отвечает за интерфейс программы в консольном окне.

На создание базы данных потребуется около 4-5 недель для изучения материала и создания основных функций.

Язык программирования Python

Python – высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объем полезных функций.

Python поддерживает несколько парадигм программирования: структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное. Основные архитектурные черты – динамическая типизация, автоматическое управление памятью, полная интроспекция,

механизм обработки исключений, поддержка многопоточных вычислений и удобные высокоуровневые структуры данных. Код в Python организовывается в функции и классы, которые могут объединяться в модули.

Концепция программы

Программа выполнена в виде консольного приложения без графического интерфейса, т.к. она нужна для отработки практических навыков студента и показывает его уровень владения данным языком программирования.

Во время разработки были использованы принципы модульного программирования. Когда программа запускается, перед пользователем появляется основное меню программы в виде 10 пунктов, которые описывают весь функционал программы:

1. Добавить нового случайного подписчика
2. Добавить нового подписчика вручную
3. Загрузить список подписчиков из файла
4. Показать список подписчиков
5. Удалить подписчика из списка
6. Редактировать
7. Сохранить данные в базу
8. Поиск
9. Сортировка
0. Выход

При выборе с 1 по 9 пункт, программа выполняет необходимую функцию и возвращает пользователя к меню выбора до тех пор, пока он не завершит программу с помощью пункта 0. Выход. Выбрав первый или второй пункт, пользователь имеет возможность выбора: сохранить новые данные (в файл) или нет.

Структура программы

Для решения задачи было решено начать с создания интерактивного меню в виде отдельной функции main(), которую мы в итоге и вызываем. Функция -- это цикл, который не завершится, пока пользователь не выберет “Выход”. В цикле используется каскадная условная конструкция, которая позволяет совершать выбор и взаимодействовать с программой. Каждое условие вызывает функцию или ряд функций, описанных в собственных модулях Front и Back, которые разделены по аналогии с Frontend и Backend, где функции, которые помогают взаимодействовать с пользователем и обрабатывают данные, а сам процесс не виден, соответственно.

Таблица 1 – Функции, использованные в программе

Функция	**Описание**
`print_sub()`	Выводит данные о подписчике
`print_sub_list()`	Выводит все данные о подписчиках на экран
`Choose()`	Позволяет выбрать пользователью: сохранить созданного подписчика или нет
`Search()`	Выполняет поиск среди данных о подписчиках по ключевому слову
`print_heading()`	Выводит шапку таблицы о подписчках
`add_rand_sub()`	Создает рандомного подписчика
`add_new_sub()`	Создает нового подписчика
`sub2file(sub, filename)`	Сохраняет данные об одном подписчике в указаныний файл
`load_database(filename)`	Загружает данные из файла
`subss2file( subs, filename )`	Сохраняет данные о подписчках в указаныний файл
`sort_subs(database)`	Сортирует данные по названию журнала

В дальнейшем в основную часть программы была добавлена функция создания нового и загрузки старых файлов для работы с базой данных loading().

```
def loading():
    while True:
        print("""
        1.Новый
        2.Загрузить
        """)
        ans = input().strip()
        if ans in ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']:
            ans = int(ans)
        else:
            continue
        if ans == 2:
            saves = []
            f = open("save.txt","r")
            while True:
                s = f.readline()
                if s == '':
                    break
                saves = saves+[s]
            for i in range(len(saves)):
                print(i+1)
                print(saves[i])
            number_of_file = int(input())
            name_of_file = saves[number_of_file-1]
            break
        elif ans == 1:
            new_file = input()+".txt"
            f = open("save.txt","a")
            f.write(new_file)
            f.write("\n")
            name_of_file = new_file
            break

    return name_of_file.strip()
file_of_names = loading()
```

Функция при выборе “Загрузить”, загружает файл, в котором содержатся имена ранее созданных файлов, выводит их на экран, нумеруя, и дает пользователю выбрать с каким файлом дальше работать. Когда при выборе “Новый” пользователь создает новый файл и в дальнейшем работает с ним.

Модуль Back

Содержит в себе описание класса Subscriber. Переменные name, surname, age, magazine, payment, subscription являются переменными Subscriber, значение которых являются общими для всех экземпляров этого класса. Метод __init__ - конструктор класса, который Pyton вызывает при создании нового экземпляра класса. Получаем доступ к атрибутам объекта, используя оператор точки с объектом.

```
1) def add_rand_sub():
    """Создает рандомного подписчика"""
    sub = []
    Names = ["Hermione", "Harry", "Ron", "Draco", "Albus", "Nevil", "Lucius",
"Sedric", "Goil", "Tom", "Grum", "Hagrid"]
    Surnames = ["Granger", "Potter", "Uisley", "Malfoi", "Dambldor",
"Longbotto1m", "Digori", "Nosurname", "Reddle",
                "Forester"]
    Magazines = ["Gryffindor", "Hufflepuff", "Ravenclaw", "Slytherin"]

    name = Names[randint(0, 11)]
    surname = Surnames[randint(0, 9)]
    magazine = Magazines[randint(0, 3)]

    sub = sub + [Subscriber(name, surname, randint(12, 100), magazine,
randint(0, 100), randint(1, 12))]
    return sub
```

Функция создает и возвращает нового случайного подписчика в виде списка атрибутов, созданного на основе простого генератора с помощью randint(), вызванного из подключенного модуля random.

```
2) def add_new_sub():
    """Создает нового подписчика"""
    sub = []

    name = input("Введите Имя - ")
    surname = input("Введите Фамилию - ")
    age = input("Введите возраст - ")
    magazine = input("Введите название журанала - ")
    payment = input("Введите сумму оплаты - ")
    subscription = input("Введите срок подписки (от 1 до 12 месяцев) - ")

    sub = sub + [Subscriber(name, surname, age, magazine, payment,
subscription)]
    return sub
```

Функция делает то же, что и первая, но не из случайных значений, а просит пользователя самостоятельно заполнить все данные о новом подписчике.

```
3) def sub2file(sub, filename):
    """Сохраняет данные об одном подписчике в указаныйй файл"""
```

```
    f = open(filename, "a")

    for d in sub:
        f.write(d.name)
        f.write(", ")
        f.write(d.surname)
        f.write(", ")
        f.write(str(d.age))
        f.write(", ")
        f.write(d.magazine)
        f.write(", ")
        f.write(str(d.payment))
        f.write(", ")
        f.write(str(d.subscription))
        f.write("\n")

    f.close()
```

Функция открывает или создает файл (если он еще не создан) на редактирование с сохранением, и, проходя по каждому атрибуту экземпляра класса, записывает его, разделяя запятой с пробелом в одну строчку. Данная функция вызывается, когда перед пользователем встает выбор, сохранить ли нового подписчика в файл.

```
4) def load_database(filename):
    """загружает данные из файла"""
    database = []
    f = open(filename, 'r')
    while True:
        s = f.readline()
        if s == '':
            break
        s = s.split(',')
        d = Subscriber(s[0].strip(),
(s[1].strip()),(s[2].strip()),(s[3].strip()),(s[4].strip()),(s[5].strip()))
        database = database + [d]

    f.close()
    return database
```

Функция открывает файл на чтение и читает каждую строку файла до тех пор, пока они не закончатся. В каждом периоде цикла считывается строка, разбивается на части помощью разделителя .split, и каждая часть записывается в список атрибутов объекта. После чего объект с атрибутами заносится в единый список database.

```
5) def subss2file( subs, filename ):
    """Сохраняет данные о подписчках в указаныйй файл"""
    f = open( filename, "w" )

    for d in subs:
        f.write(d.name)
        f.write(", ")
        f.write(d.surname)
        f.write(", ")
        f.write(str(d.age))
        f.write(", ")
        f.write(d.magazine)
        f.write(", ")
        f.write(str(d.payment))
        f.write(", ")
        f.write(str(d.subscription))
        f.write("\n")

    f.close()
```

Функция открывает или создает файл (если он еще не создан) на перезапись, и, проходя по каждому атрибуту экземпляра класса, записывает его, разделяя запятой с пробелом в одну строчку.

Данная функция вызывается, когда пользователь захочет сохранить отредактированную базу данных в файл.

```
6) def sort_subs(database):
    """Сортирует данные по названию журнала"""
    a = True
    while a:
        a = False
        for i in range(len(database) - 1):
            if database[i + 1].magazine < database[i].magazine:
                database[i + 1], database[i] = database[i], database[i + 1]
                a = True
```

Функция позволяет сортировать все данные о пользователях по названию журнала. Для сортировки был использован пузырьковый метод, когда сравнивается текущий элемент со следующим, когда следующий элемент меньше текущего, то они меняются местами и до тех пор, пока данные не будут отсортированы. В данном случае объекты меняются местами, если атрибут magazine будет меньше следующего.

Модуль Front

```
1) def print_sub(d):
    """Выводит данные о подписчике"""
    print("{:^12}".format(d.name), end=(' | '))
    print("{:^12}".format(d.surname), end=(' | '))
    print("{:^12}".format(d.age), end=(' | '))
    print("{:^12}".format(d.magazine), end=(' | '))
    print("{:^12}".format(d.payment), end=(' | '))
    print("{:^12}".format(d.subscription))
```

Функция выводит данные, когда создается новый (случайный) подписчик на экран.

```
2) def print_sub_list( subs ):
    """Выводит все данные о подписчиках на экран"""
    count = 1
    for d in subs:
        print("{:^6}".format(count), end=(' | '))
        print_sub( d )
        count += 1
```

Функция проходит по списку объектов и выводит их на экран в виде таблицы, где каждому подписчику присваивается его порядковый номер.

```
3) def choose(sub,file):
    """Позволяет выбрать пользователю: сохранить созданного подписчика или
нет"""
    while True:
        print('''Желаете ли внести данного попдписчика в базу данных?
        1. Да
        2. Нет
        ''')
        ans1 = int(input())
        if ans1 == 1:
            print('Подписчик добавлен')
            sub2file(sub, file)
            break
        elif ans1 == 2:
            break
```

Функция выбора из двух вариантов ответа, когда создается новый подписчик. Пользователь может при ответе “Да” внести нового подписчика в базу данных, используя sub2file(), а может и не вносить.

```
4) def search(search_word, base):
    """Выполняет поиск среди данных о подписчиках по ключевому слову"""
    cnt = 0
    count = 1
    for sub in base:
        cnt += 1
        if (sub.name.find(search_word) != -1) or (sub.surname.find(search_word)
!= -1) or (
                sub.magazine.find(search_word) != -1):
            print("{:^6}".format(count), end=(' | '))
```

```
            print_sub(base[cnt - 1])
        count += 1
```

Функция проходится по базе данных и ищет ключевое слово, которое ввел пользователь, среди имен, фамилий и названий журнала. Если есть совпадения, то выводятся отдельной таблицей данные о подписчиках с нужной информацией.

```
5) def print_heading():
    """Выводит шапку таблицы о подписчках"""
    print("{:^6}".format("Number"), end=(' | '))
    print("{:^12}".format("Name"), end=(' | '))
    print("{:^12}".format("Surname"), end=(' | '))
    print("{:^12}".format("Age"), end=(' | '))
    print("{:^12}".format("Magazine"), end=(' | '))
    print("{:^12}".format("Payment"), end=(' | '))
    print("{:^12}".format("Subscription"))
```

Функция, которая выводится, когда нужно вывести список (одного) подписчиков на экран. Нужна для оформления и читабельности программы.

Заключение

Создана программа на языке Python, которая позволяет создать и редактировать базу данных подписчиков журнала. Реализованы: создание/загрузка файловой системы, работа с собственными модулями и функциями в программе, можно создать нового/случайного подписчика. Сохранение данных в файл, загрузка списка подписчиков из файла, вывод на экран всех данных о подписчиках, выборочное удаление подписчиков по присвоенному номеру, редактирование данных о выбранном подписчике, сохранение редактированной базы данных в файл, поиск по базе данных, сортировка всех подписчиков по названию журнала.

Программа в виде консольного приложения состоит из модулей, содержащих функции для работы с базой данных, класса, описывающего информацию о подписчике, текстовый файл, где хранятся данные, функции, которая отвечает за интерфейс программы в консольном окне.

На создание базы данных потребовалось около 3 недель. При первой попытке была неудачная реализация базы данных исключительно через списки, из-за чего работа вошла в процесс стагнации. Далее, изучив новый материал и понятие классов, программа была написана в короткие строки. Главную трудность вызывала работа с атрибутами класса из-за непонимания.

Программу можно усовершенствовать, добавив ей графический интерфейс, обработав все исключения, параллельно тестируя код для того, чтобы база данных выглядела, как полноценное приложение.

Список использованных источников

Печатные издания:

1. Лутц М. Программирование на Python. 2011. - 992 с.

2. Лутц М. Изучаем Python. 2010. - 1280 с.

3. Николай Прохоренок, Владимир Дронов. Python 3 и PyQt 5. Разработка приложений. –СПб.: БХВ-Петербург, 2017. – 832с.: ил.

1. github.com/VetrovSV/Programming — Репозитарий с материалами по языку Python.

2. python.org - официальная документация Python

Приложение

Основной код

```
from random import *
from Front import *
from Back import *


DataBase = []

def loading():
    while True:
        print("""
        1.Новый
        2.Загрузить
        """)
        ans = input().strip()
        if ans in ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']:
            ans = int(ans)
        else:
            continue
        if ans == 2:
            saves = []
            f = open("save.txt","r")
            while True:
                s = f.readline()
                if s == '':
                    break
                saves = saves+[s]
            for i in range(len(saves)):
                print(i+1)
                print(saves[i])
            number of file = int(input())
            name_of_file = saves[number_of_file-1]
            break
        elif ans == 1:
            new_file = input()+".txt"
            f = open("save.txt","a")
            f.write(new_file)
            f.write("\n")
            name_of_file = new_file
            break

    return name_of_file.strip()


def main():
    while True:
        print("""Введите цифру для
    1. Добавить нового случайного подписчика
    2. Добавить нового подписчка вручную
    3. Загрузить список подписчиков из файла
    4. Показать список подписчиков
    5. Удалить подписчика из списка
    6. Редактировать
    7. Сохранить данные в базу
    8. Поиск
    9. Сортировка

    0. -- выход
            """)
        ans = input().strip()
```

```
        if ans in ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']:
            ans = int(ans)
        else:
            continue
        if ans == 1:
            l = add_rand_sub()
            print_heading()
            print_sub_list( l )
            choose(l,file_of_names)
        elif ans == 2:
            l = add_new_sub()
            print_heading()
            print_sub_list(l)
            choose(l,file_of_names)

        elif ans == 3:
            print(file_of_names)
            DataBase = load_database(file_of_names)
            print("Список подписчиков загружен")
        elif ans == 4:
            print("Список подписчиков:")
            print_heading()
            print_sub_list(DataBase)
        elif ans == 5:
            print('Введите номер подписчика для его удаления')
            numb = int(input())
            DataBase.remove(DataBase[numb-1])
            print_heading()
            print_sub_list(DataBase)
        elif ans == 6:
            print('Данные какого подписчика Вы хотите изменить?')
            numb = int(input())
            l = add_new_sub()
            DataBase[numb-1] = l[0]
            print_heading()
            print_sub_list(DataBase)
        elif ans == 7:
            subss2file(DataBase, file_of_names)
        elif ans == 8:
            search_name = input('Введите имя/словосочетание для поиска - ')
            print_heading()
            search(search_name, DataBase)
        elif ans == 9:
            print("Сортированный список подписчиков")
            print_heading()
            sort_subs(DataBase)
            print_sub_list(DataBase)
        elif ans == 0:
            break

file_of_names = loading()
main()
```

Модуль Back

```
from random import *


class Subscriber:
    """Информация о подписчике"""

    name = ''
    surname = ''
    age = 0
    magazine = ''
    payment = 0
    subscription = 0

    def __init__(self, name1, surname1, age1, magazine1, payment1,
subscription1):
        self.name = name1
        self.surname = surname1
        self.age = age1
        self.magazine = magazine1
        self.payment = payment1
        self.subscription = subscription1


def add_rand_sub():
    """Создает рандомного подписчика"""
    sub = []
    Names = ["Hermione", "Harry", "Ron", "Draco", "Albus", "Nevil", "Lucius",
"Sedric", "Goil", "Tom", "Grum", "Hagrid"]
    Surnames = ["Granger", "Potter", "Uisley", "Malfoi", "Dambldor",
"Longbotto1m", "Digori", "Nosurname", "Reddle",
                "Forester"]
    Magazines = ["Gryffindor", "Hufflepuff", "Ravenclaw", "Slytherin"]

    name = Names[randint(0, 11)]
    surname = Surnames[randint(0, 9)]
    magazine = Magazines[randint(0, 3)]

    sub = sub + [Subscriber(name, surname, randint(12, 100), magazine,
randint(0, 100), randint(1, 12))]
    return sub


def add_new_sub():
    """Создает нового подписчика"""
    sub = []

    name = input("Введите Имя - ")
    surname = input("Введите Фамилию - ")
    age = input("Введите возраст - ")
    magazine = input("Введите название журанала - ")
    payment = input("Введите сумму оплаты - ")
    subscription = input("Введите срок подписки (от 1 до 12 месяцев) - ")

    sub = sub + [Subscriber(name, surname, age, magazine, payment,
subscription)]
    return sub


def sub2file(sub, filename):
```

```
    """Сохраняет данные об одном подписчике в указаныий файл"""
    f = open(filename, "a")

    for d in sub:
        f.write(d.name)
        f.write(", ")
        f.write(d.surname)
        f.write(", ")
        f.write(str(d.age))
        f.write(", ")
        f.write(d.magazine)
        f.write(", ")
        f.write(str(d.payment))
        f.write(", ")
        f.write(str(d.subscription))
        f.write("\n")

    f.close()


def load_database(filename):
    """загружает данные из файла"""
    database = []
    f = open(filename, 'r')
    while True:
        s = f.readline()
        if s == '':
            break
        s = s.split(',')
        d = Subscriber(s[0].strip(),
(s[1].strip()),(s[2].strip()),(s[3].strip()),(s[4].strip()),(s[5].strip()))
        database = database + [d]

    f.close()
    return database


def subss2file( subs, filename ):
    """Сохраняет данные о подписчках в указаныий файл"""
    f = open( filename, "w" )

    for d in subs:
        f.write(d.name)
        f.write(", ")
        f.write(d.surname)
        f.write(", ")
        f.write(str(d.age))
        f.write(", ")
        f.write(d.magazine)
        f.write(", ")
        f.write(str(d.payment))
        f.write(", ")
        f.write(str(d.subscription))
        f.write("\n")

    f.close()


def sort_subs(database):
    """Сортирует данные по названию журнала"""
    a = True
    while a:
        a = False
```

```
        for i in range(len(database) - 1):
            if database[i + 1].magazine < database[i].magazine:
                database[i + 1], database[i] = database[i], database[i + 1]
```

Модуль Front

```
from Back import *

def print_sub(d):
    """Выводит данные о подписчике"""
    print("{:^12}".format(d.name), end=(' | '))
    print("{:^12}".format(d.surname), end=(' | '))
    print("{:^12}".format(d.age), end=(' | '))
    print("{:^12}".format(d.magazine), end=(' | '))
    print("{:^12}".format(d.payment), end=(' | '))
    print("{:^12}".format(d.subscription))


def print_sub_list( subs ):
    """Выводит все данные о подписчиках на экран"""
    count = 1
    for d in subs:
        print("{:^6}".format(count), end=(' | '))
        print_sub( d )
        count += 1


def choose(sub,file):
    """Позволяет выбрать пользователю: сохранить созданного подписчика или
нет"""
    while True:
        print('''Желаете ли внести данного попдписчика в базу данных?
        1. Да
        2. Нет
        ''')
        ans1 = int(input())
        if ans1 == 1:
            print('Подписчик добавлен')
            sub2file(sub, file)
            break
        elif ans1 == 2:
            break


def search(search_word, base):
    """Выполняет поиск среди данных о подписчиках по ключевому слову"""
    cnt = 0
    count = 1
    for sub in base:
        cnt += 1
        if (sub.name.find(search_word) != -1) or (sub.surname.find(search_word)
!= -1) or (
                sub.magazine.find(search_word) != -1):
            print("{:^6}".format(count), end=(' | '))
            print_sub(base[cnt - 1])
        count += 1


def print_heading():
    """Выводит шапку таблицы о подписчках"""
    print("{:^6}".format("Number"), end=(' | '))
    print("{:^12}".format("Name"), end=(' | '))
    print("{:^12}".format("Surname"), end=(' | '))
```

```
print("{:^12}".format("Age"), end=(' | '))
print("{:^12}".format("Magazine"), end=(' | '))
print("{:^12}".format("Payment"), end=(' | '))
print("{:^12}".format("Subscription"))
```