Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Воронежский государственный лесотехнический университет

имени Г.Ф. Морозова»

Кафедра вычислительной техники и информационных систем

**Пояснительная записка**

курсовой работы

на тему «Разработка информационной системы в интегрированной среде программирования»

Вариант 1

09.03.02 Информационные системы и технологии

По дисциплине «Высокоуровневые методы программирования»

|  |  |  |
| --- | --- | --- |
| Студент группы \_ИС2-191-OБ\_\_\_  (номер группы)  Руководитель, \_к.т.н.\_\_\_\_\_\_\_\_\_\_\_  (ученая степень, ученое звание) | \_\_\_\_\_\_\_\_\_\_\_  (подпись)  \_\_\_\_\_\_\_\_\_\_  (подпись) | Величко В.А.  (инициалы и фамилия)  \_О.В. Оксюта\_\_  (инициалы и фамилия) |

Воронеж 2022

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Воронежский государственный лесотехнический университет

имени Г.Ф. Морозова»

Кафедра вычислительной техники и информационных систем

**ЗАДАНИЕ**

для курсовой работы

Студенту Величко Владиславу Андреевичу группы ИС2-191-OБ

(фамилия и полные инициалы)

Воронежского государственного лесотехнического университета

имени Г.Ф. Морозова

Направление 09.03.02 Информационные системы и технологии

Срок представления работы к защите «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Исходные данные для проектирования: для своей предметной области выполнить проектирование программного продукта для работы с базами данных.

Перечень вопросов, подлежащих разработке: цель курсового проектирования, краткие сведения по теме, обзор литературных источников; описать функции для получения значений и модификации элементов-данных.

Руководитель, к.т.н \_\_\_\_\_\_\_\_\_\_\_\_\_ О.В. Оксюта

(ученая степень, ученое звание) (подпись) (инициалы и фамилия)

Задание принял студент \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Величко В.А.

(подпись) (число, месяц, год) (инициалы и фамилия)

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 4

1. Язык Python и среда разработки 5

1.1 Общее описание языка 5

1.3 Описание среды разработки PyCharm 9

2. Выполнение практических заданий 10

2.1 Парсинг 10

2.2 Парсер ВК 11

ЗАКЛЮЧЕНИЕ 13

СПИСОК ЛИТЕРАТУРЫ 14

ПРИЛОЖЕНИЕ 15

# ВВЕДЕНИЕ

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью объектно-ориентированным в том плане, что всё является объектами. Необычной особенностью языка является выделение блоков кода пробельными отступами. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Сам же язык известен как интерпретируемый и используется в том числе для написания скриптов. Недостатками языка являются зачастую более низкая скорость работы и более высокое потребление памяти написанных на нём программ по сравнению с аналогичным кодом, написанным на компилируемых языках, таких как C или C++.

Python стал одним из самых популярных языков, он используется в анализе данных, машинном обучении, DevOps и веб-разработке, а также в других сферах, включая разработку игр. За счёт читабельности, простого синтаксиса и отсутствия необходимости в компиляции язык хорошо подходит для обучения программированию, позволяя концентрироваться на изучении алгоритмов, концептов и парадигм. Отладка же и экспериментирование в значительной степени облегчаются тем фактом, что язык является интерпретируемым. Применяется язык многими крупными компаниями, такими как Google или Facebook. По состоянию на октябрь 2021 года Python занимает первое место в рейтинге TIOBE популярности языков программирования с показателем 11,27%. «Языком года» по версии TIOBE Python объявлялся в 2007, 2010, 2018 и 2020 годах.

# Язык Python и среда разработки

1.1 Общее описание языка

* Высокоуровневый

Языки программирования делятся на высокоуровневые и низкоуровневые. Низкоуровневые языки — языки, близкие к машинному коду или его конструкциям (например, байт-кодам). Классикой таких языков являются C, Assembler, Forth. Высокоуровневые языки — соответственно разрабатываются для у добства использования и скорости написания программы. В них применяются определённые абстракции — структуры данных, набор вспомогательных функций и так далее. Это такие языки как Python, JS, PHP, Go.

### Интерпретируемый

Языки делятся на интерпретируемые(Python, JS, PHP, R, Ruby) и компилируемые (С, С++, Pascal). В первом случае программа выполняется специальной программой —  интерпретатором, во втором программа сначала преобразуется в понятные компьютеру исполняемые файлы.

### Объектно-ориентированный

Все языки также разделяются  на процедурные, функциональные и объектно-ориентированные в зависимости от того, с помощью каких конструкций создаётся программа и как происходит её выполнение.

В объектно-ориентированных языках основа это классы и экземпляры классов это равносильно типу и объекту этого типа. Выполнение условных задач или же просто работа программы строится на взаимодействии различных классов.

Python хоть и является объектно-ориентированным языком но также поддерживает и процедурное программирование это значит, что программу можно написать без единого класса.

В основе функциональных языков лежит отличная от предыдущих вычислительная система, называемая лямбда-исчисление, которая, тем не менее, эквивалентна машине Тьюринга, о чём доказана соответствующая теорема (спасибо моему преподавателю за этот абзац).

### Императивный

Языки программирования также могут быть разделены на импертивные и декларативные. В императивном языке программист будет указывать последовательность команд для выполнения (это все языки программирования которые мы считаем «языками программирования» простите за тавтологию).

Декларативные же языки в свою очередь ожидают от нас описания результата, который мы хотим получить в ходе выполнения запроса. Яркий пример декларативного языка это SQL (Stucted Query Language) или же структурированный язык запросов. Именно в нём мы описываем конкретный результат выполнения программы а не последовательность команд. Декларативными языками еще называют HTML, CSS, SVG, VRML, SQL, lex/VACC.

* Строго типизированный

Что касается типизации. В сильно типизированном языке интерпретатор, при выполнении команд, не станет неявно приводить типы в отличии от слабо типизированных языков, в котором приведения типа могут происходить неявно.

### Общего назначения

Языки делятся на языки общего назначения (Python, Java, Kotlin, C, Ruby) и специализированные (например предметно-ориентированные языки или DSL)

DSL(предметно-ориентированный язык) — компьютерный язык, специализированный для конкретной области применения (в противоположность языку общего назначения, применимому к широкому спектру областей и не учитывающему особенности конкретных сфер знаний). Построение такого языка и/или его структура данных отражают специфику решаемых с его помощью задач. Является ключевым понятием языково- ориентированного программирования.  Примеры таких языков Perl, SQL, HTML, Haskell, Verilog, AutoLisp

### Динамическая типизация

Динамическая типизация предполагает, что в процессе выполнения команды переменная может содержать объекты различных типов. То есть мы объявляем переменную не указывая явно, какой тип данных в ней будет содержаться, и в процессе выполнения программы в одной переменной может побывать как текст так и число, а может и булево значение.

Статическая типизация предполагает, что при задании переменной сразу указывается тип данных, который она может содержать.

1.2 Особенности Python

Интроспекция в программировании это это способность программы получать различную информацию об объектах во время работы программы. Если конкретно, то вы можете узнать какому классу принадлежит объект, с каким типом данных вы работаете в данный момент (помните про динамическую типизацию?) или список атрибутов и методов, доступных для указанного объекта

Примеры полезных функций интроспекции в Python:

* dir();
* type();
* hasattr();
* id();
* isinstance().

Python не отстает, он идёт, что называется, в коробке с Linux и SQL Server, интерпретатор Python можно легко поставить на Windows, не забыли про macOS, и iOS. Да вообще везде работает, в 8 раз медленней чем Swift на iOS, но работает.

В Python существуют свои встроенные паттерны проектирования, лучшим примером для понимания служат Декоратор и Итератор.

Декоратор позволяет расширять функциональность без наследования. Чтобы разобраться в паттерне Итератор, следует просто изучить механизм работы итераторов и генераторов языка.

При использовании Core Python вы получаете «в коробке» большой набор полезных стандартных библиотек. Примером может служить библиотека для работы с БД SQLite, а также json, math, re, random, csv, html и много других.

Форматирование текста в Python является частью его синтаксиса и да, это те самые отступы, которые позволяют повышать читаемость кода.

1.3 Описание среды разработки PyCharm

PyCharm — интегрированная среда разработки для языка программирования Python. Предоставляет средства для анализа кода, графический отладчик, инструмент для запуска юнит-тестов и поддерживает веб-разработку на Django. PyCharm разработана компанией JetBrains на основе IntelliJ IDEA.

PyCharm— это кроссплатформенная среда разработки, которая совместима с Windows, macOS, Linux. PyCharm Community Edition (бесплатная версия) находится под лицензией Apache License, а PyCharm Professional Edition (платная версия) является проприетарным ПО.

# Выполнение практических заданий

2.1 Парсинг

Парсинг (Parsing) – это принятое в информатике определение синтаксического анализа. Для этого создается математическая модель сравнения лексем с формальной грамматикой, описанная одним из языков программирования. Например, PHP, Perl, Ruby, Python.

Когда человек читает, то, с точки зрения науки филологии, он совершает синтаксический анализ, сравнивая увиденные на бумаге слова (лексемы) с теми, что есть в его словарном запасе (формальной грамматикой).

Программа (скрипт), дающая возможность компьютеру «читать» – сравнивать предложенные слова с имеющимися во Всемирной сети, называется парсером. Сфера применения таких программ очень широка, но все они работают практически по одному алгоритму.Независимо от того на каком формальном языке программирования написан парсер, алгоритм его действия остается одинаковым:

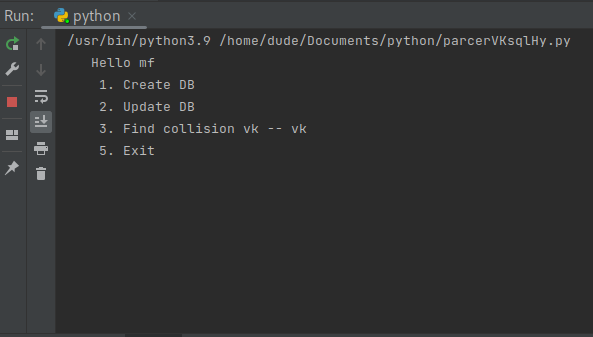
1. выход в интернет, получение доступа к коду веб-ресурса и его скачивание;
2. чтение, извлечение и обработка данных;
3. представление извлеченных данных в удобоваримом виде – файлы .txt, .sql, .xml, .html и других форматах.

В интернете часто встречаются выражения, из которых следует, будто парсер (поисковый робот, бот) путешествует по Всемирной сети. Но зачастую эта программа никогда не покидает компьютера, на котором она инсталлирована.

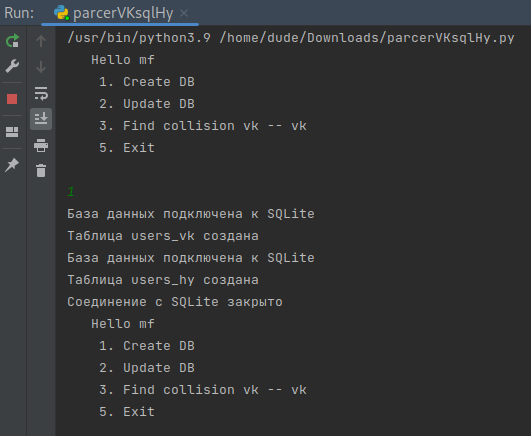
Этим парсер коренным образом отличается от компьютерного вируса – автономной программы, способной к размножению, хотя по сути своей работы он похож на трояна. Ведь он получает данные, иногда конфиденциального характера, не спрашивая желания их владельца.

2.2 Парсер ВК

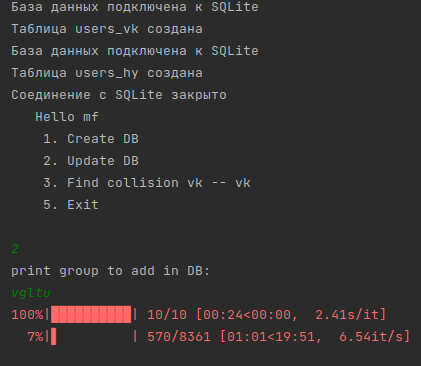
При первом запуске программы требуется последовательно выполнить первый и второй пункт (рис 1)

Рисунок 1

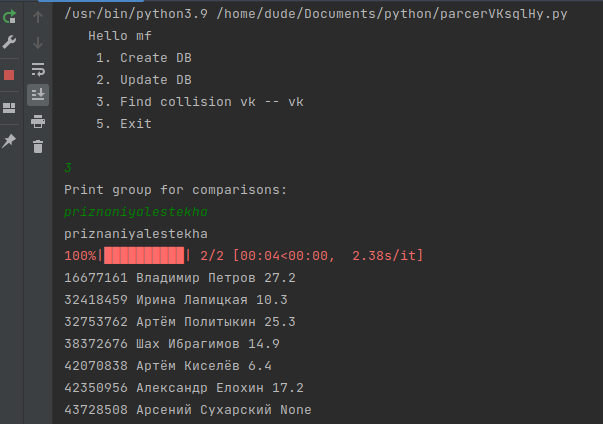
Первый пункт отвечает за создание баз данных (рис 2)

Рисунок 2

При выборе второго пункта требуется ввести id группы и если подписчики не скрыты администратором, они добавятся в базу данных (рис 3)

Рисунок 3

Далее пункт 3 для сравнения аудиториий нескольких групп (рис 4)

Рисунок 4

# ЗАКЛЮЧЕНИЕ

На основании материала, изученного в данной работе, можно сформулировать следующие выводы:

Возникновение языка Python было обусловлено прогрессом технологий и возрастающими достижениями в области программирования и разработки

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ.

Python стал одним из самых популярных языков, он используется в анализе данных, машинном обучении, DevOps и веб-разработке, а также в других сферах, включая разработку игр. За счёт читабельности, простого синтаксиса и отсутствия необходимости в компиляции язык хорошо подходит для обучения программированию, позволяя концентрироваться на изучении алгоритмов, концептов и парадигм. Отладка же и экспериментирование в значительной степени облегчаются тем фактом, что язык является интерпретируемым.

В результате выполнения данной работы изучены возможности языка программирования Python. Согласно заданиям разработаны интерфейсы пользователя, продуманы алгоритмы и написан программный код.

# СПИСОК ЛИТЕРАТУРЫ

1. «Изучаем Python. 3-е издание» Марк Лутц 2017
2. «Высокопроизводительный Python: практическое пособие для людей», Миша Горелик, Ян Освальд

## «Python. Разработка на основе тестирования», Гарри Персиваль

## «Python. К вершинам мастерства», Лучано Рамальо

ПРИЛОЖЕНИЕ

import sqlite3  
import sys  
from tqdm import tqdm  
import vk  
import time  
  
  
token = "d3368746d3368746d336874678d34a4ed9dd336d3368746b1a6fe82798777c57b6ed251" # Сервисный ключ доступа  
session = vk.Session(access\_token=token)  
vk\_api = vk.API(session)  
  
  
def get\_members(groupid): #получает id группы и возвращает список подписчиков  
 allusers = []  
 try:  
 first = vk\_api.groups.getMembers(group\_id=groupid, v=5.81,count = "1", lang='ru')  
 count = first["count"] // 1000  
 for i in tqdm(range(count + 1)):  
 people = vk\_api.groups.getMembers(group\_id=groupid, v=5.81, offset=1000\*i,count = "1000", fields="first\_name, last\_name, contacts, bdate", lang='ru')["items"]  
 allusers = allusers + people  
 time.sleep(2)  
 except Exception as error:  
 print(error)  
 return allusers  
  
  
def create\_db(): #создает базу данных и таблцы  
 try:  
 sqlite\_connection = sqlite3.connect('users.db')  
 sqlite\_create\_table\_query = '''CREATE TABLE users\_vk (  
 id INTEGER PRIMARY KEY,  
 first\_name TEXT NOT NULL,  
 last\_name text NOT NULL,  
 bdate text,  
 mobile\_phone text,  
 home\_phone text,  
 CONSTRAINT unique\_id UNIQUE(id));'''  
 cursor = sqlite\_connection.cursor()  
 print("База данных подключена к SQLite")  
 cursor.execute(sqlite\_create\_table\_query)  
 sqlite\_connection.commit()  
 print("Таблица users\_vk создана")  
 cursor.close()  
 sqlite\_connection = sqlite3.connect('users.db')  
 sqlite\_create\_table\_query = '''CREATE TABLE users\_hy (  
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 first\_name TEXT NOT NULL,  
 last\_name text NOT NULL,  
 middle\_name text NOT NULL,  
 bdate text,  
 mobile\_phone text,  
 mail text);'''  
 cursor = sqlite\_connection.cursor()  
 print("База данных подключена к SQLite")  
 cursor.execute(sqlite\_create\_table\_query)  
 sqlite\_connection.commit()  
 print("Таблица users\_hy создана")  
 cursor.close()  
 except sqlite3.Error as error:  
 print("Ошибка при подключении к sqlite", error)  
 finally:  
 if (sqlite\_connection):  
 sqlite\_connection.close()  
 print("Соединение с SQLite закрыто")  
  
  
def add\_data\_db(list):#принимает список пользователей и добавляет их в бд  
 sqlite\_connection = sqlite3.connect('users.db')  
 for i in tqdm(list):  
 try:  
 cursor = sqlite\_connection.cursor()  
 sqlite\_insert\_query = """INSERT INTO users\_vk (id, first\_name, last\_name, bdate, mobile\_phone, home\_phone) VALUES (?, ?, ?, ?, ?, ?)"""  
 data = (i["id"], i["first\_name"], i["last\_name"],i.get("bdate"), i.get("mobile\_phone"), i.get("home\_phone"))  
 cursor.execute(sqlite\_insert\_query, data)  
 sqlite\_connection.commit()  
 cursor.close()  
 except sqlite3.Error as error:  
 exc\_type, exc\_value, exc\_tb = sys.exc\_info()  
 #print("Ошибка", error)  
 finally:  
 continue  
 if (sqlite\_connection):  
 sqlite\_connection.close()  
  
  
def find\_collision\_vk(list):# получает список пользователей и сравнивает его со списком из бд. возвращает список id  
 sqlite\_connection = sqlite3.connect('users.db')  
 list\_god\_pep = []  
 try:  
 cursor = sqlite\_connection.cursor()  
 sqlite\_insert\_query = """SELECT id FROM users\_vk"""  
 cursor.execute(sqlite\_insert\_query)  
 list\_god\_pep = cursor.fetchall()  
 cursor.close()  
 except sqlite3.Error as error:  
 print("Ошибка при подключении к sqlite", error)  
 collision = []  
 for i in list:  
 for j in list\_god\_pep:  
 if i["id"] == j[0]:  
 collision.append(i)  
 if (sqlite\_connection):  
 sqlite\_connection.close()  
 return collision  
  
  
while 0 == 0: #основная функция  
 print(""" Hello mf  
 1. Create DB   
 2. Update DB  
 3. Find collision vk -- vk  
 4. Find collision hy -- vk  
 5. Exit  
 """)  
 key = input()  
 if key == "1":  
 create\_db()  
 if key == "2":  
 print("print group to add in DB: ")  
 groups = input()  
 groups = groups.split(" ")  
 for i in groups:  
 lists = get\_members(i)  
 add\_data\_db(lists)  
 if key == "3":  
 print("Print group for comparisons: ")  
 groups = input()  
 groups = groups.split(" ")  
 for i in groups:  
 with open("people.txt", "a+") as f:  
 print(i)  
 f.write(i + "\n")  
 lists = get\_members(i)  
 peop = find\_collision\_vk(lists)  
 for j in peop:  
 tmppep = "{} {} {} {}".format(j["id"], j["first\_name"], j["last\_name"], j.get("bdate"))  
 print(tmppep)  
 f.write(tmppep + "\n")  
 if key == "4":  
 print("")  
 if key == "5":  
 sys.exit()