Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

09.03.01 "Информатика и вычислительная техника"

профиль "Программное обеспечение средств вычислительной техники и автоматизированных систем"

**ОТЧЕТ**

по учебной практике

на кафедре Прикладной Математики и Кибернетики

Выполнил:

студент гр. ИП-012 \_\_\_\_\_\_\_\_\_\_\_\_\_/Ожгибесов.В.А/

«28» мая 2022г.

Руководитель практики

доцент каф. ПМиК \_\_\_\_\_\_\_\_\_\_\_\_\_/Приставка П.А./

«28» мая 2022г. Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Новосибирск 2022 г.

Оглавление

[Постановка задачи 3](#_Toc103540778)

[Описание используемых алгоритмов 4](#_Toc103540779)

[Листинг программы 6](#_Toc103540780)

# Постановка задачи

Разработать программу, реализующую ввод, хранение и обработку данных о котировках криптовалют на основе данных сайта coinmarketcap.com.

Общие требования к программе:

1. Язык разработки: **Python версии не ниже 3.x**
2. Операционная система: определяются студентом
3. Набор свойств криптовалют:
   * Name – наименование
   * Market\_cap – рыночная капитализация
   * Price – стоимость 1 ед. в долларах США (USD)
4. Ввод данных

|  |  |
| --- | --- |
| Оценки «хорошо» и  «удовлетворительно» | Оценка «отлично» |
| Из файла currencies.csv. Файл содержит данные о 25 наиболее ценных криптовалютах на 29.01.2020 в формате:  Name; Market\_cap; Price  Файл доступен для скачивания в ЭИОС в директории с заданием на практику. | Непосредственно с главной страницы страницы сайта coinmarketcap.com в момент запуска программы.  Загрузка и парсинг веб-страницы производится с помощью библиотек Requests и Beautifulsoup или их аналогов |

1. Хранение

Типы и структуры для хранения данных: определеяются студентом

1. Обработка

Реализовать функцию поиска информации о свойствах криптовалюты по ее названию

# Описание используемых алгоритмов

def parser\_file(data) – проверяет наличие файла“currencies22.csv” в директории. В случае его наличия осуществляет парсинг файла, иначе вызывает исключение Ошибка и завершает программу.

**Принимает**:

* Data – список словарей

**Возвращает**: список словарей с ключами: name, marketCap, price

def get\_html(url) – получает html код сайта.

**Принимает**:

* URL – ссылка сайта

**Возвращает**: список словарей с ключами: name, marketCap, price

def get\_content(html) - получает с сайта данные криптовалюты, как name, marketCap, price

**Принимает**:

* URL – ссылка сайта

def parse() – проверяет доступность сайта и если он доступен, осуществляет парсинг сайта “<https://coinmarketcap.com/>”.

def search\_list(data, key) – осуществляет линейный поиск подстроки в строке по заданному ключу

**Принимает**:

* Data – список словарей
* Key – ключ поиска

**Возвращает**: список словарей, соответствующих ключу

def create\_json(data) – создает json файл

**Принимает**:

* Data – список словарей

def create\_csv(data) – создает csv файл

**Принимает**:

* Data – список словарей

def print\_data(data) – осуществляет вывод списка

**Принимает**:

* Data – список словарей

def main() – главная функция, осуществляет логическую составляющую скрипта(вызов необходимых функций).

# Листинг программы

import csv

import json

import time

from bs4 import BeautifulSoup

from selenium import webdriver

from selenium.webdriver.chrome.service import Service

from webdriver\_manager.chrome import ChromeDriverManager

import pandas as pd

FILE\_NAME = "currencies22.csv"

URL = "https://coinmarketcap.com/"

HEADERS = {"user-agent": "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/101.0.4951.54 Safari/537.36"}

S\_BOLD = "\033[1m"

S\_RESET = "\033[0m"

C\_RED = "\033[31m"

C\_GREEN = "\033[32m"

C\_YELLOW = "\033[33m"

C\_CYAN = "\033[36m"

cryptocurrency = []

def parser\_file(data):

    try:

        file = open(FILE\_NAME, 'r')

        table = csv.reader(file, delimiter=';')

        for row in table:

            item = {"name":      row[0],

                    "marketCap": row[1],

                    "price":     row[2]

                    }

            data.append(item)

        file.close()

    except FileNotFoundError:

        print(C\_RED + S\_BOLD + f"Ошибка! Файл \'{FILE\_NAME}\' не обнаружен!")

        print(S\_RESET)

        exit()

    return data

def print\_data(data):

    print(S\_BOLD + C\_GREEN)

    print(f"{'Название':35}  {'Рыночная капитализация':>25}    {'Цена':>15}")

    print(C\_CYAN)

    for item in data:

        print(f"{item['name']:35}  {item['marketCap']:>25}    {item['price']:>15}")

    print(C\_YELLOW)

    print("Кол-во элементов: ", len(data))

    print(S\_RESET)

def get\_html(url, params=None):

    driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))

    driver.get(url)

    time.sleep(1)

    heig\_d=500

    while heig\_d <10000:

        driver.execute\_script(f"0,window.scrollTo(0, {heig\_d});")

        heig\_d += 500

        time.sleep(0.5)

    page = driver.page\_source

    return page

def get\_content(html):

    soup = BeautifulSoup(html, 'html.parser')

    page = soup.find(class\_="h7vnx2-1 bFzXgL").find\_all('tr')

    for row in page:

        if (row.find(class\_="sc-1eb5slv-0 iworPT") != None):

            items = {

                "name":row.find(class\_='sc-1eb5slv-0 iworPT').text,

                "marketCap": row.find(class\_='sc-1ow4cwt-0 iosgXe').text,

                "price": row.find(class\_='sc-131di3y-0 cLgOOr').text

            }

            cryptocurrency.append(items)

def parse():

    html = get\_html(URL)

    get\_content(html)

def create\_json(data):

    with open('data.json', 'w', encoding='utf-8') as file:

        json.dump(data, file, indent=4, ensure\_ascii='False')

def create\_csv(data):

    dataframe = pd.DataFrame(data)

    dataframe.to\_csv('data.csv', index=False, sep=';')

def search\_list(data, key):

    items = []

    for item in data:

        if item.get("name").upper().startswith(key.upper()):

            items.append(item)

    return items

def menu():

    flag = False

    cryptocurrency.clear()

    while True:

        if cryptocurrency:

            print\_data(cryptocurrency)

        print("1. Parsing")

        if flag == False:

            print("2. Read File")

        if flag == True:

            print("2. Search")

            print("3. Create JSON")

            print("4. Create CSV")

        print("0. Exit")

        cmd = input("Select: ")

        if cmd == "1":

            try:

                parse()

                flag = True

            except Exception:

                print(C\_RED +"Ошибка" + S\_RESET)

                flag = False

        elif cmd == "2" and flag == True:

            found = search\_list(cryptocurrency, input("Введите строку для поиска криптовалюты: "))

            if found:

                print\_data(found)

            else:

                print(C\_RED + "Криптовалюты не найдены!" + S\_RESET)

        elif cmd =="2" and flag == False:

            parser\_file(cryptocurrency)

            print\_data(cryptocurrency)

        elif cmd == "3" and flag == True:

            create\_json(cryptocurrency)

        elif cmd == "4" and flag == True:

            create\_csv(cryptocurrency)

        elif cmd == "0":

            print()

            break

        else:

            print(C\_RED +"Ошибка! Некорректный ввод!" + S\_RESET)

menu()