Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

«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 \_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

ФИО студента

«15» мая 2022г.

Руководитель практики

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

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

Новосибирск 2022 г.

**Оглавление**

[Условие задачи 3](#_Toc103544870)

[Описание алгоритмов 3](#_Toc103544871)

[Листинг программы 5](#_Toc103544872)

[Результаты тестирования 8](#_Toc103544873)

[Список использованных источников 10](#_Toc103544874)

# **Условие задачи**

Разработать программу, реализующую ввод, хранение и обработку данных о не более 25 криптовалютах на основе данных сайта coinmarketcap.com в браузере Chrome. Реализовать программу на Python3.10, на операционной системе Windows10, хранящую следующие свойства криптовалют:

* Name – наименование
* Market\_cap – рыночная капитализация
* Price – стоимость 1 ед. в долларах США (USD)

Реализовать ввод данных с сайта в момент запуска программы с помощью библиотек BeautifulSoup и сохранение полученных данных в CSV-таблице.

Реализовать поиск информации о свойствах криптовалюты по её названию.

# **Описание алгоритмов**

Для реализации автоматического сбора данных с сайта и их структурирования в функции CryptoParsing(), которая вызывается в меню в пункте 1), я воспользовался библиотеками Selenium, Webdriver\_Manager и BeautifulSoup. В функции CryptoParsing(url) сначала происходит установка нужной версии драйвера и запуск браузера Chrome, после чего браузеру передаётся URL-адрес, по которому открывается сайт coinmarketcap.com.

Так как страница сайта является динамической – страница сайта формируется по мере просмотра, то необходимо сначала её всю прокрутить, после чего собирать данные о первых криптовалютах. Для этого вызывается скрипт, который возвращает высоту страницы, и вплоть до этого значения происходит постепенный скроллинг страницы с перерывами в 1 одну секунду для подгрузки страницы. Реализована задержка с помощью метода time.sleep(1) библиотеки time.

После скроллинга страницы в переменную html сохраняется её html-код и дальнейшей обработки кода используются BeatifulSoup() с html.parser. Исследовав код страницы сайта было обнаружено, что данные внутри тегов 'tr', принадлежавших div с классом 'bFzXgL'.

Поэтому с помощью команды soup.find(class\_="bFzXgL").find\_all('tr') я ищу все вхождения тегов 'tr' в этот div b сохраняю их в отдельный список.

Изображение выглядит как текст

Автоматически созданное описание

**Рис.1.** HTML – код страницы сайта coinmarketcap.com

Внутри этого списка поэлементно происходит сбор данных по классам, которые, согласно html-коду, хранят в себе информацию о Name, Price и Market\_cap криптовалют. Так как на одной страницу находится 100 криптовалют, а по условию задания их необходимо считать не больше 25, необходимости переключаться между страницами не было, и при достижении этого значения парсинг сайта останавливается. Полученная информация хранится в списке списков, который возвращается как результат выполнения функции.

При необходимости можно просмотреть данные о первых 25 считанных криптовалют, вызвав функцию PrintCrypto(data) из меню под пунктом 2). Вывод в виде таблице реализован с помощью библиотеки Pandas, так как с её помощью можно полученный список списков можно преобразовать в DataFrame, который легко можно вывести с помощью функции print().

Также в программе была прописана функция SearchCrypto(data, key) в меню под пунктом 3), в которой был реализован поиск криптовалют по первым буквам их наименований без учёта регистра и с последующим их выводом в виде таблицы.

Для сохранения полученных данных в виде CSV-таблицы была подключена библиотека Pandas, которая, после преобразования полученного списка списков о криптовалютах в DataFrame, легко сохраняет его в виде CSV-таблицы. Данный функционал реализован в функции CreateCSV (data) в меню под пунктом 4).

В этой функции было дополнительно реализовано сохранение данных в файл с названием, соответствующим текущей дате. Для этого была подключена библиотека datetime, с помощью которой производится расчёт системного времени.

Вместо того, чтобы парсить сайт в условиях, если не будет доступа к сети Интернет, была реализована функция LoadCSV() в меню под пунктом 5). Эта функция преобразует данные из CSV-таблицы в DataFrame и возвращает с помощью функции dataframe.values.tolist().copy() список списков, содержащий информацию о всех криптовалютах в этой таблице. В результате можно также, как после парсинга сайта, просмотреть таблицу и искать криптовалюты по первым буквам наименования.

Дополнительно, подключив библиотеку colorama, был изменён цвет консольного шрифта на зелёный, а также добавлена возможность очищать терминал с помощью Escape-последовательности перед вызовом каждой функции из меню, а также во время парсинга сайта.

Для выхода из программы есть 0) пункт с названием "Exit".

# **Листинг программы**

import time

import colorama

import pandas as pd

from datetime import datetime

from selenium import webdriver

from webdriver\_manager.chrome import ChromeDriverManager

from colorama import Fore, Back, Style

from bs4 import BeautifulSoup

colorama.init()

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

FileName = str(datetime.now().date())+'.csv'

ColumnsName = ['name','price','marketcap']

def CryptoParsing():

LimitData = 25

Data=[]

count=0

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

print('\033[2J')

print('\033[H')

driver.get(URL)

print('\033[2J')

print('\033[H')

for heig\_d in range(0, driver.execute\_script("return document.body.scrollHeight"), 500):

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

print('\033[2J')

print('\033[H')

time.sleep(1)

html = driver.page\_source

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

items = soup.find(class\_="bFzXgL").find\_all('tr')

for item in items:

if (item.find(class\_="iworPT") != None):

item\_name = item.find(class\_='iworPT').text

item\_priсe = item.find(class\_='cLgOOr').text

item\_marketcap = item.find(class\_='iosgXe').text

Data.append([item\_name,item\_priсe,item\_marketcap])

count+=1

if(count==LimitData):

break

print('\033[2J')

print('\033[H')

return Data

def PrintCrypto(data):

dataframe = pd.DataFrame(data,columns=ColumnsName)

print(dataframe.head(25))

def SearchCrypto(data, key):

items = []

for item in data:

if item[0].upper().startswith(key.upper()):

items.append(item)

return items

def CreateCSV(data):

dataframe = pd.DataFrame(data,columns=ColumnsName)

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

def LoadCSV():

dataframe = pd.read\_csv(FileName, sep=';')

data=dataframe.values.tolist().copy()

return data

def menu():

CryptoCurrency = []

while True:

print("1. Parsing CryptoCurrency")

print("2. Print CryptoCurrency")

print("3. Search CryptoCurrency")

print("4. Create CSV")

print("5. Load today's CSV")

print("0. Exit")

sw = input("Select: ")

print('\033[2J')

print('\033[H')

if sw == "1":

try:

CryptoCurrency=CryptoParsing()

print("Parsing was completed")

except Exception:

CryptoCurrency.clear()

print("Please try again")

elif sw == "2":

PrintCrypto(CryptoCurrency)

elif sw == "3":

key = input("Enter (name): ")

Items=SearchCrypto(CryptoCurrency,key)

if not Items:

print("CryptoCurrency hasn't been found!!!")

else:

print()

i = 0

print("{0:>21} {1:>15} {2:>15}".format(ColumnsName[0],ColumnsName[1],ColumnsName[2]))

for item in SearchCrypto(CryptoCurrency,key):

print("{0}{1:>20} {2:>15} {3:>15}".format(i,item[0],item[1],item[2]))

i += 1

elif sw == "4":

CreateCSV(CryptoCurrency)

print("File CSV was created")

elif sw == "5":

try:

CryptoCurrency=LoadCSV()

print("Loading was completed")

except Exception:

CryptoCurrency.clear()

print("File CSV not found")

elif sw == "0":

break

print()

print('\033[2J')

print('\033[H')

print('\033[32m')

menu()

# **Результаты тестирования**

Изображение выглядит как текст

Автоматически созданное описание

**Рис. 2.** Парсинг сайта с помощью selenium

Изображение выглядит как текст, компьютер, электроника, ноутбук

Автоматически созданное описание

**Рис. 3.** Удачное завершение парсинга

Изображение выглядит как текст, компьютер, ноутбук, электроника

Автоматически созданное описание

**Рис. 4.** Список данных о 25 криптовалют

Изображение выглядит как текст, компьютер, внутренний, электроника

Автоматически созданное описание

**Рис. 5.** Результат поиска криптовалют по первым буквам в списке

Изображение выглядит как текст, компьютер, снимок экрана

Автоматически созданное описание

**Рис. 6.** CSV-таблица с результатами парсинга сайта

Изображение выглядит как текст, компьютер, внутренний, монитор

Автоматически созданное описание

**Рис. 7.** Удачное завершение загрузки данных из CSV-таблицы

Изображение выглядит как текст, компьютер, снимок экрана, внутренний

Автоматически созданное описание

**Рис. 8.** Список криптовалют, загруженных из CSV-таблицы

# **Список использованных источников**

1. Васильев А.Н. Программирование на Python [Электронный ресурс] URL: [https://www.litres.ru/aleksey-nikolaevich/programmirovanie-na-python-v-primerah-i-zada-63455717/](https://www.litres.ru/aleksey-nikolaevich/programmirovanie-na-python-v-primerah-i-zada-63455717/%20) (дата обращение: 15.05.2022)
2. Чтение и запись CSV-файлов в Python с Pandas [Электронный ресурс] URL: <https://pythobyte.com/reading-and-writing-csv-files-in-python-with-pandas-4e74976e/>
3. Webdriver Manager for Python [Электронный ресурс] URL: <https://pypi.org/project/webdriver-manager/>
4. Beautiful Soup Documentation [Электронный ресурс] URL: [https://www.crummy.com/software/BeautifulSoup/bs4/doc/](https://www.crummy.com/software/BeautifulSoup/bs4/doc/%20)
5. Цветной вывод текста в Python [Электронный ресурс] URL: <https://all-python.ru/osnovy/tsvetnoj-vyvod-teksta.html>