

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-014

_____/Альхимович М.В./
ФИО студента

«28» мая 2022г.

Руководитель практики

доцент каф. ПМиК

_____/Приставка П.А./

«28» мая 2022г.

Оценка _____

Новосибирск 2022 г.

**Задание по учебной практике для студентов очной формы обучения направления
09.03.01 «Информатика и вычислительная техника» профиль
«Программное обеспечение средств вычислительной техники и
автоматизированных систем»**

Текст задания

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»
<p>Из файла <code>currencies22.csv</code>. Файл содержит данные о 25 наиболее ценных криптовалютах на 05.03.2022 в формате</p> <p style="text-align: center;">Name; Market_cap; Price</p> <p>Файл доступен для скачивания в ЭИОС в директории с заданием на практику.</p>	<p>Непосредственно с главной страницы сайта coinmarketcap.com в момент запуска программы.</p> <p>Загрузка и парсинг веб-страницы производится с помощью библиотек <code>Requests</code> и <code>Beautifulsoup</code> или их аналогов</p> <p>Примечание: допускается считывание строчек в количестве менее 25 (Например, 10 строчек с данными о криптовалютах)</p>

5. Хранение

Типы и структуры для хранения данных: определяются студентом

6. Обработка

Реализовать функцию поиска информации о свойствах криптовалюты по ее названию.

Описание алгоритмов

Стандарт Python: Python 3.10.4

Используемые библиотеки и модули:

CSV (comma-separated value) - это формат представления табличных данных (например, это могут быть данные из таблицы или данные из БД).

В стандартной библиотеке Python есть модуль csv, который позволяет работать с файлами в CSV формате.

```
import csv
```

Requests - библиотека, позволяющая отправлять запросы HTTP в Python и получать ответы, соответственно.

```
import requests
```

Requests.exceptions - расширение для библиотеки Requests, которое необходимо для отлавливания исключений ответов с запроса, таких как:

ConnectionError - ошибка соединения, Timeout - превышено ожидание ответа,

TooManyRedirects - слишком много перенаправлений.

```
from requests.exceptions import ConnectionError, Timeout, TooManyRedirects
```

BeautifulSoup - это пакет Python для анализа документов HTML и XML. Он создает дерево синтаксического анализа для проанализированных страниц, которое можно использовать для извлечения данных из HTML, что полезно для парсинга веб-страниц.

```
from bs4 import BeautifulSoup
```

Используемые функции:

def fill_struct_list(): - функция; создает двумерный массив, который содержит в себе структуры, сформированные с помощью csv.reader по разделителю ';' из исходной таблицы. Затем форматирует информацию в удобный для работы и чтения вид.

Возвращаемое значение: массив структур tmp_struct_list.

def print_table(struct_list): - процедурная функция; создает структуру с помощью конструктора PrettyTable(), а затем построчно добавляет информацию в данную структуру по заранее заданным нами полям "Name", "Market_cap", "Price". Выводит всю информацию в виде приятной для глаза таблицы.

def search_table(struct_list): - функция; функция поиска по ключу в переданной структуре. Находит по ключу необходимое поле и создает новую структуру, в которую будут переданы все строки таблицы с необходимыми данными.

Возвращаемое значение: массив структур `final_elements_arr`.

def parse_site(): - функция; создает header, который был вручную добавлен при первом вхождении мной на сайт: User-Agent. Он необходим для того, чтобы в последующих запросах алгоритма на сайт, сам сайт расценивал нас как пользователя, а не скрипт. Таким образом, мы сможем отправлять запросы без риска быть заблокированы защитой сайта. После этого, была считана вся информация с сайта и попутно переведена в lxml формат с помощью пакета анализа документов BeautifulSoup. Затем выделена необходимая информация по тегу "tbody", а далее найдена вся информация по тегам "p", "span" и "div", для нахождения параметров таблицы Name, Market_cap и Price, соответственно. Далее вся информация занесена по прошлому методу в массив структур с данными.

Возвращаемое значение: массив структур `tmp_struct_list`.

main(): - интерфейс пользователя и разделение на задачи для считывания с файла и с сайта.

Листинг программы

```
import csv
import requests
from bs4 import BeautifulSoup
from prettytable import PrettyTable
from requests.exceptions import ConnectionError, Timeout, TooManyRedirects

def fill_struct_list():
    tmp_struct_list = []
    try:
        file_path = open("currencies22.csv", "r")
        table = csv.reader(file_path, delimiter=';')
        for i in table:
            item = {"Name": i[0], "Market_cap": i[1], "Price": i[2]}
            tmp_struct_list.append(item)
        file_path.close()
    except FileNotFoundError:
        print("No file")

    return tmp_struct_list

def print_table(struct_list):
    table = PrettyTable()
    table.field_names = ["Name", "Market_cap", "Price"]
```

```

for i in struct_list:
    table.add_row([i["Name"], i["Market_cap"], i["Price"]])
print(table)

def search_table(struct_list):
    print("Write key for search")
    key = input()
    fined_elements_arr = []
    for i in struct_list:
        if i["Name"] == key:
            fined_elements_arr.append(i)

    return fined_elements_arr

def parse_site():
    tmp_struct_list = []
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.84 Safari/537.36
OPR/85.0.4341.79"}
    try:
        response = requests.get(url="https://coinmarketcap.com", headers=headers)
        print(response) # Response [200] - ответ: успешно получили ответ с сервера и
зашли на него
        soup = BeautifulSoup(response.text, "lxml")
        html_table = soup.find("tbody")
        name_arr = html_table.find_all("p", class_="sc-1eb5slv-0 iworPT")
        market_cap_arr = html_table.find_all("span", class_="sc-1ow4cwt-1 ieFnWP")
        price_arr = html_table.find_all("div", class_="sc-131di3y-0 cLg00r")
        for i in range(len(name_arr)):
            item = {"Name": name_arr[i].text, "Market_cap": market_cap_arr[i].text,
"Price": price_arr[i].text}
            tmp_struct_list.append(item)
        except(ConnectionError, Timeout, TooManyRedirects) as exp:
            print("Error", exp, sep=":")

    return tmp_struct_list

def main():
    print("Write programm mode:",
        "1 - Read from file.",
        "2 - Read from site.",
        "e - Exit", sep='\n')

```

```

struct_list = []
while True:
    input_key = input()
    if input_key in ['1', '2', 'e']:
        if input_key == 'e':
            break;
        elif input_key == '1':
            exit_flag = 0
            struct_list = fill_struct_list()
            print_table(struct_list)
            print_table(search_table(struct_list))
        elif input_key == '2':
            struct_list = parse_site()
            print_table(struct_list)
            print_table(search_table(struct_list))
    else:
        print("Unvalid input.")

if __name__ == "__main__":
    main()

```

Результаты тестирования

```

C:\Users\SeveralCamper\Desktop\Edu.Practice\Educational-practice>python educational_practice.py
Write programm mode:
1 - Read from file.
2 - Read from site.
e - Exit

```

```

C:\Users\Server\Desktop\learn\python\educational_practice\python_educational_practice.py
Write program mode:
- Read from file.
- Read from site.
- Exit

```

Name	Market_cap	Price
Bitcoin	\$39,042.91	\$740,185,900,096
Ethereum	\$2,629.03	\$314,699,571,974
Tether	\$1.00	\$79,727,654,799
BNB	\$375.66	\$61,923,153,319
USD Coin	\$1.00	\$52,874,805,350
XRP	\$0.7222	\$34,571,861,293
Terra	\$83.70	\$30,924,825,975
Cardano	\$0.8485	\$28,572,471,711
Solana	\$88.44	\$27,996,936,059
Avalanche	\$76.12	\$20,206,218,196
Binance USD	\$0.9996	\$17,945,162,902
Polkadot	\$16.74	\$16,527,777,456
Dogecoin	\$0.1229	\$16,305,497,774
TerraUSD	\$1.00	\$13,551,925,953
Shiba Inu	\$0.00002401	\$13,184,282,172
Polygon	\$1.48	\$11,259,979,784
Wrapped Bitcoin	\$39,020.00	\$10,401,835,175
Cronos	\$0.4018	\$10,149,874,833
Dai	\$0.9999	\$9,732,171,809
Cosmos	\$30.20	\$8,642,212,818
Litecoin	\$101.93	\$7,111,312,915
NEAR Protocol	\$10.62	\$6,842,198,633
Chainlink	\$13.86	\$6,457,496,696
Uniswap	\$8.92	\$6,128,650,374
TRON	\$0.05834	\$5,928,496,012

Write key for search

```

2 - Read from site.
e - Exit

```

Name	Market_cap	Price
Bitcoin	\$39,042.91	\$740,185,900,096
Ethereum	\$2,629.03	\$314,699,571,974
Tether	\$1.00	\$79,727,654,799
BNB	\$375.66	\$61,923,153,319
USD Coin	\$1.00	\$52,874,805,350
XRP	\$0.7222	\$34,571,861,293
Terra	\$83.70	\$30,924,825,975
Cardano	\$0.8485	\$28,572,471,711
Solana	\$88.44	\$27,996,936,059
Avalanche	\$76.12	\$20,206,218,196
Binance USD	\$0.9996	\$17,945,162,902
Polkadot	\$16.74	\$16,527,777,456
Dogecoin	\$0.1229	\$16,305,497,774
TerraUSD	\$1.00	\$13,551,925,953
Shiba Inu	\$0.00002401	\$13,184,282,172
Polygon	\$1.48	\$11,259,979,784
Wrapped Bitcoin	\$39,020.00	\$10,401,835,175
Cronos	\$0.4018	\$10,149,874,833
Dai	\$0.9999	\$9,732,171,809
Cosmos	\$30.20	\$8,642,212,818
Litecoin	\$101.93	\$7,111,312,915
NEAR Protocol	\$10.62	\$6,842,198,633
Chainlink	\$13.86	\$6,457,496,696
Uniswap	\$8.92	\$6,128,650,374
TRON	\$0.05834	\$5,928,496,012

Write key for search

XRP

Name	Market_cap	Price
XRP	\$0.7222	\$34,571,861,293

Polkadot	\$16.74	\$16,527,777,456
Dogecoin	\$0.1229	\$16,305,497,774
TerraUSD	\$1.00	\$13,551,925,953
Shiba Inu	\$0.00002401	\$13,184,282,172
Polygon	\$1.48	\$11,259,979,784
Wrapped Bitcoin	\$39,020.00	\$10,401,835,175
Cronos	\$0.4018	\$10,149,874,833
Dai	\$0.9999	\$9,732,171,809
Cosmos	\$30.20	\$8,642,212,818
Litecoin	\$101.93	\$7,111,312,915
NEAR Protocol	\$10.62	\$6,842,198,633
Chainlink	\$13.86	\$6,457,496,696
Uniswap	\$8.92	\$6,128,650,374
TRON	\$0.05834	\$5,928,496,012

Write key for search

XRP

Name	Market_cap	Price
XRP	\$0.7222	\$34,571,861,293

```

2
<Response [200]>

```

Name	Market_cap	Price
Bitcoin	\$573,079,758,381	\$30,098.09
Ethereum	\$249,809,402,436	\$2,068.19
Tether	\$75,664,748,915	\$0.9988
USD Coin	\$51,094,220,810	\$1.00
BNB	\$48,965,538,444	\$299.89
XRP	\$20,674,267,676	\$0.4277
Cardano	\$18,670,831,653	\$0.5532
Solana	\$17,810,087,696	\$52.00
Binance USD	\$17,738,078,875	\$1.00
Dogecoin	\$11,897,907,171	\$0.08968

Write key for search

```

Cronos      | $0.4018 | $10,149,874,833 |
Dai         | $0.0999 | $9,732,171,809  |
Cosmos      | $30.20  | $8,642,212,818  |
Litecoin    | $101.93 | $7,111,312,915  |
NEAR Protocol | $10.62  | $6,842,198,633  |
Chainlink   | $13.86  | $6,457,496,696  |
Uniswap     | $8.92   | $6,128,650,374  |
TRON        | $0.05834 | $5,928,496,012 |
+-----+
Write key for search
XRP
+-----+
| Name | Market_cap | Price |
+-----+
| XRP  | $0.7222   | $34,571,861,293 |
+-----+
2
<Response [200]>
+-----+
| Name | Market_cap | Price |
+-----+
| Bitcoin | $573,079,758,381 | $30,098.09 |
| Ethereum | $249,809,402,436 | $2,068.19 |
| Tether | $75,664,748,915 | $0.9988 |
| USD Coin | $51,094,220,810 | $1.00 |
| BNB | $48,965,538,444 | $299.89 |
| XRP | $20,674,267,676 | $0.4277 |
| Cardano | $18,670,831,653 | $0.5532 |
| Solana | $17,810,087,696 | $52.80 |
| Binance USD | $17,738,078,875 | $1.00 |
| Dogecoin | $11,897,907,171 | $0.08968 |
+-----+
Write key for search
Solana
+-----+
| Name | Market_cap | Price |
+-----+
| Solana | $17,810,087,696 | $52.80 |
+-----+

```

```

Cosmos      | $30.20  | $8,642,212,818  |
Litecoin    | $101.93 | $7,111,312,915  |
NEAR Protocol | $10.62  | $6,842,198,633  |
Chainlink   | $13.86  | $6,457,496,696  |
Uniswap     | $8.92   | $6,128,650,374  |
TRON        | $0.05834 | $5,928,496,012 |
+-----+
Write key for search
XRP
+-----+
| Name | Market_cap | Price |
+-----+
| XRP  | $0.7222   | $34,571,861,293 |
+-----+
2
<Response [200]>
+-----+
| Name | Market_cap | Price |
+-----+
| Bitcoin | $573,079,758,381 | $30,098.09 |
| Ethereum | $249,809,402,436 | $2,068.19 |
| Tether | $75,664,748,915 | $0.9988 |
| USD Coin | $51,094,220,810 | $1.00 |
| BNB | $48,965,538,444 | $299.89 |
| XRP | $20,674,267,676 | $0.4277 |
| Cardano | $18,670,831,653 | $0.5532 |
| Solana | $17,810,087,696 | $52.80 |
| Binance USD | $17,738,078,875 | $1.00 |
| Dogecoin | $11,897,907,171 | $0.08968 |
+-----+
Write key for search
Solana
+-----+
| Name | Market_cap | Price |
+-----+
| Solana | $17,810,087,696 | $52.80 |
+-----+
e
C:\Users\SeveralCamper\Desktop\Edu.Practice\Educational-practice>

```

Список использованных источников:

<https://docs.python.org/3/tutorial/index.html>
<https://www.8host.com/blog/rabota-s-veb-dannymi-s-pomoshhyu-requests-i-beautiful-soup-v-python-3/>
<https://www.8host.com/blog/web-scraping-stranic-s-pomoshhyu-beautiful-soup-i-python-3/>
<https://www.crummy.com/software/BeautifulSoup/bs4/doc/>
<https://docs.python-requests.org/en/latest/>
<https://pythonru.com/biblioteki/kratkoe-rukovodstvo-po-biblioteke-python-requests>
<https://pythonworld.ru/moduli/modul-csv.html>

Ссылка на github: <https://github.com/SeveralCamper/Educational-practice>