ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Факультет программной инженерии и компьютерной техники

ЛАБОРАТОРНАЯ РАБОТА № 3

ПО ДИСЦИПЛИНЕ «Технологии программирования»

Выполнили:

Юсюмбели В. И.

Плохотнюк В. С.

Толстухин М.С.

Группа: Р3411

Преподаватель:

Оголюк А. А.

Санкт-Петербург

2019

```
import sys
from html.parser import HTMLParser
from enum import Enum


class TAG(Enum):
    html = 0,
    tr = 1,
    td = 2,
    table = 3,


class STATUS(Enum):
    HTML_START = 0,
    TABLE_START = 1,
    TABLE_FOUND = 2,
    TABLE_ROW_FOUND = 3,
    RANK = 4,
    RANK_END = 5,
    MALE = 6,
    MALE_END = 7,
    FEMALE = 8,
    FEMALE_END = 9,
    ROW_END = 10,


class COLUMN(Enum):
    NONE = -1,
    RANK = 0,
    MALE = 1,
    FEMALE = 2,


def is_int(s):
    try:
        int(s)
        return True
    except ValueError:
        return False


class TableHTMLParser(HTMLParser):
    status = STATUS.HTML_START
    resultList = []
    currentRank = 0

    def handle_starttag(self, tag,
attrs):

        if tag == TAG.html.name:
            self.status =
STATUS.HTML_START

        if tag == TAG.table.name
and self.status ==
STATUS.HTML_START:
            for attr in attrs:
                if attr[0] ==
'summary' and attr[1] ==
'Popularity for top 1000':
self.resultList = []
                    self.status =
STATUS.TABLE_FOUND

        if tag == TAG.tr.name and
(self.status == STATUS.TABLE_FOUND
or self.status == STATUS.ROW_END):
            self.status =
STATUS.TABLE_ROW_FOUND

        if tag == TAG.td.name:
            if self.status ==
STATUS.TABLE_ROW_FOUND:
                self.status =
STATUS.RANK
            if self.status ==
STATUS.RANK_END:
                self.status =
STATUS.MALE
            if self.status ==
STATUS.MALE_END:
                self.status =
STATUS.FEMALE

    def handle_data(self, data):
        if self.status ==
STATUS.RANK:
            if is_int(data):
                self.currentRank =
int(data)
            else:
                self.currentRank =
-1
        if self.status ==
STATUS.MALE:
            if self.currentRank !=
-1:

self.resultList.append([data])
        if self.status ==
STATUS.FEMALE:
            if self.currentRank !=
-1:

self.resultList[self.currentRank -
1].append(data)

    def handle_endtag(self, tag):
        if tag == TAG.td.name:
            if self.status ==
STATUS.RANK:
                self.status =
STATUS.RANK_END
            if self.status ==
STATUS.MALE:
                self.status =
STATUS.MALE_END
            if self.status ==
STATUS.FEMALE:
                self.status =
STATUS.FEMALE_END
        if tag == TAG.tr.name:
```

```
            if self.status ==
STATUS.FEMALE_END:
                self.status =
STATUS.ROW_END
        if tag == TAG.table.name
and self.status == STATUS.ROW_END:
            del self.resultList[-
1]


def extr_name(filename):
    file = open(filename, 'rt')
    html = file.read()
    file.close()
    parser = TableHTMLParser()
    parser.feed(html)
    return parser.resultList


def list_names_with_rank(result,
is_male):
    return [f'{item[int(is_male)]}
{i + 1}' for i, item in
enumerate(result)]


def main():
    # print(1)
    args = sys.argv[1:]
    if not args:
        print('use: [--file] file
[file ...]')
        sys.exit(1)
    else:
        if args[0] == '--file':
            args = args[1:]

        output = ''
        output2 = ''
        for filename in args:
            output += '\'' +
filename[4:8] + '\''
            output2 += filename +
'\n'
            result =
extr_name(filename)
            count = 0
            for name in result:
                if count < 10:
                    output2 +=
f"{count + 1} {name[0]} {name[1]}"
                    output2 +=
'\n'
                else:
                    output2 +=
'\n'
                    break
                count += 1

            names =
sorted(list_names_with_rank(result
, False) +
list_names_with_rank(result,
True))
            for name in names:
                output += f",
'{name}'"
            output += '\n'

        print(output)
        print('TOP-10 RATED')
        print(output2)


if __name__ == '__main__':
    main()
```