ФЕДЕРАЛЬНОЕ АГЕНТСТВО ЖЕЛЕЗНОДОРОЖНОГО ТРАНСПОРТА
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Петербургский государственный университет путей сообщения Императора Александра I»
(ФГБОУ ВО ПГУПС)

Факультет «Автоматизация и интеллектуальные технологии»

ПИШ ИСКРА

Курсовой проект

по дисциплине

«Программирование на ЯВУ»

Выполнил обучающийся
Курс 1
Группа АСУМ-518

Чепорнюк М.А.

Проверил

Волков Е.А.

Санкт-Петербург
2025

В рамках ознакомительной практики мы ознакомились с языком программирования Python. Под конец курса, нам необходимо реализовать консольное приложение, которое осуществляет считывание текстового файла csv, состоящего из строк следующего формата:

YEAR;MONTH;DAY;HOUR;MINUTE;TEMPERATURE

Ниже приведен код такого приложения:

```
import argparse
import os
import sys

def parse_line(row, line_no):
    def warn(msg):
        print("ВНИМАНИЕ " + msg)

    for i in range(len(row)):
        row[i] = row[i].strip()

    try:
        year = int(row[0])
        month = int(row[1])
        day = int(row[2])
        hour = int(row[3])
        minute = int(row[4])
        temperature = int(row[5])
    except Exception:
        warn("Строка {} не удалось преобразовать поля в числа -> пропуск Строка {}".format(line_no, row))
        return None

    return {
        "year": year,
        "month": month,
        "day": day,
        "hour": hour,
        "minute": minute,
        "temp": temperature,
    }

def year_stats(datas):
    if not datas:
        print("ВНИМАНИЕ Нет данных для обработки")
        return None

    temp_count = 0
    temp_sum = 0

    temp_max = None
    temp_min = None
```

```
    temp_avg = None

    for line in datas:
        new_temp = line['temp']
        temp_sum = temp_sum + new_temp
        temp_count = temp_count + 1

        if (temp_min is None) or (new_temp < temp_min):
            temp_min = new_temp

        if (temp_max is None) or (new_temp > temp_max):
            temp_max = new_temp

    temp_avg = temp_sum / temp_count

    return {
        "avg": temp_avg,
        "max": temp_max,
        "min": temp_min
    }

def month_stats(datas, month_filter=None):
    if not datas:
        print("ВНИМАНИЕ Нет данных для обработки")
        return {}

    monthly_data = {}

    for line in datas:
        month = line['month']
        if month not in monthly_data:
            monthly_data[month] = []
        monthly_data[month].append(line)

    if month_filter is not None:
        if month_filter in monthly_data:
            monthly_data = {month_filter: monthly_data[month_filter]}
        else:
            print(f"ВНИМАНИЕ Нет данных для месяца {month_filter}")
            return {}

    result = {}
    for month, data in monthly_data.items():
        temp_count = 0
        temp_sum = 0
        temp_max = None
        temp_min = None

        for line in data:
            new_temp = line['temp']
            temp_sum = temp_sum + new_temp
            temp_count = temp_count + 1
```

```
            if (temp_min is None) or (new_temp < temp_min):
                temp_min = new_temp

            if (temp_max is None) or (new_temp > temp_max):
                temp_max = new_temp

        temp_avg = temp_sum / temp_count if temp_count > 0 else 0

        result[month] = {
            "avg": temp_avg,
            "max": temp_max,
            "min": temp_min,
            "count": temp_count
        }

    return result

def print_month_stats(month_stats_dict):
    if not month_stats_dict:
        print("Нет данных для отображения")
        return

    for month, stats in sorted(month_stats_dict.items()):
        print(f"================МЕСЯЦ {month:2d}================")
        print(f"СРЕДНЯЯ:      {stats['avg']:.2f}")
        print(f"МАКСИМАЛЬНАЯ: {stats['max']}")
        print(f"МИНИМАЛЬНАЯ:  {stats['min']}")
        print(f"КОЛИЧЕСТВО:   {stats['count']}")
        print("=====================================")
        print()

def main():
    parser = argparse.ArgumentParser(
        description="Приложение для вычисления статистики температуры из CSV-файла",
        add_help=False
    )

    parser.add_argument(
        "-h",
        "--help",
        action="store_true",
        help="Показать справку по использованию программы"
    )

    parser.add_argument(
        "-f",
        "--file",
        help="Входной CSV-файл",
        required=False
    )
```

```
parser.add_argument(
    "-m",
    "--month",
    type=int,
    help="Номер месяца (1..12) для вывода статистики только по нему",
    required=False
)

try:
    args = parser.parse_args()
except SystemExit:
    print()
    parser.print_help()
    print()
    print("Примеры использования")
    print("  python parser.py -f temperature_big.csv")
    print("  python parser.py -f temperature_big.csv -m 1")
    return

if args.help:
    parser.print_help()
    print()
    print("Примеры использования")
    print("  python parser.py -f temperature_big.csv")
    print("  python parser.py -f temperature_big.csv -m 1")
    return

if args.file is None:
    parser.print_help()
    print()
    print("Примеры использования")
    print("  python parser.py -f temperature_big.csv")
    print("  python parser.py -f temperature_big.csv -m 1")
    return

filename = args.file
month_filter = args.month

if not os.path.exists(filename):
    print(f"ОШИБКА Файл '{filename}' не найден")
    print("Убедитесь что файл находится в той же папке что и скрипт")
    return

if month_filter is not None:
    if not (1 <= month_filter <= 12):
        print("ОШИБКА номер месяца должен быть от 1 до 12")
        return

print(f"Чтение файла {filename}")
line_datas = []

try:
```

```
        with open(filename, 'r', encoding='utf-8') as f:
            line_number = 0
            for line in f:
                line_number += 1
                line = line.strip()

                if line == "":
                    continue

                row = line.split(";")
                if len(row) < 6:
                    print(f"ВНИМАНИЕ Строка {line_number} недостаточно данных -> пропуск")
                    continue

                line_data = parse_line(row, line_number)
                if line_data is not None:
                    line_datas.append(line_data)

        print(f"Успешно обработано строк {len(line_datas)}")

        if not line_datas:
            print("Нет данных для анализа")
            return

        if month_filter is not None:
            monthly_stats = month_stats(line_datas, month_filter)
            if monthly_stats:
                print(f"\nСТАТИСТИКА ЗА МЕСЯЦ {month_filter}")
                print_month_stats(monthly_stats)
        else:
            monthly_stats = month_stats(line_datas)
            if monthly_stats:
                print("\nСТАТИСТИКА ПО МЕСЯЦАМ")
                print_month_stats(monthly_stats)

                year_data = year_stats(line_datas)
                if year_data:
                    print("================ГОД================")
                    print(f"СРЕДНЯЯ:      {year_data['avg']:.2f}")
                    print(f"МАКСИМАЛЬНАЯ: {year_data['max']}")
                    print(f"МИНИМАЛЬНАЯ:  {year_data['min']}")
                    print("===================================")

    except Exception as e:
        print(f"ОШИБКА при чтении файла {e}")
        print("Проверьте правильность формата файла")

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

Кроме того, в рамках работы, необходимо протестировать функционал программы. Для этого, я подготовил ряд сценариев, которые буду сопровождать скриншотами из вывода в консоли:

1) Запуск программы и чтение исходного файла (команда *python parser.py -f temperature_big.csv*)

```
  python parser.py -f temperature_big.csv
  python parser.py -f temperature_big.csv -m 1
PS C:\PISH_Python> python parser.py -f temperature_big.csv
Чтение файла: temperature_big.csv
Успешно обработано строк: 518400

СТАТИСТИКА ПО МЕСЯЦАМ:
================МЕСЯЦ  1================
СРЕДНЯЯ:      15.50
МАКСИМАЛЬНАЯ: 30
МИНИМАЛЬНАЯ:  1
КОЛИЧЕСТВО:   43200
====================================

================МЕСЯЦ  2================
СРЕДНЯЯ:      15.50
МАКСИМАЛЬНАЯ: 30
```

```
223
ПРОБЛЕМЫ  ВЫХОДНЫЕ ДАННЫЕ  КОНСОЛЬ ОТЛАДКИ  ТЕРМИНАЛ  ПОРТЫ

МИНИМАЛЬНАЯ:  1
КОЛИЧЕСТВО:   43200
====================================

================МЕСЯЦ 12================
СРЕДНЯЯ:      15.50
МАКСИМАЛЬНАЯ: 30
МИНИМАЛЬНАЯ:  1
КОЛИЧЕСТВО:   43200
====================================

================ГОД================
СРЕДНЯЯ:      15.50
МАКСИМАЛЬНАЯ: 30
МИНИМАЛЬНАЯ:  1
====================================
PS C:\PISH_Python>
```

В результате, как сказано в задании, мы получили температуру за все месяцы и за целый год, значит, программа в этом плане работает корректно.

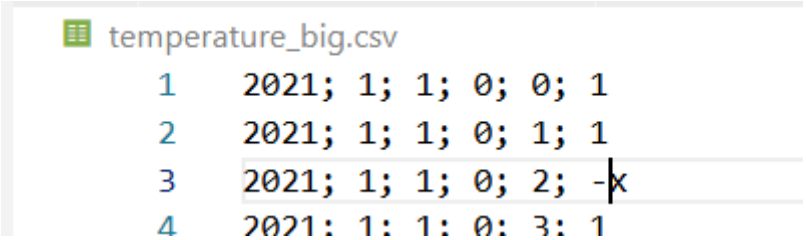
2) Вывод статистики за конкретный месяц (команда *python parser.py -f temperature_big.csv -m 4)*

```
PS C:\PISH_Python> python parser.py -f temperature_big.csv -m 4
Чтение файла: temperature_big.csv
Успешно обработано строк: 518400

СТАТИСТИКА ЗА МЕСЯЦ 4:
================МЕСЯЦ  4================
СРЕДНЯЯ:      15.50
МАКСИМАЛЬНАЯ: 30
МИНИМАЛЬНАЯ:  1
КОЛИЧЕСТВО:   43200
====================================
```

В результате, как сказано в задании, мы получили температуру за конкретный месяц с помощью параметра -m

3) Изменить, например, третью строку в исходном файле и вместо числового значения температуры вписать букву. Для показа работы программы, выполним команду из предыдущего пункта, которая выводит результат за 4-ый месяц.




```
PS C:\PISH_Python> python parser.py -f temperature_big.csv -m 4
Чтение файла: temperature_big.csv
ВНИМАНИЕ: Строка 3: не удалось преобразовать поля в числа -> пропуск. Строка: ['2021', '1', '1', '0', '2', '-x']
Успешно обработано строк: 518399

СТАТИСТИКА ЗА МЕСЯЦ 4:
================МЕСЯЦ  4================
СРЕДНЯЯ:      15.50
МАКСИМАЛЬНАЯ: 30
МИНИМАЛЬНАЯ:  1
КОЛИЧЕСТВО:   43200
====================================
```

Программа отрабатывает корректно, указывает в какой строке допущена ошибка и выводит саму строку для наглядности.

4) Запуск программы без параметров (команда *python parser.py*)

```
PS C:\PISH_Python> python parser.py
usage: parser.py [-h] [-f FILE] [-m MONTH]

Приложение для вычисления статистики температуры из CSV-файла.

options:
  -h, --help         show this help message and exit
  -f, --file FILE    Входной CSV-файл
  -m, --month MONTH  Номер месяца (1..12) для вывода статистики только по нему

Примеры использования:
  python parser.py -f temperature_big.csv
  python parser.py -f temperature big.csv -m 1
```

Программа не выводит ничего кроме окна справки о программе. Она оповещает пользователя о функции программы, выдает ряд параметров, которыми пользователь может воспользоваться, объясняет их и показывает примеры использования.

5) Запуск программы с параметром без аргумента (команда *python parser.py -f*)

```
PS C:\PISH_Python> python parser.py -f
usage: parser.py [-h] [-f FILE] [-m MONTH]
parser.py: error: argument -f/--file: expected one argument

usage: parser.py [-h] [-f FILE] [-m MONTH]

Приложение для вычисления статистики температуры из CSV-файла

options:
  -h, --help         Показать справку по использованию программы
  -f, --file FILE    Входной CSV-файл
  -m, --month MONTH  Номер месяца (1..12) для вывода статистики только по нему

Примеры использования
  python parser.py -f temperature_big.csv
  python parser.py -f temperature_big.csv -m 1
```

Как и требовалось по заданию, пользователь уведомляется ошибкой со стороны argparse, после чего ему выводится справка о программе (--help)

6) Запуск программы с заведомо неверным именем файла

```
PS C:\PISH_Python> python parser.py -f temperature_biggggg.csv
ОШИБКА Файл 'temperature_biggggg.csv' не найден
Убедитесь что файл находится в той же папке что и скрипт
```

Программа сообщает пользователю, что файл с указанным именем не найден и дает подсказку

7) Запуск программы с вводом несуществующего аргумента для параметра -m

```
PS C:\PISH_Python>   python parser.py -f temperature_big.csv -m 22
ОШИБКА номер месяца должен быть от 1 до 12
```

8) Запуск пустого файла

```
PS C:\PISH_Python> python parser.py -f temperature_big1.csv
Чтение файла temperature_big1.csv
Успешно обработано строк 0
Нет данных для анализа
```

В рамках отчета постарался отобразить основные user stories, с которыми должна отрабатывать программа, в самом коде их предусмотрено больше.

По результатам работы, можно сделать вывод, что программа работает корректно и информативно.