ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«УФИМСКИЙ УНИВЕРСИТЕТ НАУКИ И ТЕХНОЛОГИЙ»

Институт информатики, математики и робототехники
Кафедра математического и компьютерного моделирования

Лабораторная работа №5: Работа с форматом CSV в Python

ОБУЧАЮЩЕГОСЯ
4 курса группы ПИ-4ИВТ221Б

Санникова Михаила Александровича

Уровень высшего образования:	высшее образование – бакалавриат
Направление подготовки (специальность)	09.03.03 “Прикладная информатика”
Направленность (профиль) программы	Информационные и вычислительные технологии
Дата выполнения	20.10.2025

УФА – 2025

Постановка задачи:

Создать файл students.csv

Написать функцию для чтения файла

Добавить в файл students.csv:
• Новую запись о студенте 4,Кузнецова Ольга,20,4.2
• Обновите оценку студента с id=3 до 4.1
• Добавьте новый столбец scholarship и проставьте значения True для студентов с оценкой выше 4.0

Написать функцию, которая проверяет:
• Корректность числовых значений (id, age, grade)
• Отсутствие пустых значений в обязательных полях
• Уникальность идентификаторов студентов

Написать функцию, которая возвращает студентов с оценкой выше заданного значения

Практическая часть:

Листинг кода с комментариями

```
import csv
import os
from functools import wraps


def main():
    filename = 'students.csv'

    # Задание 1: Создание CSV-файла
    create_initial_csv(filename)

    # Задание 2: Чтение CSV-файла
    print("=" * 60)
    print("ЧТЕНИЕ CSV-ФАЙЛА")
    print("=" * 60)
    read_csv_file(filename)

    # Задание 3: Модификация CSV-файла
    print("\n" + "=" * 60)
    print("МОДИФИКАЦИЯ CSV-ФАЙЛА")
    print("=" * 60)
    modify_data = modify_csv_file(filename)

    # Задание 4: Валидация данных
    print("\n" + "=" * 60)
    print("ВАЛИДАЦИЯ ДАННЫХ")
    print("=" * 60)
    if is_data_valid(modify_data):
        print("✓ Все данные прошли валидацию успешно!")
    else:
        print("✗ Данные содержат ошибки!")

    # Чтение CSV-файла
    print("=" * 60)
    print("ЧТЕНИЕ CSV-ФАЙЛА")
    print("=" * 60)
    read_csv_file(filename)

    # Задание 5: Фильтрация данных
    print("\n" + "=" * 60)
    print("ФИЛЬТРАЦИЯ ДАННЫХ")
    print("=" * 60)
    filtered_students = filter_students(modify_data, min_grade=4.0)
    print(f"Найдено {len(filtered_students)} студентов с оценкой выше 4.0:")
    for student in filtered_students:
        print(f"  - {student['name']} (оценка: {student['grade']})")


def create_initial_csv(filename):
    """Задание 1: Создание начального CSV-файла"""
    data = [
        ['id', 'name', 'age', 'grade'],
        [1, 'Иванов Иван', 20, 4.5],
        [2, 'Петрова Мария', 19, 4.8],
        [3, 'Сидоров Алексей', 21, 3.9]
```

```
    ]

    with open(filename, 'w', encoding='utf-8', newline='') as file:
        writer = csv.writer(file)
        writer.writerows(data)

    print(f"✓ Создан файл {filename} с начальными данными")


def check_file_exists(func):
    @wraps(func)
    def wrapper(file_path):
        if not os.path.exists(file_path):
            print(f"Не существует файла с заданным путем: {file_path}")
            return None
        return func(file_path)

    return wrapper


@check_file_exists
def read_csv_file(filename):
    """Задание 2: Чтение CSV-файла"""
    try:
        students = []

        with open(filename, 'r', encoding='utf-8') as file:
            reader = csv.DictReader(file)

            print("Заголовки столбцов:", reader.fieldnames)

            total_age = 0
            total_grade = 0
            count = 0

            print("\nВСЕ ЗАПИСИ:")
            print("-" * 40)
            for row in reader:
                # Преобразование типов данных
                row['id'] = int(row['id'])
                row['age'] = int(row['age'])
                row['grade'] = float(row['grade'])
                students.append(row)

                # Вывод информации о студенте
                print(f"ID: {row['id']}, Имя: {row['name']}, "
                      f"Возраст: {row['age']}, Оценка: {row['grade']}")

                # Суммирование для средних значений
                total_age += row['age']
                total_grade += row['grade']
                count += 1

            # Вычисление и вывод средних значений
            if count > 0:
                avg_age = total_age / count
                avg_grade = total_grade / count
```

```
            print(f"\nСредний возраст: {avg_age:.2f}")
            print(f"Средняя оценка: {avg_grade:.2f}")

        return students

    except Exception as e:
        print(f"Ошибка при чтении файла: {e}")
        return None


def modify_csv_file(filename):
    """Задание 3: Модификация CSV-файла"""
    try:
        # Чтение существующих данных
        students = []
        with open(filename, 'r', encoding='utf-8') as file:
            reader = csv.DictReader(file)
            for row in reader:
                # Преобразование типов
                row['id'] = int(row['id'])
                row['age'] = int(row['age'])
                row['grade'] = float(row['grade'])
                students.append(row)

        # Добавление новой записи
        new_student = {
            'id': 4,
            'name': 'Кузнецова Ольга',
            'age': 20,
            'grade': 4.2
        }
        students.append(new_student)
        print("✓ Добавлена новая запись: Кузнецова Ольга")

        # Обновление оценки студента с id=3
        for student in students:
            if student['id'] == 3:
                student['grade'] = 4.1
                print("✓ Обновлена оценка студента Сидоров Алексей до 4.1")

        # Добавление столбца scholarship
        for student in students:
            student['scholarship'] = 'True' if student['grade'] > 4.0 else 'False'
        print("✓ Добавлен столбец scholarship")

        # Запись обновленных данных
        fieldnames = ['id', 'name', 'age', 'grade', 'scholarship']
        with open(filename, 'w', encoding='utf-8', newline='') as file:
            writer = csv.DictWriter(file, fieldnames=fieldnames)
            writer.writeheader()
            writer.writerows(students)

        print("✓ Файл успешно обновлен!")
        return students

    except Exception as e:
        print(f"Ошибка при модификации файла: {e}")
```

```
        return None


def is_data_valid(students):
    """Задание 4: Валидация данных"""
    if not students:
        return False

    errors = []
    student_ids = set()

    for i, student in enumerate(students, 1):
        # Проверка обязательных полей
        required_fields = ['id', 'name', 'age', 'grade']
        for field in required_fields:
            if field not in student:
                errors.append(f"Запись {i}: отсутствует обязательное поле '{field}'")

        # Проверка уникальности идентификаторов
        if 'id' in student:
            if student['id'] in student_ids:
                errors.append(f"Дублирующийся ID: {student['id']}")
            student_ids.add(student['id'])

        # Проверка числовых значений
        if 'id' in student and (not isinstance(student['id'], int) or student['id'] <= 0):
            errors.append(f"ID {student.get('id', 'N/A')}: должен быть положительным целым числом")

        if 'age' in student and (not isinstance(student['age'], int) or student['age'] <= 0):
            errors.append(f"Студент {student.get('name', i)}: возраст должен быть положительным
целым числом")

        if 'grade' in student and (
                not isinstance(student['grade'], (int, float)) or student['grade'] < 0 or student['grade'] > 5):
            errors.append(f"Студент {student.get('name', i)}: оценка {student['grade']} вне диапазона 0-5")

        # Проверка отсутствия пустых значений
        if 'name' in student and not student['name'].strip():
            errors.append(f"Студент с ID {student.get('id', i)}: имя не может быть пустым")

    # Вывод результатов валидации
    if errors:
        print("Найдены ошибки валидации:")
        for error in errors:
            print(f"  ✗ {error}")
        return False
    else:
        return True


def filter_students(students, min_grade=0):
    """Задание 5: Фильтрация данных"""
    filtered = []

    for student in students:
        if student['grade'] > min_grade:
            filtered.append(student)
```

```
    return filtered


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

Скриншоты выполнения программы

```
✓ Создан файл students.csv с начальными данными
============================================================
ЧТЕНИЕ CSV-ФАЙЛА
============================================================
Заголовки столбцов: ['id', 'name', 'age', 'grade']

ВСЕ ЗАПИСИ:
--------------------------------------
ID: 1, Имя: Иванов Иван, Возраст: 20, Оценка: 4.5
ID: 2, Имя: Петрова Мария, Возраст: 19, Оценка: 4.8
ID: 3, Имя: Сидоров Алексей, Возраст: 21, Оценка: 3.9

Средний возраст: 20.00
Средняя оценка: 4.40

============================================================
МОДИФИКАЦИЯ CSV-ФАЙЛА
============================================================
✓ Добавлена новая запись: Кузнецова Ольга
✓ Обновлена оценка студента Сидоров Алексей до 4.1
✓ Добавлен столбец scholarship
✓ Файл успешно обновлен!

============================================================
ВАЛИДАЦИЯ ДАННЫХ
============================================================
✓ Все данные прошли валидацию успешно!
```

```
============================================================
ЧТЕНИЕ CSV-ФАЙЛА
============================================================
Заголовки столбцов: ['id', 'name', 'age', 'grade', 'scholarship']

ВСЕ ЗАПИСИ:
--------------------------------------
ID: 1, Имя: Иванов Иван, Возраст: 20, Оценка: 4.5
ID: 2, Имя: Петрова Мария, Возраст: 19, Оценка: 4.8
ID: 3, Имя: Сидоров Алексей, Возраст: 21, Оценка: 4.1
ID: 4, Имя: Кузнецова Ольга, Возраст: 20, Оценка: 4.2

Средний возраст: 20.00
Средняя оценка: 4.40

============================================================
ФИЛЬТРАЦИЯ ДАННЫХ
============================================================
Найдено 4 студентов с оценкой выше 4.0:
  - Иванов Иван (оценка: 4.5)
  - Петрова Мария (оценка: 4.8)
  - Сидоров Алексей (оценка: 4.1)
  - Кузнецова Ольга (оценка: 4.2)
```