## 🧪 **Лабораторная работа №7: Управление базой данных студентов**

### 🎯 ****Цель:****

Научиться:

* структурировать код в функциях,
* работать со списками и словарями,
* применять сортировку, фильтрацию и поиск,
* организовать взаимодействие с пользователем через меню.

## 📘 **Описание:**

Файл с базой запросить у учителя!

Напиши программу для работы с небольшой «базой данных студентов».  
Каждый студент представлен как словарь с ключами:

{

"name": "Имя",

"age": число,

"grade": средний\_балл

}

Все студенты хранятся в списке students.

## ⚙️ **Функциональные требования:**

Создай функции:

1. **add\_student(students)**  
   — запрашивает у пользователя имя, возраст и средний балл,  
   — добавляет словарь в список студентов.
2. **show\_students(students)**  
   — красиво выводит список всех студентов с нумерацией.  
   Если список пуст — вывести "Нет данных".
3. **find\_best(students)**  
   — находит студента с **наивысшим средним баллом** и выводит его данные.
4. **sort\_by\_grade(students)**  
   — сортирует студентов **по среднему баллу** в порядке убывания и возвращает новый список.
5. **filter\_by\_age(students, min\_age)**  
   — возвращает студентов, чей возраст **старше или равен** заданному.
6. **main()** — основное меню:

1 - Добавить студента

2 - Показать всех студентов

3 - Найти лучшего студента

4 - Сортировать по среднему баллу

5 - Отфильтровать по возрасту

6 - Выйти

## 💡 **Подсказка по коду:**

def add\_student(students):

name = input("Имя: ")

age = int(input("Возраст: "))

grade = float(input("Средний балл: "))

students.append({"name": name, "age": age, "grade": grade})

print(f"Студент {name} добавлен.\n")

def show\_students(students):

if not students:

print("Нет данных.\n")

return

print("\nСписок студентов:")

for i, s in enumerate(students, start=1):

print(f"{i}. {s['name']} — возраст: {s['age']}, балл: {s['grade']}")

print()

def find\_best(students):

if not students:

print("Нет данных.\n")

return

best = max(students, key=lambda s: s['grade'])

print(f"Лучший студент: {best['name']} (средний балл: {best['grade']})\n")

def sort\_by\_grade(students):

if not students:

print("Нет данных.\n")

return

sorted\_students = sorted(students, key=lambda s: s['grade'], reverse=True)

show\_students(sorted\_students)

def filter\_by\_age(students, min\_age):

filtered = [s for s in students if s['age'] >= min\_age]

if not filtered:

print("Нет студентов подходящего возраста.\n")

else:

print(f"\nСтуденты старше {min\_age}:")

for s in filtered:

print(f"{s['name']} — {s['age']} лет, балл: {s['grade']}")

print()

def main():

students = []

while True:

print("Меню:")

print("1 - Добавить студента")

print("2 - Показать всех студентов")

print("3 - Найти лучшего студента")

print("4 - Сортировать по среднему баллу")

print("5 - Отфильтровать по возрасту")

print("6 - Выйти")

choice = input("Выберите пункт: ")

print()

if choice == "1":

add\_student(students)

elif choice == "2":

show\_students(students)

elif choice == "3":

find\_best(students)

elif choice == "4":

sort\_by\_grade(students)

elif choice == "5":

age = int(input("Введите минимальный возраст: "))

filter\_by\_age(students, age)

elif choice == "6":

print("Работа завершена.")

break

else:

print("Ошибка ввода. Повторите попытку.\n")

main()