Практическое задание 1. Сортировки

**Оценка 3:**

• Объявить структуру данных Student, состоящую из полей:

struct Student {

Имя студента (до 64 символов);

Оценка по математике (0-100);

Оценка по физике (0-100);

Оценка по информатике (0-100);

Общий балл (0-300);

};

2) Написать функцию struct Student addStudent(name, math, phy, inf):  
 1) Создаем локальную переменную newStudent,

2) Заполняем поля структуры newStudent входными аргументами

3) В общий балл записываем сумму баллов по всем 3 предметам.

3) Написать функцию void printStudentInfo(struct Student) выводящую всю информацию о студенте через printf()

4) Инициализировать массив из N структур в main: struct Student students[N]; N объявить как константную переменную или через #define

5) Заполнить случайными данными N=100 студентов, используя написанную функцию addStudent()

6) Реализовать сортировку Вставками **или** Шелла **или** Выбором, где отсортировать необходимо массив структур по убыванию по полю «Общий балл».

7) Вывести на экран список студентов до сортировки и после

**Оценка 4:**

8) Ко всему выше сделанному реализовать алгоритм сортировки CountingSort() **или** RadixSort()

9) Вспоминаем ОС:

Вывести на экран модель процессора и его частоту, информация о процессоре лежит в **/proc/cpuinfo**

**Вариант 1**: написать bash скрипт, состоящий из команд вывода на экран инфы о проце и запуска бинарника с сортировками

**Вариант 2**: используя функцию system("linux команда") из stdlib.h

10) Замерить время работы функций сортировки с помощью clock() или gettimeofday() (см. в гугле) для разных N = 100, 10000, 100000.

11) Вывести на экран время работы сортировки и размер данных.