

ЗАДАНИЕ № 1 ПО ПРАКТИКУМУ

для студентов 1 потока 1 курса ф-та ВМК МГУ

Тема: «Методы сортировки».
Языки программирования: Си.

ПОСТАНОВКА ЗАДАЧИ

Реализовать два метода сортировки массива чисел и провести их экспериментальное сравнение. Тип элементов массива, конкретные методы и вид сортировки определяются вариантом задания. Для каждого из реализуемых методов необходимо предусмотреть возможность работы с массивами длины от 1 до N ($N \geq 1$). Значение N в зависимости от варианта задания либо фиксировано, либо память под массив следует выделять динамически (рекомендуется последнее).

При реализации каждого метода вычислить *число сравнений* элементов и *число перемещений* (обменов) элементов.

Сравнение методов сортировки необходимо проводить на одних и тех же исходных массивах, при этом следует рассмотреть массивы разной длины. Для вариантов с фиксированным значением N рассмотреть, как минимум, $n = 10, 20, 50, 100$. Для вариантов с динамическим выделением памяти — $n = 10, 100, 1000, 10000$. Генерация исходных массивов для сортировки реализуется отдельной функцией, создающей в зависимости от заданного параметра и заданной длины конкретный массив, в котором:

- элементы уже упорядочены (1);
- элементы упорядочены в обратном порядке (2);
- расстановка элементов случайна (3, 4).

Результаты экспериментов оформить на основе нескольких запусков программы в виде следующей сводной таблицы.

Название метода сортировки						
n	Параметр	Номер сгенерированного массива				Среднее значение
		1	2	3	4	
10	Сравнения
	Перемещения
20	Сравнения
	Перемещения
⋮						

ВАРИАНТЫ ЗАДАНИЯ

I. Данные (элементы массива).

1. Целые числа (**int**).
2. 64-разрядные целые числа (**long long int**).
3. Вещественные числа двойной точности (**double**).

II. Вид сортировки.

1. Числа упорядочиваются по неубыванию.
2. Числа упорядочиваются по невозрастанию.
3. Числа упорядочиваются по неубыванию модулей, т.е. при сравнении элементов не учитывается знак.

4. Числа упорядочиваются по невозрастанию модулей, т.е. при сравнении элементов не учитывается знак.

III. Методы сортировки.

1. Метод «пузырька» (см. [3] 130-132; [4] 27-28; [5] 101-102).
2. Метод простого выбора (см. [3] 169-171; [4] 15-16; [5] 99-100).
3. Метод Шелла (см. [3] 105-107; [4] 37-40; [5] 105-107).
4. Быстрая сортировка, рекурсивная реализация ([5] 114-117).
5. Пирамидальная сортировка ([2] 178-197).

Таким образом, вариант определяется следующими четырьмя параметрами: (1) переменное или фиксированное число элементов в массиве, (2) тип элементов массива, (3) вид сортировки, (4) два метода сортировки.

ТРЕБОВАНИЯ К ПРОГРАММЕ

1. Каждый из предложенных методов сортировки и генерации массива оформляется в виде отдельной Си-функции. Функции не имеют возвращаемых значений и принимают по два параметра: число n и массив a . Выделение динамической памяти под массив, если предусмотрено вариантом задания, делается до вызова соответствующих функций.
2. Программа должна быть снабжена поясняющими комментариями в объёме, достаточном для её понимания.
3. Программа сдаётся в виде .zip-архива, содержащего файл с исходным кодом, а также отчёт в формате PDF.

ЛИТЕРАТУРА

1. Трифонов Н. П., Пильщиков В. Н. Задания практикума на ЭВМ (1 курс). Методическая разработка (составители). — М.: ВМК МГУ, 2001.
2. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ. Второе издание. — М.: «Вильямс», 2005.
3. Кнут Д. Искусство программирования для ЭВМ. Том 3. — М.: Мир, 1978.
4. Лорин Г. Сортировка и системы сортировки. — М.: Наука, 1983.
5. Вирт Н. Алгоритмы и структуры данных. — М.: Мир, 1989.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Для генерации массива случайных элементов следует использовать функции стандартной библиотеки `rand` и `srand` следующим образом.

- 1) Необходимо подключить заголовочные файлы `stdlib.h` и `time.h`.
- 2) Для инициализации генератора случайных чисел сделать вызов `srand(time(NULL))`.
- 3) Далее, для получения случайных чисел использовать функции `rand()`, возвращающую случайное целое число в диапазоне от 0 до `RAND_MAX`. Как правило, эта константа равна 32767.
- 4) Для формирования элемента массива может понадобиться несколько вызовов `rand()`. Так, для генерации случайного 64-битного целого числа можно использовать выражение вида `rand() * rand() * rand() * rand()`, либо формировать число отдельными байтами.