Продолжение табл. 12.

17	Дана целочисленная прямоугольная матрица размером n*m. Определить номер строки, в которой находится самая длинная серия одинаковых элементов. Пример строки с серией из четырех чисел 3: 1 2 3 3 3 3 5.
18	Дан массив из n элементов целого типа. Преобразовать массив следующим образом, чтобы сначала располагались все элементы равные 0, затем все остальные.

11.2. Практическая работа 2

Тема: «Эмпирический анализ простых алгоритмов сортировки».

Задание 1. Оценить эффективность простого алгоритма сортировки на массиве, заполненном случайными числами (в среднем случае).

1. Составить программу сортировки (функцию) одномерного целочисленного массива A[n], используя алгоритм согласно варианту индивидуального задания - *Алгоритм задания 1* (см. табл. 14). Провести тестирование и отладку программы на исходном массиве из 10 элементов, сформированном значениями с клавиатуры.

2. Провести контрольные прогоны программы для размеров массива n = 100, 1000, 10000, 100000 и 1000000 элементов (сформированные с использованием генератора псевдослучайных чисел):

   a. с вычислением времени выполнения T(n) в мс. Полученные результаты свести в таблицу А по формату табл. 13.

   b. с подсчетом фактического количества операций сравнения $C_ф$ и количества операций перемещения $M_ф$. Полученные результаты – суммы $T_п(n)=C_ф+M_ф$ – вставить в сводную таблицу А.

Таблица 13. Сводная таблица результатов.

n	**T(n), мс**	**$T_п(n)=C_ф+M_ф$**
100		
1000		
10000		
100000		
1000000		

3. Построить график зависимости практической $T_п(n)=(C_ф+M_ф)$ вычислительной сложности алгоритма от размера n массива, оси координат должны быть соответствующим образом подписаны.

4. Определить ёмкостную сложность алгоритма (объём памяти V от n).

5. Проанализировать полученные результаты, сделать соответствующие выводы.

Задание 2. Оценить эффективность алгоритма простой сортировки в случаях строгой упорядоченности по возрастанию и убыванию.

1. Получить формулы для функции роста $T_т(n)$ с подсчетом количества операций сравнения C и количества операций перемещения M для теоретической оценки вычислительной сложности алгоритма сортировки в двух случаях – строгого возрастания и строгого убывания значений в массиве.
2. Провести контрольные прогоны программы на массивах длиной 100, 1000, 10000, 100000 и 1000000 элементов, отсортированных:

А) строго в убывающем порядке значений элементов, результаты представить в сводной таблице B по формату табл. 13 (с добавлением столбца $T_т(n)=C+M$ и соответствующими расчётными значениями);

Б) строго в возрастающем порядке значений элементов, результаты представить в сводной таблице C по формату табл. 13 (с добавлением столбца $T_т(n)=C+M$ и соответствующими расчётными значениями);

3. Сравнить результаты этих случаев с результатами задания 1. Построить график зависимости практической вычислительной сложности алгоритма T(n) от размера n массива во всех трёх рассмотренных случаях (случайное заполнение, строгое убывание, строгое возрастание).
4. Сделать вывод о зависимости (или независимости) алгоритма сортировки от исходной упорядоченности массива.

Задание 3. Сравнить эффективность двух алгоритмов простых сортировок

1. Выполнить разработку и программную реализацию *Алгоритма задания 3* вашего индивидуального варианта (см. табл. 14). Провести тестирование и отладку программы на исходном массиве из 10 элементов, сформированном значениями с клавиатуры.

2. Сформировать таблицы с результатами прогонов программы в соответствии с форматом табл. 13 на тех же массивах, что и в задании 2.

3. На основе анализа кода алгоритма определить формулы $T_т(n)$ функций роста в лучшем и худшем случае.
4. Добавьте в таблицы для лучшего и худшего случаев столбец $T_т(n)=C+M$ и заполните соответствующими расчётными значениями.
5. Определить ёмкостную сложность алгоритма (объём памяти V от n), сравнить её с ёмкостной сложностью первого алгоритма.

6. Построить сравнительные графики $T_п(n)$: один для случая строгого возрастания и второй – для строгого убывания значений в массиве.

7. Сделать итоговый вывод о вычислительной сложности алгоритмов.

Таблица 14. Варианты индивидуальных заданий.

№	Алгоритм задания 1	Алгоритм задания 3
1	Простой вставки (***Insertion sort***)	Простого выбора ***(Selection sort)***
2	Простого обмена (***Exchange sort***)	Простой вставки (***Insertion sort***)
3	Простого выбора ***(Selection sort)***	Простого обмена (***Exchange sort***)

Отчёт:

В отчёте по каждой сортировке необходимо привести словесное описание алгоритма и его блок-схему, а также программный код (с комментариями), результаты тестирования на массиве n=10 и контрольных прогонов на массивах длиной 100, 1000, 10000, 100000 и 1000000 элементов.

По каждому пункту в заданиях 1-3 приведите полученные результаты в виде таблиц A, B и C и графиков.

По итогам каждого задания сформулируйте соответствующие выводы.

В выводах по всей работе опишите, какие знания, умения и практические навыки получены в ходе выполнения практической работы.

11.3. Практическая работа 3

Тема: «Асимптотический анализ эффективности на примерах алгоритмов сортировки».

Задание 1. Эмпирическая оценка эффективности алгоритмов в среднем случае.

Требования по выполнению задания

1. Разработать алгоритм ускоренной сортировки, определенной в варианте (табл. 17), реализовать код на языке C++. Сформировать таблицу 1.1 результатов эмпирической оценки сложности сортировки по формату табл. 15 для массива, заполненного случайными числами. Определить ёмкостную сложность алгоритма.

Таблица 15. Сводная таблица результатов

n	T(n), мс	$T_п(n)=C_ф+M_ф$
100		
1000		
10000		
100000		
1000000		

1. Разработать алгоритм быстрой сортировки, определенной в варианте (приложение 1), реализовать код на языке C++. Сформировать таблицу 2.1 результатов эмпирической оценки сортировки по формату табл. 15 для массива, заполненного случайными числами. Определить ёмкостную сложность алгоритма.