Лабораторная работа № 1 по курсу дискретного анализа: сортировка за линейное время

Выполнил студент группы М8о-207-18 МАИ *Тояков Артем.*

Условие

1. Требуется разработать программу, осуществляющую ввод пар «ключ-значение», их упорядочивание по возрастанию ключа указанным алгоритмом сортировки за линейное время и вывод отсортированной последовательности.

2. Сортировка подсчётом. Тип ключа: числа от 0 до 65535. Тип значения: строки фиксированной длины 64 символа, во входных данных могут встретиться строки меньшей длины, при этом строка дополняется до 64-х нулевыми символами, которые не выводятся на экран.

Метод решения

1. Считываю ключ и значение в формате моего варианта.

2. Записываю их в динамический массив, выделяя перед этим память для новой ячейки(если это требуется) и очередного значения.

3. Вызываю функцию сортировки подсчётом для заполненного динамического массива.

4. Печатаю результат, параллельно освобождая память для значений.

5. Освобождаю память моего массива.

Описание программы

Я создал структуру elements, в которой хранил мою пару ключ-значение, где ключ был числом типа int, а значение хранилось в динамическом массиве типа char*.

MAIN: В самом начале функции main я отключаю синхронизацию iostreams с stdio для более быстрой работы моей программы. Затем я создаю динамический массив типа elements под именем table размером 2, который будет хранить мои пары и выделяю в нём память под два элемента, а также я создаю 3 дополнительные переменные. Size будет отвечать за размер моего массива. С помощью перемнной Iterator я буду обращаться к элементам моего массива, а maxsize будет давать мне понять, когда пора увеличить размер table с помощью функции resize. Далее я выделяю память под value первого элемента и запускаю цикл while, в котором я буду вводить мои элементы и сразу вносить их в массив. Внутри цикла изменяются переменные size и iterator, а также проводится проверка для функции resize. В самом конце цикла выделяется память под значение

следующего элемента. После выхода из цикла освобождается память из-под значения последнего (невведённого) элемента и вызывается функция counting sort. Затем я печатаю отсортированную таблицу и освобождаю память всех значений. В самом конце я освобождаю память из-под моего массива.

RESIZE: Функция принимает на вход текую таблицу table и её размер size. В этой функции создаётся новый массив, размер которого в 2 раза больше полученного, и все значения table переприсваиваются в новый массив temp. Освобождается память старого массива. Функция возвращает новый увеличенный массив temp.

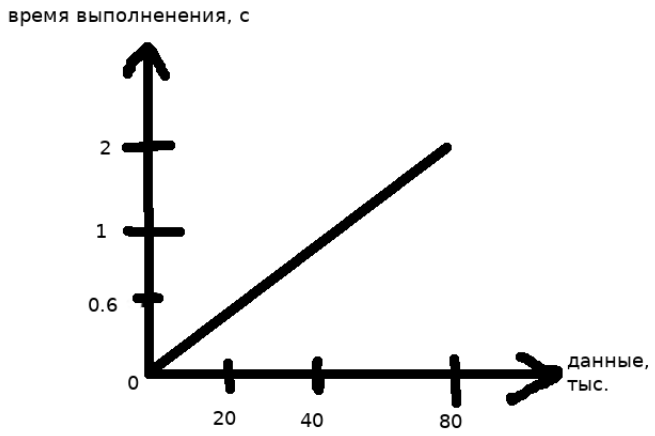
MAXIMUM: На вход подаётся таблица и её размер. Функция будет возвращать максимальное значение max, которое изначально равно 0. Проходя по всему массиву оно будет при сравнении его со всеми элементами и в итоге станет максимальным.

COUNTING SORT: На вход функции подаётся таблица table, её размер size и результат функции maximum max. Затем я создаю новый динамический массив sort table размера size и ещё один массив count table размера max + 1. Далее будет 4 цикла. В первом я прохожу по значениям от 0 до max и заполняю весь count table нулями. Во втором я прохожу по всем индексам от 0 до size - 1 и считаю количеством чисел, которые есть в исходной table и записываю его в count table. В третьем цикле я прохожу от 1 до max и считаю сколько чисел меньше или равно до каждого числа в count table. В четвёртом цикле я прохожу от size - 1 до 0 и присваиваю в sort table[–count table[table[i]]] значения table. В конце я освбождаю память table и count table и возвращаю sort table.

Дневник отладки

При первой отправке возникла ошибка компиляции. Было выведено предупреждение, говорившее о том, что ввод значений в элемент массива был осуществлён некорректно. Я поместил его в условие цикла while, после чего программа скомпилировалась успешно, потому что компилятор правильно воспринимал значение, которое возвращал scanf.

Тест производительности



Исходя из данных графика можно понять что сложность алгоритма сортировки подсчётом o(n), как и было заявлно ранее.

Недочёты

Вместо vector я использовал динамический массив, так как не смог полностью понять его принцип работы. Это не очень эффективно, потому что у вектора есть набор методов, легко обеспечивающих взаимодействие с его элементами, и если бы мне понадобилось как-то иначе взаимодействовать с ключами и значениями (например удалять их), то это было бы довольно проблематично при использовании обычного динамического массива.

Выводы

Алгоритм сортировки подсчётом целесообразно применять, когда нужно отсортировать большое количество чисел в маленьком диапазоне (например миллион чисел от 0 до 1000). Соответственно он будет неэффективен, если в исходном массиве есть очень большие числа.

Этим алгоритмом можно эффективно отсортировать например зарплаты сотрудников.

В целом данная сортировка программируется очень простым образом и интуитивно понятна и никаких сложностей при реализации я не встреитил. Эти свойства являются большим плюсом сортировки подсчётом.