Лабораторная работа №3

Записи с вариантами, обработка таблиц

Цель работы: приобрести навыки работы с типом данных «запись», содержащим вариантную часть, и с данными, хранящимися в таблицах. Сравнить несколько различных алгоритмов сортировок массива при использовании таблиц записей с большим числом полей и таблиц ключей. Оценить относительную эффективность программы (в процентах) по времени и по используемому объему памяти

Задание (Вариант 2):

Создать таблицу, содержащую не менее 40-ка записей (тип – запись с вариантами). Упорядочить данные в ней по возрастанию ключей, где ключ – любое невариантное поле (по выбору программиста — выбрано поле Население), используя: а) саму таблицу, б) массив ключей

Ввести список стран, содержащий название страны, количество жителей, столицу, материк, основной вид туризма (экскурсионный - количество объектов, основной вид (природа, история, искусство); пляжный – основной сезон, температура воздуха и воды, время полета до страны; спортивный – вид спорта (горные лыжи, серфинг, восхождения), минимальная стоимость отдыха.. Вывести список стран, где можно покататься на горных лыжах.

Требования к входным данным:

На вход программы подаются данные трех типов.

1. Значение поля является словом. Слова — строки, состоящие только из букв (латиница). Длина слова не должна превышать `LEN_STRING`=50. Одно поле подразумевает запись только одного слова.
2. Значение поля является числом. В общем случае это целое число, не выходящее из диапазона [-2,147,483,647, 2,147,483,647]. Учитывая, что население страны не может быть отрицательным и нулевым (как и минимальная стоимость отдыха, кол-во объектов и т. д.), то в таких случаях вводятся дополнительные ограничения. Также в случае задание температуры разрежено вводить числа с плавающей точкой в диапазоне [-70, 70] (диапазон выбран исходя из возможно диапазона температур)
3. Выбор определенного поля, указанного в подсказке ввода. Нужно ввести выбранное число. Кроме указанных в подсказке, никакие другие числа недопустимы.

В случае ввода пустой строки, программа будет ожидать введения хотя-бы одного символа.

Интерфейс программы:

```
************************************************************
Выберите одно из следующих действий:
0: Показать таблицу
1: Отсортировать таблицу по ключам (не меняя исходную)
2: Отсортировать таблицу
3: Добавить запись
4: Удалить запись
5: Сравнить эффективность сортировки таблицы и ключей
6: Сравнить эффективность сортировок (для таблиц по ключам)
7: Вывести список горнолыжных курортов
8: Закончить работу
```

0. Показать все элементы таблицы. Пример вывода одной из записей:

```
Страна Ukraine
НАСЕЛЕНИЕ 500
Столица Kiev
Материк Antarctica
Тип туризма: спортивный
Стоимость отдыха: 45
Вид: skiing
```

1. Сортировка массива по ключам. В выводе продемонстрирован изначальный массив индексов таблицы и отсортированный. Далее выведены элементы таблицы в отсортированном порядке. Поле для сортировки: НАСЕЛЕНИЕ
2. Сортировка таблицы и ее вывод.
3. Добавление нового элемента. (зеленым отображен ввод консоли).

```
Введите название страны: Russia
Введите количество жителей: 20000
Введите столицу: Moscow
Выберите материк:
0-Africa, 1-Antarctica, 2-Asia, 3-Australia, 4-Europe, 5-North_America, 6-South_America: 4
Выберите вид туризма: 0 - экскурсионный 1 - пляжный 2 - спортивный : 2
Введите минимальную стоимость отдыха: 100
Выберите основной вид:
0 - горные лыжи, 1 - серфинг, 2 - восхождения : 0
```

4. Удаление эл-та. Для удаления необходимо указать индекс удаляемой записи. В случае выхода из диапазона индексов, программа напечатает сообщение об ошибке. В случае успешного удаления, программа напечатает соответствующее сообщение.
5. Программа печатает время, требуемое для сортировки таблицы ключей и исходной таблицы (алгоритм быстрой сортировки).
6. Программа сравнивает алгоритм сортировки пузырьком и быстрой сортировки для таблицы ключей.
7. Вывод всех стран с горнолыжными курортами.
8. Выход из программы

Обращение к программе:

Через консоль

Внутренние структура данных:

```
enum continents { Africa, Antarctica, Asia, Australia, Europe, North_America,
South_America } ;
enum kind_excursion { nature, history, art } ;
enum kind_sport { skiing, surfing, climbing } ;
enum union_tourism { excursion, sport, beach } ;
enum seasons { winter, spring, summer, autumn } ;
struct excursion_tourism {
    int obj_count;
    kind_excursion kind;
    void create_excursion();
    void show();
};
struct beach_tourism {
    float temp_water;
    float temp_air;
    int time_fly;
    seasons season;
    void create_beach();
    void show();
};
struct sport_tourism {
    kind_sport kind;
    int min_cost;
    void create_sport();
    void show();
};
union kind_tourism {
    excursion_tourism excursion;
```

```
    beach_tourism beach;
    sport_tourism sport;
};
class MyRecord {
private:
    char country[LEN_STRING];
    long int population;
    char capital[LEN_STRING];
    continents cont;
    union_tourism kind;
    kind_tourism tourism;

}
```

Структура данных таблицы ключей:

```
class Table_key {

public:
    int id;
    long int population;
    Table_key();
    Table_key(int i, MyRecord &obj);
};
```

Сравнение эффективности сортировок (на таблице ключей):

Кол-во эл-тов	Время работы сортировки пузырьком, мс	Время работы быстрой сортировки, мс
20	2	2
50	10	6
100	47	19
150	59	21
200	113	25

Сравнение эффективности сортировки по таблице и таблице ключей (быстрая сортировка):

Кол-во эл-тов	Время работы для таблицы ключей, мс	Время работы для таблицы записей, мс
20	4	5
50	5	8
100	13	29
150	17	32
200	30	63

Оценка памяти:

Для хранения таблицы ключей дополнительно требуется хранение 8 байт для каждой

переменной (значение индекса + значение поля).

Таблице ключей занимает 8 + 8 = 16 байт

Эл-т исходной таблицы занимает 132 байта (на 725 % больше)

Вывод:

В процессе тестирования было проведено сравнение сортировок таблицы записей двумя способами: сортировка непосредственно таблицы и сортировка отдельной таблицы, содержащей ключ — поле сортировки и индекс эл-та в таблице записей. К недостаткам последнего способа можно отнести использование дополнительного кол-ва памяти (+8 байт на каждый эл-т). К преимуществам: существенное уменьшение времени работы за счет экономии времени операции перезаписи всех полей структуры.

Также было проведено сравнение двух видов сортировок: сортировки пузырьком и быстрой сортировки. Полученные данные подтверждают алгоритмическую сложность выполнения этих алгоритмов («пузырек» - O(n^2), быстрая — O(NlogN)).

Для хранения информации о различных видах туризма была использована вариативная запись. Ее преимущества относительно обычной структуры заключается в использовании меньшего кол-ва памяти. Объем памяти, необходимый для записи с вариантами складывается из длин полей фиксированной части и максимального по длине поля вариантной части. К недостаткам можно отнести необходимость отслеживания правильности хранения и обработки данных.

Теоретическая часть:

1. Как выделяется память под вариантную часть записи?

   Объем памяти, необходимый для записи с вариантами складывается из длин полей фиксированной части и максимального по длине поля вариантной части.

2. Что будет, если в вариантную часть ввести данные, несоответствующие описанным?

   Тип данных в вариантной части при компиляции не проверяется, поэтому, контроль за правильностью ее использования возлагается на программиста. Поведение программы в этом случае непредсказуемо.

3. Кто должен следить за правильностью выполнения операций с вариантной частью записи?

   Программист.

4. Что представляет собой таблица ключей, зачем она нужна?

   Таблица ключей представляет из себя массив структур, полями которых является индекс записи в исходной таблице записей и поле-ключ, по которому можно вести поиск, сортировку и другую обработку записей. Она позволяет сократить время, которое используется для перестановки эл-тов таблицы, а также сохранить исходную таблицу в

начальном состоянии (если необходимо провести несколько сортировок по разным ключам).

5. В каких случаях эффективнее обрабатывать данные в самой таблице, а когда – использовать таблицу ключей?

   Если каждая запись таблицы состоит из небольшого кол-ва полей, то выгодней обрабатывать данные в таблице. В других случаях целесообразней использовать таблицу ключей.

6. Какие способы сортировки предпочтительнее для обработки таблиц и почему?

   Если обработка данных производится непосредственно в таблице, то эффективнее использовать алгоритмы сортировки, которые предполагают наименьшее кол-во операций перестановки — перезаписи переменных (алгоритм сортировки вставками целесообразнее использовать по сравнению с сортировкой пузырьком). Если же сортировка производится по таблице ключей, эффективнее использовать сортировки с наименьшей сложностью работы (быстрая сортировка, сортировка слиянием, пирамидальная сортировка).