**Типы и Структуры Данных**

**Oтчет**

## *Работа № 2: Записи с вариантами. Обработка таблиц*

***Студент: Нгуен Фыок Санг***

***Группa: ИУ7И-36Б***

*2019*

1. **Oписание условия задачи:**

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), используя:

а) саму таблицу

б) массив ключей

(возможность добавления и удаления записей в ручном режиме обязательна)

Ввести список машин, имеющихся в автомагазине, содержащий: марку автомобиля, страну-производитель, цену, цвет и состояние: новый – гарантия (в годах); нет - год выпуска, пробег, количество ремонтов, количество собственников. Вывести цены не новых машин указанной марки с одним предыдущим собственником, отсутствием ремонта в указанном диапазоне цен.

**II. Oписание ТЗ:**

1. **Исходные данные**: таблица (массив записей) из файла.
2. **Результат**: цены не новых машин указанной марки с одним предыдущим собственником, отсутствием ремонта в указанном диапазоне цен.
3. **Задачи, реализуемой программой**:
   * Вывести цены не новых машин указанной марки с одним предыдущим собственником, отсутствием ремонта в указанном диапазоне цен.
   * Упорядочить данные в ней по возрастанию ключей.

**III. Описание СД:**

1. **Для ввода:** 
   * Использовать массив записей размером 100.
2. **Для обработки:**
   * Использовать матрицу размера 100 \* 3 для сохранения ключевого поля.
3. **Для результатa:**
   * Использовать массив записей размером 100.

***Структура записей:***

struct old\_car // Информация о не новых автомобилях

{

int year;

int miles;

int num\_repairs;

int num\_owners;

};

union car\_status // автомобиль новый или не новый

//вариантная часть записи

{

int guarantee;

struct old\_car detail;

};

struct car\_infor // Информация автомобилях

{

int id;

char mark[50];

char country[50];

int price;

char color[10];

int new;

union car\_status status;

};

**IV. Oписание алгоритма:**

1. ***Удалить элемент из массива***
   * Двигать элементы вперед
   * Освобождается последний элемент
2. ***Вставить элемент в массив***
   * Выделяется память для последнего элемента (n+1)
   * Назначить значение
3. ***быстрая сортировка на отрезке:***
   * Выбрать элемент из массива. Назовём его опорным.
   * *Разбиение*: перераспределение элементов в массиве таким образом, что элементы меньше опорного помещаются перед ним, а больше или равные после.
   * Рекурсивно применить первые два шага к двум подмассивам слева и справа от опорного элемента. Рекурсия не применяется к массиву, в котором только один элемент или отсутствуют элементы.

**V. Набор тестов:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N (number  of cars) | Quick\_sort | | Bubble\_sort | |
| key\_array | table | key\_array | table |
| 69600 | 7 | 21 | 5788 | 30032 |
| 36540 | 4 | 12 | 1564 | 4755 |
| 12180 | 1 | 4 | 171 | 261 |
| 5220 | 1 | 1 | 32 | 47 |

**VI. Ответы на вопросы:**

1. ***Как выделяется память под вариантную часть записи?*** 
   * Выделяется область памяти под вариантную часть записи, равная размеру максимального поля вариантных частей.
2. ***Что будет, если в вариантную часть ввести данные, несоответствующие описанным?*** 
   * Данные не проверены.
3. ***Кто должен следить за правильностью выполнения операций с вариантной частью записи?*** 
   * Программист.
4. ***Что представляет собой таблица ключей, зачем она нужна?*** 
   * Таблицa ключей, содержит индекс элемента в исходной таблице и выбранный ключ.
   * Можно уменьшить время обработки (ускорить поиск и сортировку исходной таблицы).
5. ***В каких случаях эффективнее обрабатывать данные в самой таблице, а когда – использовать таблицу ключей?*** 
   * Если в таблице много полей, для ускорения сортировки, рекомендуется использовать **массив ключей**.
   * Если в таблице мало полей (2, 3), для оптимальной памяти, рекомендуется использовать **саму таблицу.**
6. ***Какие способы сортировки предпочтительнее для обработки таблиц и почему?***
   * В общем случае рекомендуется использовать быструю сортировку по оптимальному алгоритму. Однако это зависит от входных данных.