

Министерство образования и науки РФ

НГТУ

Кафедра Автоматики

Курсовая работа

Программирование на языке Си

Таблица: 6
Вариант: 24

Преподаватель:
Голодных Геннадий
Петрович

Факультет: АВТФ
Группа: АП-819
Студент: Зубарев Василий

Дата: 10.05.09

Новосибирск 2009

Задание:

Написать программу, выполняющую действия с базой данных по вариантам заданий из таблицы 6. База данных должна быть организована в виде массива структур. Память под массив структур выделяется статически. Количество записей ограничено 200. Программа должна обеспечивать следующий набор операций:

- 0) ввод записи с произвольным номером;
- 1) вывод записи с заданным номером;
- 2) сортировка записей по заданному полю в порядке убывания или возрастания;
- 3) вывод всех записей в отсортированном порядке на экран или принтер;
- 4) чтение записей из файла;
- 5) сохранение всех записей в файле.

Для сортировки БД использовать массив указателей. Ввод и вывод данных должен сопровождаться необходимыми подсказками.

Поля базы данных:

С – название пункта, расстояние, количество рейсов

Таблица:

№	Поля записи БД (элементы структуры)	Кол-во записей в БД	№ поля для сортировки	Выделение памяти под структуры	Тип сортировки	Устройство для вывода записей
24	С	200	2	Статическое	Убывает	Экран

Листинг программы:

```
#include <stdio.h>
#include <string.h>
#define MAX 200

struct AERO {
    char name[30];
    int dist;
    int flights;
};

int db_sort (struct AERO **db) {
    // Field - dist
    int i = 0, j;
    struct AERO *buf;
    for (i = 0; i < MAX; i++) {
        for (j = i + 1; j < MAX; j++) {
            if (db[i]->dist < db[j]->dist) {
                buf = db[i];
                db[i] = db[j];
                db[j] = buf;
            }
        }
    }
    return 0;
};
```

Подключение стандартной библиотеки ввода вывода и библиотеки работы со строками

Описание структуры. Name — имя города, dist — расстояние, flights — количество рейсов

Функция сортировки массива указателей.

```

int db_load(struct AERO *db) {
    int i = 0;

    FILE * db_file;
    db_file = fopen ("input.txt", "r");
    if (db_file != NULL) {
        while (!feof(db_file)) {
            fscanf(db_file, "%s %d %d",
                &db[i].name,
                &db[i].dist,
                &db[i].flights);
            i++;
        }
        fclose (db_file);
    }
    return 0;
};

```

Функция загрузки БД из файла input.txt

```

int db_save(struct AERO **db) {
    int i = 0;

    FILE * db_file;
    db_file = fopen ("output.txt", "w");
    if (db_file != NULL) {
        while (i < MAX) {
            if (db[i]->dist != 0) {
                fprintf(db_file, "%s %d %d\n",
                    db[i]->name,
                    db[i]->flights,
                    db[i]->dist);
            }
            i++;
        }
        fclose (db_file);
        return 0;
    } else {
        return -1;
    }
};

```

Функция сохранения результатов жизнедеятельности в файл output.txt

```

int db_select(int id, struct AERO **db) {
    if (db[id]->dist != 0) {
        printf("%s %d %d\n",
            db[id]->name,
            db[id]->flights,
            db[id]->dist);
    }
    return 0;
};

```

Функция выбора записи из БД по ее ID. MySQL отдыхает.

```

int main(int argc, char** argv) {
    struct AERO db[MAX], *sort[MAX];
    char ch;
    int id, i;

    for (i = 0; i < MAX; i++) {
        db[i].dist = db[i].flights = 0;
        strcpy(db[i].name, "");
        sort[i] = &db[i];
    }

    printf( "0 - Insert row\n
            1 - Select row\n
            2 - Sort by distance\n
            3 - Print database\n
            4 - Load from file\n
            5 - Save to file\n
            >>");

    while (ch = getchar()) {
        switch (ch) {
            case '0':
                printf("Enter ID: ");
                scanf("%d", &id);
                printf("Enter fields (name, distance,
flights): ");
                scanf("%s %d %d", &db[id].name,
&db[id].dist, &db[id].flights);
                break;
            case '1':
                printf("Enter ID: ");
                scanf("%d", &id);
                db_select(id, sort);
                break;
            case '2':
                printf("Sorting...\n");
                db_sort(sort);
                break;
            case '3':
                printf("Printing all rows...\n");
                for (i = 0; i < MAX; i++) {
                    db_select(i, sort);
                }
                break;
            case '4':
                printf("Loading from file...\n");
                db_load(db);
                break;
            case '5':

```

Начинается функция main(), в которой и происходит самое интересное.

В этом месте я не придумал ничего лучше, чем сделать вот так :(

```
        printf("Saving to file...\n");
        db_save(sort);
        break;
    }
    printf(">");
}

return 0;
}
```

Пример вывода:

```
0 - Insert row
1 - Select row
2 - Sort by distance
3 - Print database
4 - Load from file
5 - Save to file
```

```
>>4
Loading from file...
>>3
Printing all rows...
Novosibirsk 20 3300
Vladivistok 10 7000
Krasnoyarsk 30 4000
Saratov 15 2000
Piter 10 300
>>2
Sorting...
>>3
Printing all rows...
Vladivistok 10 7000
Krasnoyarsk 30 4000
Novosibirsk 20 3300
Saratov 15 2000
Piter 10 300
>>5
Saving to file...
>>
```

Вывод: В ходе курсовой работы мы научились основным возможностям языка программирования СИ, таким как структуры, указатели, функции, массивы, etc. Написали свою базу данных с возможностью ввода, вывода данных с экрана или из файла, и сортировки их по указанному полю. Так же научились выделять память и писать свою сортировку.