

Тема : знакомство с С (Си).

Содержание: **упражнение 1, упражнение 2, доп. упражнение**

Упражнение 1:

- Написать функцию: `void sortMull (student *st, int size);` которая сортирует массив структур `student`, используя пузырьковую сортировку. Массив структур читается из файла `“data.txt”` (смотрите предыдущий пример) и сортируется по фамилиям (в алфавитном порядке, по возрастанию) . Упорядоченные записи выводятся на экран.

Данные файла “data.txt”:

Ivanovich 22 2.9
Sidorov 21 3.5
Petrov 20 4.2
Ivanova 20 4.1
Baranov 23 3.3
Isakov 21 3.9
Kolobkov 19 4.6
Volodin 25 4.7
Dubov 24 3.1
Ivanov 18 4.0

Sorted data:

Baranov 23 3.3
Dubov 24 3.1
Isakov 21 3.9
Ivanov 18 4.0
Ivanova 20 4.1
Ivanovich 22 2.9
Kolobkov 19 4.6
Petrov 20 4.2
Sidorov 21 3.5
Volodin 25 4.7

- Для сравнения строк символов `char []` использовать строковую функцию:
`int strcmp(const char *s1, const char *s2);`
Для использованию этой функции не забудьте добавить в программу:
`#include <string.h>`
- Пример копирования структур:

```
student st1, temp;  
st1.name = "Ivanov";  
st1.age = 20;  
st1.mark = 4.0;.  
temp = st1; // prostoe prisvaivanie  
//ili ze kopiruem po otdelnosti vse elementy struktury  
temp.age = st1.age;  
temp.mark = st1.mark;  
// zdesj kopiruem stroku simvolov, a ne ukazatelj, t.e. nelzja  
ispolzovatj temp.name = st1.name  
strcpy(temp.name, st1.name);
```

Решение:

```
#include <stdio.h>
#include <stdlib.h> // stdlib.h - необходима для использования функций
// динамического выделения памяти (realloc, malloc, calloc)
```

```
typedef struct stud{
    char name[21]; // фамилия
    int age; // возраст
    double mark; // средняя оценка
} student;
```

```
void sortingMull(student * st, int n);
```

```
int main(void)
{
    student *st;
    FILE *fData;
    int size = 0;
    char fname[] = "data.txt"; // название файла

    if((fData = fopen(fname, "r" )) == NULL){
        printf( "Ошибка при открытии файла %s\n", fname);
        exit(1);
    }

    //("st" должна быть инициализирована перед вызовом функции "realloc")
    st = NULL; // инициализация

    while(!feof(fData)){
        // динамически увеличивает выделенную память на 1 запись
        if((st = realloc(st, (size+1)*sizeof(student))) == NULL){
            printf("Ошибка при распределении памяти\n");
            exit(1);
        }
        fscanf(fData, "%s %d %lf", st[size].name, &st[size].age,
            &st[size].mark); // читаем запись из файла
        // выводим запись на экран, st[size].name ~ (st+size)->name
        printf("%s %d %.1f\n", (st+size)->name, (st+size)->age,
            (st+size)->mark);
        size++; // количество прочитанных записей
    }
    fclose(fData); // закрытие файла
```

```
    sortingMull(st, size); // сортировка элементов массива
    printf("\nSorted data:\n");
    for(i=0; i<size; i++){ // вывод отсортированного массива на экран
        printf("%s %d %.1f\n", (st+i)->name, (st+i)->age, (st+i)->mark);
    }
```

```

    //osvobozenie pamjati
    free(st);
    getchar();
}

// sortirovka elementov massiva po familii studenta
void sortingMull(student * st, int n){
    int i, j;
    student temp;

    for(i=1; i<n; i++){
        for(j=0; j<n-i; j++){
            if(strcmp(st[j].name, st[j+1].name) > 0){
                strcpy(temp.name, st[j].name); // temp = st[j];
                temp.age = st[j].age;
                temp.mark = st[j].mark;

                strcpy(st[j].name, st[j+1].name); // st[j] = st[j+1];
                st[j].age = st[j+1].age;
                st[j].mark = st[j+1].mark;

                strcpy(st[j+1].name, temp.name); // st[j+1] = temp;
                st[j+1].age = temp.age;
                st[j+1].mark = temp.mark;
            }
        }
    }
}

```

Упражнение 2:

- Написать функцию: `void sortShell (student *st, int size);` которая сортирует массив структур `student`, используя сортировку методом Шелла. Массив структур читается из файла “data.txt” (смотрите предыдущий пример) и сортируется по возрасту (по убыванию) . Упорядоченные записи выводятся на экран.

```

void sortingShell(student * st, int n){
    int gap, i, j;
    student temp;
    for(gap = n/2; gap > 0; gap /= 2)
        for (i = gap; i < n; i++)
            for (j = i - gap; j >= 0 && st[j].age < st[j+gap].age; j -= gap) {
                temp = st[j];
                st[j] = st[j + gap]
                st[j + gap] = temp;
            }
}

```

Доп. упражнение (не обязательно) :

- Выполнить упражнение 1 и 2, сортирую не сами элементы массива, а индексы массива. Используется для избежание трудоемкого копирования массивных структур при сортировке.

Пример:

не отсортированный массив

Index (индекс)	0	1	2	3
Value (значение)	8	3	1	5

отсортированный массив

Index (индекс)	3	1	0	2
Value (значение)	8	3	1	5

```
int main(void)
{
    ...
    int *isorted; //массив индексов
    ...
    if((isorted = calloc(size, sizeof(int))) == NULL){ // выделение памяти
        printf("Ошибка при распределении памяти\n");
        exit(1);
    }

    for(i=0; i<size; i++) *(isorted + i) = i; // инициализация массива
                                //индексов, можно просто isorted[i] = i
    sortingMullByIndex(isorted, st, size); // сортировка индексов массива
    printf("\nSorted data by index:\n");
    for(i=0; i<size; i++){ // вывод отсортированного массива на экран
        printf("%s %d %.1f\n",st[isorted[i]].name,st[isorted[i]].age,
            st[isorted[i]].mark);
    }
    ...
}

// sortirovka elementov massiva po familii studenta
void sortingMullByIndex(int isorted[], student * st, int n){
    int i, j, temp;
    for(i=1; i<n; i++){
        for(j=0; j<n-i; j++){
            if(strcmp(st[isorted[j]].name, st[isorted[j+1]].name) > 0){
                temp = isorted[j];
                isorted[j] = isorted[j+1];
                isorted[j+1] = temp;
            }
        }
    }
}
```

}