Отчет по курсовому проекту N 6 по курсу

"Фундаментальная информатика"

Студент группы: М80-107Б-21, Павлов Иван Дмитриевич
Контакты: pavlov.id.2003@gmail.com
Работа выполнена: 08.04.2022
Преподаватель: Найденов Иван Евгеньевич

1 Тема

Обработка последовательностей файловой структуры на языке Си

2 Цель работы

Разработать последовательную структуру данных для представления простейшей базы данных на файлах в СП Си.

3 Задание

№4. Отпечатать список студентов, компьютеры которых нуждаются в апгрейде (Сведения о составе комплектующих личных ПЭВМ в студенческой группе: фамилия владельца, число и тип процессоров, объем памяти, тип видеоконтроллера, объем видеопамяти, чип, число и емкость винчестеров, количество интегрируемых контроллеров и внешних (переферийных) устройств, операционная система).

4 Оборудование

Процессор: AMD Ryzen 5 4600H with Radeon Graphics
ОП: 7851 Мб
НМД: 256 Гб
Монитор: 1920x1080

5 Программное обеспечение

Операционная система семейства: linux (ubuntu), версия 20.04.3 LTS
Интерпретатор команд: bash, версия 5.0.17(1)-release.
Система программирования: gcc*, версия 17
Редактор текстов: emacs, версия 25.2.2
Утилиты операционной системы: subl, make
Прикладные системы и программы: sublime text, bash
Местонахождение и имена файлов программ и данных на домашнем компьютере: /home/ggame/newlabs/

6 Идея, метод, алгоритм решения задачи

6.1 Основные функции

Для представления простейшей базы данных сделаем "строку"таблицы в виде структуры. Функция *void* create() создает пустой бинарный файл (если его нет). С помощью *void* add() можно заполнить одну строку таблицы (при этом в памяти хранится только данные одной структуры, все остальное уже лежит в файле). Функция *void* table_print() открывает бинарный файл на чтение и выводит всю таблицу. Также присутствует функция *void* delete(), которая удаляет файл.

6.2 Запрос на Си

Для реализации запроса варианта №4 необходимо получить минимальные характеристики компьютера. Создадим структуру, содержащую минимальные значения и заполним ее, затем считываем каждую строку таблицы бинарного файла, и если хранящиеся в строке данные меньше значений минимальной структуры, то выводим фамилию на экран (если таких компьютеров меньше p).

6.3 Запрос на Sql

Запрос задания также можно реализовать на Sql:

Листинг 1: pc.sql

```
select Surname from pc
WHERE (pc.PNum < 1 OR pc.RAM < 1 OR pc.Vmem < 1 OR pc.DCap < 7 OR pc.ICNum < 1 OR
    pc.PDNum < 1 OR pc.OS != "Ub") and (
select count(*) as cnt from pc
WHERE (pc.PNum < 1 OR pc.RAM < 1 OR pc.Vmem < 1 OR pc.DCap < 7 OR pc.ICNum < 1 OR
    pc.PDNum < 1 OR pc.OS != "Ub")
) > 0
```

7 Сценарий выполнения работы

Создадим таблицу из 25 структур случайных элементов, введем минимальные параметры:

```
ggame@ggame:~/newlabs/kp6$ gcc make_test.c
ggame@ggame:~/newlabs/kp6$ ./a.out 25
test1.bin
ggame@ggame:~/newlabs/kp6$ make
ggame@ggame:~/newlabs/kp6$ ./main -f 5
Enter file name
test1.bin
Enter minimal number of processors
3
Enter minimal size of RAM, GB
10
Enter minimal size of video memory, GB
2
Enter minimal number of disks
2
Enter minimal capacity of disk, GB
300
Enter minimal number of integrated controllers
4
Enter necessary OS
Ub

List of student which computers need to upgrade
bbmqbhcdar
scdxrjmowf
sarcbynecd
llnmpapqfw
wnkuewhsqm
swmdkqtbxi
nsnfwzqfjm
bcnuvqhffb
hchzvfrkml
xkitzyxacb
tomfgdwdwf
dlcgdewhta
wcsgspqoqm
xnzlgdgwpb
eyatdrmydi
zhlvihjouv
eimuotehzr
```

```
41      bipzzrzucx
42      nadqhdcnvw
43      auxnspusgd
44      cvudjsuyib
```

Мы вывели фамилии всех студентов, у которых хотя бы 1 из параметров меньше/не соответствует заданным.

8 Распечатка протокола

Листинг 2: computer.c

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include <string.h>
4   #include "computer.h"
5
6   void create()
7   {
8     char name[100];
9     FILE *data;
10    printf("Enter file name to create\n");
11    scanf("%s", name);
12    data = fopen(name, "rb");
13    if (data != NULL) {
14      printf("File already exists\n");
15    } else {
16      data = fopen(name, "a");
17    }
18    fclose(data);
19  }
20
21  void add()
22  {
23    char name[100];
24    FILE *data;
25    pc rec;
26    printf("Enter a file name\n");
27    scanf("%s", name);
28    data = fopen(name, "rb");
29    if (!data) {
30      printf("File not found\n");
31    } else {
32      data = fopen(name, "a");
33      printf("Enter a surname\n");
34      scanf("%s", rec.surname);
35      printf("Enter a number of processors\n");
36      scanf("%d", &rec.num_of_proc);
37      printf("Enter a type of processors\n");
38      scanf("%s", rec.type_of_proc);
39      printf("Enter a RAM size\n");
40      scanf("%d", &rec.mem_size_ram);
41      printf("Enter a type of controller\n");
42      scanf("%s", rec.type_of_contr);
43      printf("Enter a video memory size\n");
44      scanf("%d", &rec.mem_size_video_proc);
45      printf("Enter a type of disk\n");
46      scanf("%s", rec.type_of_disk);
47      printf("Enter a number of disks\n");
48      scanf("%d", &rec.num_of_disk);
49      printf("Enter a capacity of disk\n");
50      scanf("%d", &rec.cap_of_disk);
51      printf("Enter a number of integrated controllers\n");
52      scanf("%d", &rec.num_of_intcontr);
53      printf("Enter a number of peripheral devises\n");
```

```
54          scanf("%d", &rec.num_of_dev);
55          printf("Enter an OS\n");
56          scanf("%s", rec.os);
57          fwrite(&rec, sizeof(pc), 1, data);
58          fclose(data);
59        }
60      }
61
62      void table_print()
63      {
64        FILE *data;
65        pc read;
66        char name[100];
67        int count = 0;
68        printf("Enter a file name\n");
69        scanf("%s", name);
70        data = fopen(name, "rb");
71        if (!data) {
72          printf("File not found\n");
73        } else {
74          printf(
75          "Surname\t\tPNum\tPType\tRAM\t"
76          "CType\tVMem\tDType\tDNum\tDCap\t"
77          "ICNum\tPDNum\tOS\n");
78          while (fread(&read, sizeof(pc), 1, data) != EOF && !feof(data)) {
79            printf(
80            "%s\t%d\t%s\t%d\t%s\t%d\t%s\t%d\t%d\t%d\t%d\t%s\n",
81            read.surname, read.num_of_proc, read.type_of_proc, read.mem_size_ram,
82            read.type_of_contr, read.mem_size_video_proc, read.type_of_disk,
83            read.num_of_disk, read.cap_of_disk, read.num_of_intcontr,
84            read.num_of_dev, read.os
85            );
86          }
87          fclose(data);
88        }
89      }
90
91      void delete()
92      {
93        FILE *data;
94        char name[100];
95        printf("Enter file name\n");
96        scanf("%s", name);
97        data = fopen(name, "rb");
98        if (!data) {
99          printf("File not found\n");
100       } else {
101         remove(name);
102         printf("File was deleted\n");
103       }
104     }
105
106     void func(int p)
107     {
108       pc find, select;
109       char name[100];
110       FILE *data;
111       int count = 0;
112       printf("Enter file name\n");
113       scanf("%s", name);
114       data = fopen(name, "rb");
115       if (data == NULL) {
116         printf("File not found\n");
117         return;
118       } else {
119         data = fopen(name, "rb");
```

```
120
121         printf("Enter minimal number of processors\n");
122         scanf("%d", &select.num_of_proc);
123         printf("Enter minimal size of RAM, GB\n");
124         scanf("%d", &select.mem_size_ram);
125         printf("Enter minimal size of video memory, GB\n");
126         scanf("%d", &select.mem_size_video_proc);
127         printf("Enter minimal number of disks\n");
128         scanf("%d", &select.num_of_disk);
129         printf("Enter minimal capacity of disk, GB\n");
130         scanf("%d", &select.cap_of_disk);
131         printf("Enter minimal number of integrated controllers\n");
132         scanf("%d", &select.num_of_intcontr);
133         printf("Enter necessary OS\n");
134         scanf("%s", &select.os);
135
136         while (fread(&find, sizeof(pc), 1, data) != EOF && !feof(data)) {
137           if ((find.num_of_proc < select.num_of_proc) || (find.mem_size_ram < select.
                mem_size_ram)
138           || (find.mem_size_video_proc < select.mem_size_video_proc) || (find.
                num_of_disk < select.num_of_disk)
139           || (find.cap_of_disk < select.cap_of_disk) || (find.num_of_intcontr < select.
                num_of_intcontr)
140           || (!strcmp(find.os, select.os))) {
141             count++;
142           }
143         }
144         if (count < p) {
145           printf("Too few computers\n");
146           fclose(data);
147           return;
148         } else {
149           data = fopen(name, "rb");
150           printf("\nList of student which computers need to upgrade\n");
151           while (fread(&find, sizeof(pc), 1, data) != EOF && !feof(data)) {
152             if (!((find.num_of_proc >= select.num_of_proc) && (find.mem_size_ram >=
                    select.mem_size_ram)
153             && (find.mem_size_video_proc >= select.mem_size_video_proc) && (find.
                    num_of_disk >= select.num_of_disk)
154             && (find.cap_of_disk >= select.cap_of_disk) && (find.num_of_intcontr >=
                    select.num_of_intcontr)
155             && (!strcmp(find.os, select.os) ))) {
156               printf("\t%s\n", find.surname);
157             }
158           }
159         }
160         fclose(data);
161       }
162     }
```

Листинг 3: computer.h

```
1   #ifndef computer_h
2   #define computer_h
3
4   #include <stdio.h>
5   #include <stdlib.h>
6
7   typedef struct comp
8   {
9     char surname[50];
10    int num_of_proc;
11    char type_of_proc[50];
12    int mem_size_ram;
13    char type_of_contr[50];
14    int mem_size_video_proc;
```

```
15        char type_of_disk[50];
16        int num_of_disk;
17        int cap_of_disk;
18        int num_of_intcontr;
19        int num_of_dev;
20        char os[50];
21      } pc;
22
23      void create();
24
25      void add();
26
27      void print();
28
29      void table_print();
30
31      void delete();
32
33      void func(int p);
34
35      #endif
```

Листинг 4: main.c

```
1       #include <stdio.h>
2       #include <stdlib.h>
3       #include <string.h>
4       #include "computer.h"
5
6       int main(int argc, char const *argv[])
7       {
8         if (argc < 2) {
9           printf("Need two or three arguments:\n");
10          printf("\n -c \t\t for creating new dataset\n -a \t\t for adding new record\n -
                p \t\t for printing dataset\n -d \t\t for deleting dataset\n -f value\t for
                 make a task sample\n");
11          exit(1);
12        }
13        if (!strcmp(argv[1], "-a")) {
14          add();
15        } else if (!strcmp(argv[1], "-p")) {
16          table_print();
17        } else if (!strcmp(argv[1], "-d")) {
18          delete();
19        } else if (!strcmp(argv[1], "-c")) {
20          create();
21        } else if (!strcmp(argv[1], "-f")) {
22          int p;
23          sscanf(argv[2], "%d", &p);
24          func(p);
25        }
26        else {
27          printf("Command not found\n");
28          printf("\n -c \t\t for creating new dataset\n -a \t\t for adding new record\n -
                p \t\t for printing dataset\n -d \t\t for deleting dataset\n -f value\t for
                 make a task sample\n");
29        }
30        return 0;
31      }
```

9 Вывод

Благодаря данному КП я познакомился с бинарными файлами на языке Си и научился записывать и считывать в него структуры.