**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №11**

по дисциплине «Операционные системы»

на тему: «Управление файловой системой»

Выполнил: студент гр. ИТИ-11

Цветков А.А.

Принял: преподаватель-стажёр Карась О.В.

Гомель 2022

**Цель работы**: создать функцию для выполнения дефрагментации файловой системы.

|  |  |
| --- | --- |
| **Вариант** | **Дополнительная утилита** |
| 10 | Дефрагментация |

**Результат выполнения:**

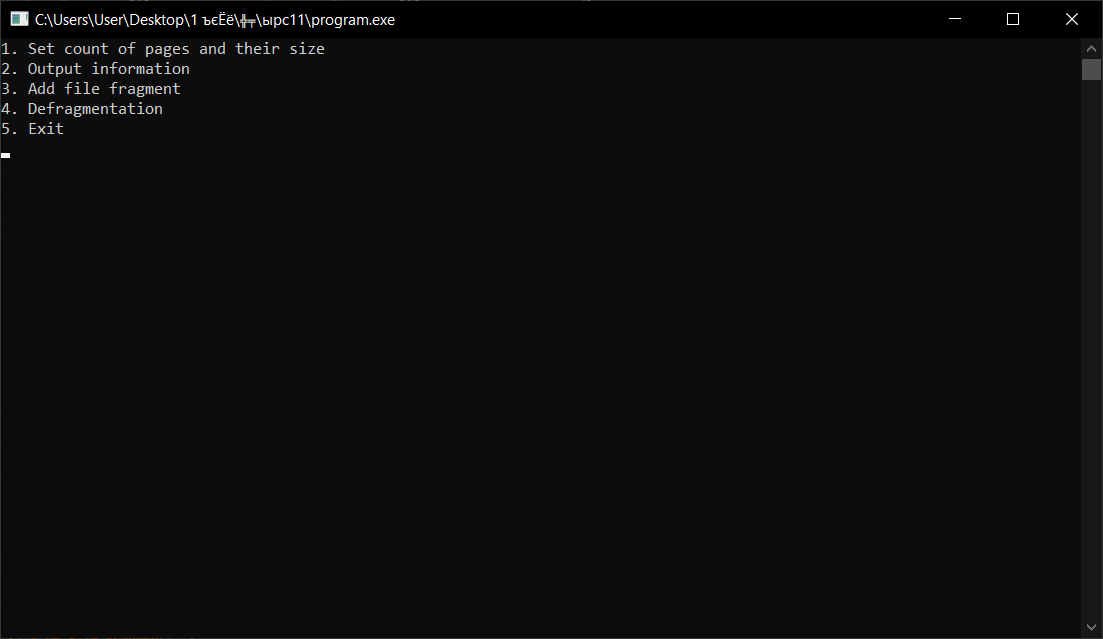


Рисунок 1 – Запуск программы

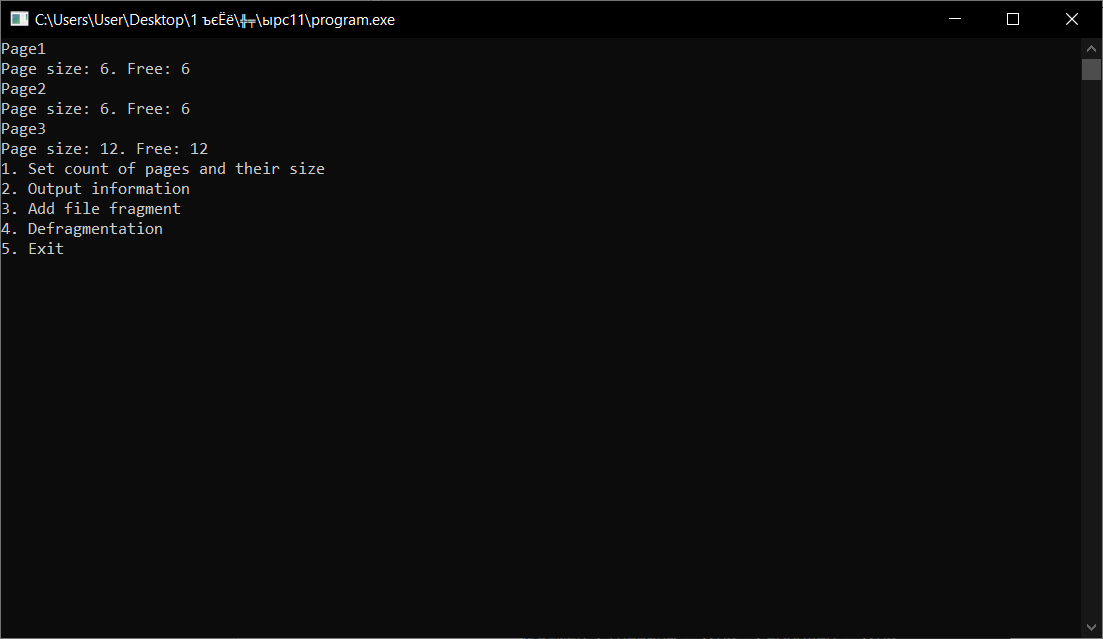


Рисунок 2 – Информация о страницах

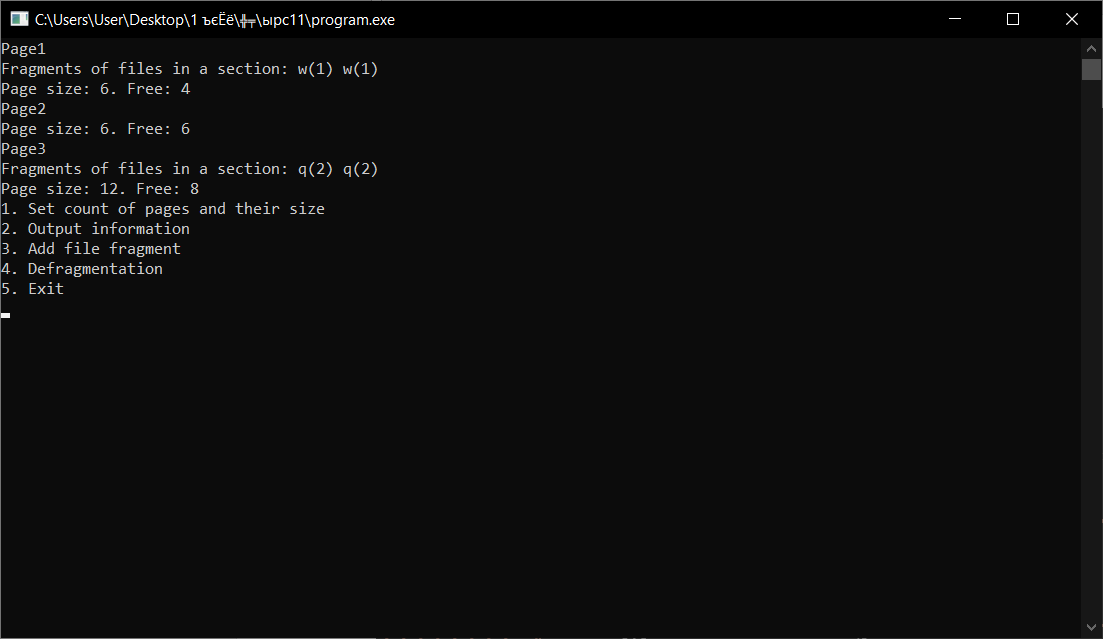


Рисунок 3 – Дефрагментация

**Вывод:** в ходе данной лабораторной работы были изучены методы дефрагментации и реализован один из них.

**ПРИЛОЖЕНИЕ А**

#include <locale.h>

#include <iostream>

#include <windows.h>

#include <vector>

#include <string>

#include <cstdlib>

#include <ctime>

using namespace std;

typedef struct {

    int pageNum;

    int isFreePage;

    int pageMemory;

    int pageFreeMemory;

} virtualPage;

typedef struct {

    string fileName;

    int fileMemory;

    int filePage;

    int isDef;

} virtualFile;

virtualPage \* page;

vector <virtualFile> files(1);

int fileNum = 0;

int pageNum = 0;

void inputData()

{

    int N, i = 0;

    cout << " Enter count of memory pages: ";

    cin >> N;

    page = new virtualPage[N];

    while (i < N)

{

cout << "Enter size of " << i + 1 << " page: ";

cin >> page[i].pageMemory;

page[i].pageNum = i + 1;

page[i].isFreePage = 1;

page[i].pageFreeMemory = page[i].pageMemory;

i++;

}

pageNum = N;

}

void outputData()

{

int j, i = 0;

bool isFirst;

while (i < pageNum)

{

j = 0;

isFirst = true;

cout << "Page" << i + 1;

while (j < fileNum)

{

if (files[j].filePage == page[i].pageNum)

{

if (isFirst)

{

cout << "\nFragments of files in a section: ";

isFirst = false;

}

cout << files[j].fileName << "(" << files[j].fileMemory << ") ";

}

j++;

}

cout << "\nPage size: " << page[i].pageMemory << ". Free: " << page[i].pageFreeMemory << endl;

i++;

}

}

int menu()

{

int n;

cout << "1. Set count of pages and their size\n";

cout << "2. Output information\n";

cout << "3. Add file fragment\n";

cout << "4. Defragmentation\n";

cout << "5. Exit\n";

cin >> n;

return n;

};

void inputFile()

{

int i = 0, isPage, memory;

string name;

if (fileNum == 0)

{

cout << "Enter file name: ";

cin >> name;

cout << "Enter page number: ";

cin >> isPage;

cout << "Enter size of file fragment: ";

cin >> memory;

}

else

{

files.insert(files.end(), files[0]);

cout << "Enter file name: ";

cin >> name;

cout << "Enter page number: ";

cin >> isPage;

cout << "Enter size of file fragment: ";

cin >> memory;

}

bool isFind = false;

while (memory > page[isPage - 1].pageFreeMemory && !isFind)

{

if (memory <= page[i].pageFreeMemory)

{

cout << "This section is busy, file was moved in section " << i + 1;

isPage = i + 1;

isFind = true;

}

i++;

if (i > pageNum)

{

cout << "Error";

return;

}

}

files[fileNum].fileName = name;

files[fileNum].filePage = isPage;

files[fileNum].fileMemory = memory;

files[fileNum].isDef = 0;

page[isPage - 1].pageFreeMemory -= memory;

page[isPage - 1].isFreePage = 0;

fileNum++;

}

void oneDefrag()

{

int i = 0, j = 0, k = 0, size, num, pg;

string name;

bool isFind;

while (i < fileNum)

{

j = 0;

k = 0;

name = files[i].fileName;

pg = files[i].filePage;

size = 0;

isFind = false;

while (j < fileNum)

{

if (name == files[j].fileName && !files[j].isDef)

{

size += files[j].fileMemory;

if (pg != files[j].filePage)

isFind = true;

}

j++;

}

j = 0;

if (isFind)

{

while (k < pageNum)

{

if (page[k].pageFreeMemory >= size)

{

while (j < fileNum)

{

if ( name == files[j].fileName && files[j].filePage != (k + 1) )

{

num = files[j].filePage - 1;

files[j].filePage = k + 1;

page[k].pageFreeMemory -= files[j].fileMemory;

files[j].isDef = 1;

page[num].pageFreeMemory += files[j].fileMemory;

}

j++;

}

}

k++;

}

}

i++;

}

}

int main()

{

int n;

while ((n = menu()) != 5)

{

system("cls");

switch (n)

{

case 1: inputData();

break;

case 2: outputData();

break;

case 3: inputFile();

break;

case 4: oneDefrag();

break;

default:

break;

}

}

system("pause");

delete [] page;

return 0;

}