Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный технологический университет»

ФИТ

Кафедра Информационных систем и технологий 1курс

Группа 2-2

**Структуры данных**

Тема:

Редактирование файлов: частичное и полное удаление, изменение поля.

Лабораторная работа 10, 11

**Выполнил:**

Волковец Сергей Николаевич

**Проверил:**

Белодед Николай Иванович

Оглавление

[Теоретические сведения 3](#_Toc196250205)

[Использование функции remove() 3](#_Toc196250206)

# Теоретические сведения

В C++ существует несколько способов удаления данных из файла. Некоторые из них перечислены ниже:  
 Открытие файла в режиме ios::trunc

Как упоминалось ранее, при открытии файла в режиме ios::trunc содержимое файла обрезается до нулевой длины. Если файл не существует, он создается.

|  |
| --- |
| ofstream myfile; myfile.open("example.txt", ios::out | ios::trunc); |

Использование функции remove()  
 Функция remove() удаляет файл из файловой системы. Она принимает в качестве аргумента имя файла, который нужно удалить.

|  |
| --- |
| #include <cstdio>  int main() {  if (remove("example.txt") != 0) {  perror("Ошибка удаления файла");  }  else {  puts("Файл успешно удален");  }  return 0; } |

Чтобы удалить структуру из массива структур по индексу, вы можете использовать следующий алгоритм:  
 1. Сначала скопируйте все структуры, которые находятся после удаляемой структуры, на одну позицию влево. Это можно сделать с помощью цикла for.  
 2. Затем уменьшите размер массива на 1, чтобы удалить последнюю структуру, которую вы только что скопировали на предыдущем шаге.

Пример кода, который демонстрирует, как удалить структуру из массива структур по индексу:

|  |
| --- |
| #include <iostream>  using namespace std;4+  struct Person {  string name;  int age; };  int main() {  Person people[3] = { {"Alice", 25}, {"Bob", 30}, {"Charlie", 35} };  int index = 1; // Индекс удаляемой структуры  // Создание нового массива меньшего размера  Person newPeople[2];    // Копирование всех структур до удаляемой структуры  for (int i = 0; i < index; i++) {  newPeople[i] = people[i];  }  // Копирование всех структур после удаляемой структуры на одну позицию влево  for (int i = index; i < 2; i++) {  newPeople[i] = people[i + 1];  }    // Вывод оставшихся структур  for (int i = 0; i < 2; i++) {  cout << newPeople[i].name << " " << newPeople[i].age << endl;  }  return 0; } |

После удаления одной из структур из массива мы можем переписать файл, чтоб у нас хранилась обновленная информация.  
 Далее разберём изменение структуры в массиве по индексу, проверяя, находится ли индекс в пределах массива. Если индекс находится в пределах массива, мы изменяем поля name и age структуры по этому индексу. Если же индекс находится за пределами массива, мы выводим сообщение об ошибке. Затем мы выводим все структуры в массиве, чтобы убедиться в том, что изменения были выполнены правильно.

|  |
| --- |
| #include <iostream> #include <string> using namespace std;  struct Person {  string name;  int age; };  int main() {  Person people[3] = { {"Alice", 25}, {"Bob", 30}, {"Charlie", 35} };    // Редактирование структуры по индексу  int index = 1;  if (index >= 0 && index < 3) {  people[index].name = "Bill";  people[index].age = 28;  }  else {  cout << "Индекс за пределами массива" << endl;  }    // Вывод отредактированных структур  for (int i = 0; i < 3; i++) {  cout << people[i].name << " " << people[i].age << endl;  }  return 0; } |

# Задачи

1. Написать функции удаления(очистки) содержимого вашей БД и полного удаления вашей БД. Функции удаляют информацию как из файла, так и из структуры БД.

Код:

// (10, 11) - Написать функции удаления(очистки) содержимого

// вашей БД и полного удаления вашей БД.

// Функции удаляют информацию как из файла, так и из структуры БД

#include <iostream>

#include <fstream>

#include <string>

#include <cstdio>

using namespace std;

struct Record {

int id;

string name;

};

const int dbCapacity = 100;

Record\* database = new Record[dbCapacity];

int dbSize = 0;

void addRecord(int id, const string& name) {

if (dbSize < dbCapacity) {

database[dbSize].id = id;

database[dbSize].name = name;

dbSize++;

cout << "Запись добавлена!" << endl;

}

else {

cout << "База данных заполнена." << endl;

}

}

void printDB() {

if (dbSize == 0) {

cout << "БД пуста." << endl;

}

else {

cout << "Содержимое БД:" << endl;

for (int i = 0; i < dbSize; i++) {

cout << "id: " << database[i].id

<< ", name: " << database[i].name << endl;

}

}

}

void writeDBToFile() {

ofstream outFile("1.txt", ios::out | ios::trunc);

if (!outFile) {

cout << "Ошибка открытия файла для записи." << endl;

return;

}

for (int i = 0; i < dbSize; i++) {

outFile << "id: " << database[i].id

<< ", name: " << database[i].name << "\n";

}

outFile.close();

cout << "БД записана в файл 1.txt." << endl;

}

void clearDBContent() {

dbSize = 0;

cout << "Содержимое БД в памяти очищено." << endl;

ofstream outFile("1.txt", ios::out | ios::trunc);

if (!outFile) {

cout << "Ошибка открытия файла для очистки." << endl;

}

outFile.close();

cout << "Содержимое файла 1.txt очищено." << endl;

}

void deleteDB() {

delete[] database;

database = nullptr;

dbSize = 0;

cout << "БД в памяти полностью удалена." << endl;

if (remove("1.txt") == 0) {

cout << "Файл 1.txt удалён." << endl;

}

else {

cout << "Ошибка при удалении файла 1.txt или файл не существует." << endl;

}

}

int main() {

setlocale(LC\_ALL, "RU");

int choice;

bool exitProgram = false;

while (!exitProgram) {

cout << "\nВыберите действие:" << endl;

cout << "1. Добавить запись" << endl;

cout << "2. Вывести содержимое БД" << endl;

cout << "3. Записать БД в файл" << endl;

cout << "4. Очистить содержимое БД и файла" << endl;

cout << "5. Полное удаление БД (память и файл)" << endl;

cout << "6. Выход" << endl;

cout << "Введите номер действия: ";

cin >> choice;

switch (choice) {

case 1: {

int id;

string name;

cout << "Введите id: ";

cin >> id;

cout << "Введите имя: ";

cin.ignore();

getline(cin, name);

addRecord(id, name);

break;

}

case 2:

printDB();

break;

case 3:

writeDBToFile();

break;

case 4:

clearDBContent();

break;

case 5:

deleteDB();

exitProgram = true;

break;

case 6:

exitProgram = true;

break;

default:

cout << "Неверный выбор. Попробуйте ещё раз." << endl;

}

}

return 0;

}

2) Написать функции удаления и редактирования элемента структуры по индексу в вашей БД.

Код:

#include <iostream>

#include <fstream>

#include <string>

#include <cstdio>

using namespace std;

struct Record {

int id;

string name;

};

const int dbCapacity = 100;

Record\* database = new Record[dbCapacity];

int dbSize = 0;

void addRecord(int id, const string& name) {

if (dbSize < dbCapacity) {

database[dbSize].id = id;

database[dbSize].name = name;

dbSize++;

cout << "Запись добавлена!" << endl;

}

else {

cout << "База данных заполнена." << endl;

}

}

void printDB() {

if (dbSize == 0) {

cout << "БД пуста." << endl;

}

else {

cout << "Содержимое БД:" << endl;

for (int i = 0; i < dbSize; i++) {

cout << "id: " << database[i].id

<< ", name: " << database[i].name << endl;

}

}

}

void writeDBToFile() {

ofstream outFile("1.txt", ios::out | ios::trunc);

if (!outFile) {

cout << "Ошибка открытия файла для записи." << endl;

return;

}

for (int i = 0; i < dbSize; i++) {

outFile << "id: " << database[i].id

<< ", name: " << database[i].name << "\n";

}

outFile.close();

cout << "БД записана в файл 1.txt." << endl;

}

void clearDBContent() {

dbSize = 0;

cout << "Содержимое БД в памяти очищено." << endl;

ofstream outFile("1.txt", ios::out | ios::trunc);

if (!outFile) {

cout << "Ошибка открытия файла для очистки." << endl;

}

outFile.close();

cout << "Содержимое файла 1.txt очищено." << endl;

}

void deleteDB() {

delete[] database;

database = nullptr;

dbSize = 0;

cout << "БД в памяти полностью удалена." << endl;

if (remove("1.txt") == 0) {

cout << "Файл 1.txt удалён." << endl;

}

else {

cout << "Ошибка при удалении файла 1.txt или файл не существует." << endl;

}

}

void deleteRecord(int index) {

if (index < 0 || index >= dbSize) {

cout << "Некорректный индекс. Удаление невозможно." << endl;

return;

}

for (int i = index; i < dbSize - 1; i++) {

database[i] = database[i + 1];

}

dbSize--;

cout << "Запись удалена." << endl;

}

void editRecord(int index, int newId, const string& newName) {

if (index < 0 || index >= dbSize) {

cout << "Некорректный индекс. Редактирование невозможно." << endl;

return;

}

database[index].id = newId;

database[index].name = newName;

cout << "Запись обновлена." << endl;

}

int main() {

setlocale(LC\_ALL, "RU");

int choice;

bool exitProgram = false;

while (!exitProgram) {

cout << "\nВыберите действие:" << endl;

cout << "1. Добавить запись" << endl;

cout << "2. Вывести содержимое БД" << endl;

cout << "3. Записать БД в файл" << endl;

cout << "4. Очистить содержимое БД и файла" << endl;

cout << "5. Полное удаление БД (память и файл)" << endl;

cout << "6. Удалить запись по индексу" << endl;

cout << "7. Редактировать запись по индексу" << endl;

cout << "8. Выход" << endl;

cout << "Введите номер действия: ";

cin >> choice;

switch (choice) {

case 1: {

int id;

string name;

cout << "Введите id: ";

cin >> id;

cout << "Введите имя: ";

cin.ignore();

getline(cin, name);

addRecord(id, name);

break;

}

case 2:

printDB();

break;

case 3:

writeDBToFile();

break;

case 4:

clearDBContent();

break;

case 5:

deleteDB();

exitProgram = true;

break;

case 6: {

int deleteIndex;

cout << "Введите индекс удаляемой записи: ";

cin >> deleteIndex;

deleteRecord(deleteIndex);

break;

}

case 7: {

int editIndex, newId;

string newName;

cout << "Введите индекс редактируемой записи: ";

cin >> editIndex;

cout << "Введите новый id: ";

cin >> newId;

cout << "Введите новое имя: ";

cin.ignore();

getline(cin, newName);

editRecord(editIndex, newId, newName);

break;

}

case 8:

exitProgram = true;

break;

default:

cout << "Неверный выбор. Попробуйте ещё раз." << endl;

}

}

return 0;

}