МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждения образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

Направление специальности 1-40 01 01 10 Программное обеспечение информационных технологий (программирование интернет-приложений)

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ:

по дисциплине «Операционные системы»

Исполнитель

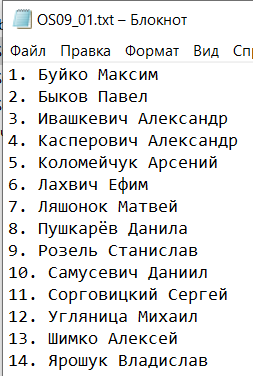
студент (ка) 3 курса группы 6 Розель Станислав Александрович

(Ф.И.О.)

Минск 2024

Задание 1

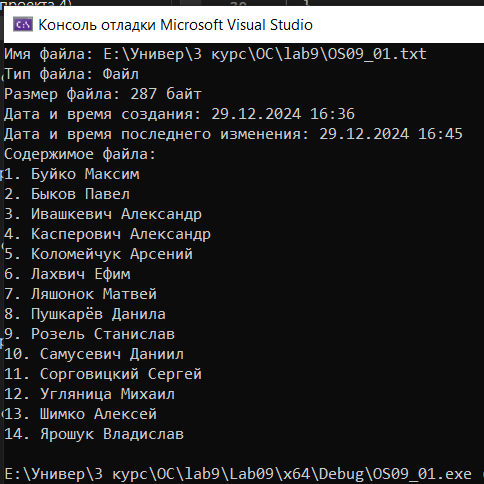
Список подгруппы:



Код программы:

|  |
| --- |
| #define \_CRT\_NON\_CONFORMING\_WCSTOK  #define \_CRT\_SECURE\_NO\_WARNINGS  #include <iostream>  #include <cstdlib>  #include "Windows.h"  #include <string>  #include <iomanip>  #include <memory>  using namespace std;  #define FILE\_PATH L"E:\\Универ\\3 курс\\OC\\lab9\\OS09\_01.txt"  void printFileInfo(LPWSTR FileName);  void printFileTxt(LPWSTR FileName);  int main()  {  setlocale(LC\_ALL, "rus");  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  LPWSTR fileName = (LPWSTR)FILE\_PATH;  printFileInfo(fileName);  printFileTxt(fileName);  return 0;  }  void printFileInfo(LPWSTR FileName)  {  WIN32\_FILE\_ATTRIBUTE\_DATA fileInfo;  if (GetFileAttributesEx(FileName, GetFileExInfoStandard, &fileInfo)) {  wcout << L"Имя файла: " << FileName << endl;  DWORD fileAttributes = fileInfo.dwFileAttributes;  wcout << L"Тип файла: ";  wcout << ((fileAttributes & FILE\_ATTRIBUTE\_DIRECTORY) ? L"Каталог" : L"Файл") << endl;  LARGE\_INTEGER fileSize;  fileSize.HighPart = fileInfo.nFileSizeHigh;  fileSize.LowPart = fileInfo.nFileSizeLow;  wcout << L"Размер файла: " << fileSize.QuadPart << L" байт" << endl;  SYSTEMTIME stUTC, stLocal;  FileTimeToSystemTime(&fileInfo.ftCreationTime, &stUTC);  SystemTimeToTzSpecificLocalTime(NULL, &stUTC, &stLocal);  wcout << L"Дата и время создания: " << stLocal.wDay << L"." << stLocal.wMonth << L"." << stLocal.wYear  << L" " << setw(2) << setfill(L'0') << stLocal.wHour << L":" << setw(2) << stLocal.wMinute << endl;  FileTimeToSystemTime(&fileInfo.ftLastWriteTime, &stUTC);  SystemTimeToTzSpecificLocalTime(NULL, &stUTC, &stLocal);  wcout << L"Дата и время последнего изменения: " << stLocal.wDay << L"." << stLocal.wMonth << L"." << stLocal.wYear  << L" " << setw(2) << setfill(L'0') << stLocal.wHour << L":" << setw(2) << stLocal.wMinute << endl;  }  else {  wcerr << L"Ошибка получения информации о файле." << endl;  }  }  void printFileTxt(LPWSTR FileName)  {  HANDLE file = CreateFile(  FileName,  GENERIC\_READ,  FILE\_SHARE\_READ,  NULL,  OPEN\_EXISTING,  FILE\_ATTRIBUTE\_NORMAL,  NULL);  if (file == INVALID\_HANDLE\_VALUE) {  cerr << "Ошибка открытия файла." << endl;  return;  }  LARGE\_INTEGER fileSize;  if (GetFileSizeEx(file, &fileSize)) {  auto buffer = make\_unique<char[]>(fileSize.QuadPart + 1);  ZeroMemory(buffer.get(), fileSize.QuadPart + 1);  DWORD bytesRead;  if (ReadFile(file, buffer.get(), fileSize.QuadPart, &bytesRead, NULL)) {  cout << "Содержимое файла:" << endl << buffer.get() << endl;  }  else {  cerr << "Ошибка чтения файла." << endl;  }  }  else {  cerr << "Ошибка получения размера файла." << endl;  }  CloseHandle(file);  } |

Результат работы программы:

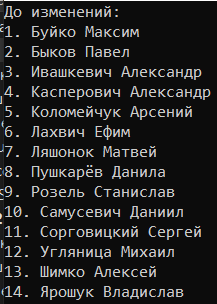
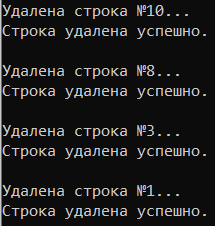
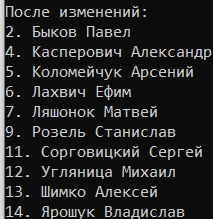


Задание 2

Код функции delRowFileTxt:

|  |
| --- |
| BOOL delRowFileTxt(LPWSTR FileName, DWORD row)  {  LARGE\_INTEGER fileSize = {};  bool rowFound = false;  try  {  if (row <= 0)  {  throw "Номер строки должен быть положительным.";  }  HANDLE of = CreateFile(  FileName,  GENERIC\_READ | GENERIC\_WRITE,  FILE\_SHARE\_READ,  NULL,  OPEN\_EXISTING,  FILE\_ATTRIBUTE\_NORMAL,  NULL);  if (of == INVALID\_HANDLE\_VALUE)  {  throw "Ошибка открытия файла.";  }  if (!GetFileSizeEx(of, &fileSize))  {  CloseHandle(of);  throw "Ошибка получения размера файла.";  }  char\* buf = new char[fileSize.QuadPart + 1];  char\* bufAfterDel = new char[fileSize.QuadPart + 1];  ZeroMemory(buf, fileSize.QuadPart + 1);  ZeroMemory(bufAfterDel, fileSize.QuadPart + 1);  DWORD bytesRead = 0;  if (!ReadFile(of, buf, fileSize.QuadPart, &bytesRead, NULL))  {  CloseHandle(of);  delete[] buf;  delete[] bufAfterDel;  throw "Ошибка чтения файла.";  }  buf[fileSize.QuadPart] = '\0';  int currentRow = 1, pos = 0, posAfter = 0;  while (buf[pos] != '\0')  {  if (currentRow == row)  {  rowFound = true;  while (buf[pos] != '\n' && buf[pos] != '\0')  {  pos++;  }  if (buf[pos] == '\n')  {  pos++;  }  currentRow++;  }  else  {  if (buf[pos] == '\n')  {  currentRow++;  }  bufAfterDel[posAfter++] = buf[pos++];  }  }  if (!rowFound)  {  CloseHandle(of);  delete[] buf;  delete[] bufAfterDel;  throw "Указанная строка не найдена.";  }  bufAfterDel[posAfter] = '\0';  SetFilePointer(of, 0, NULL, FILE\_BEGIN);  DWORD bytesWritten = 0;  if (!WriteFile(of, bufAfterDel, strlen(bufAfterDel), &bytesWritten, NULL))  {  CloseHandle(of);  delete[] buf;  delete[] bufAfterDel;  throw "Ошибка записи в файл.";  }  SetEndOfFile(of);  std::cout << "Строка удалена успешно.\n";  CloseHandle(of);  delete[] buf;  delete[] bufAfterDel;  }  catch (const char\* err)  {  std::cout << "Ошибка:\n" << err << std::endl;  return FALSE;  }  return TRUE;  } |

Результат удаления:

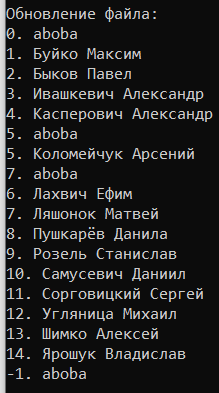
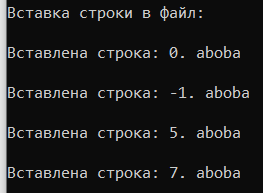
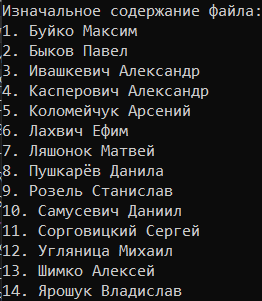
  

Задание 3

Код функции insRowFileTxt:

|  |
| --- |
| BOOL insRowFileTxt(LPWSTR fileName, LPWSTR str, DWORD row) {  WaitForSingleObject(mutex, INFINITE);  HANDLE file = CreateFile(  fileName,  GENERIC\_READ | GENERIC\_WRITE,  FILE\_SHARE\_READ,  NULL,  OPEN\_ALWAYS,  FILE\_ATTRIBUTE\_NORMAL,  NULL);  if (file == INVALID\_HANDLE\_VALUE) {  cout << "Ошибка: Не удалось открыть файл." << endl;  ReleaseMutex(mutex);  return FALSE;  }  DWORD fileSize = GetFileSize(file, NULL);  if (fileSize == INVALID\_FILE\_SIZE) {  cout << "Ошибка: Не удалось получить размер файла." << endl;  CloseHandle(file);  ReleaseMutex(mutex);  return FALSE;  }  char\* buffer = new char[fileSize + 1];  ZeroMemory(buffer, fileSize + 1);  DWORD bytesRead;  if (!ReadFile(file, buffer, fileSize, &bytesRead, NULL)) {  cout << "Ошибка: Не удалось прочитать файл." << endl;  delete[] buffer;  CloseHandle(file);  ReleaseMutex(mutex);  return FALSE;  }  buffer[fileSize] = '\0';  char\* newRow = new char[wcslen(str) \* 2 + 3];  wcstombs(newRow, str, wcslen(str) \* 2);  strcat(newRow, "\r\n");  string result;  int currentRow = 0;  bool inserted = false;  char\* line = strtok(buffer, "\n");  while (line != NULL) {  if (currentRow == row && !inserted) {  result += newRow;  inserted = true;  }  result += line;  result += "\n";  line = strtok(NULL, "\n");  currentRow++;  }  if (row == -1 || row >= currentRow) {  result += newRow;  }  SetFilePointer(file, 0, NULL, FILE\_BEGIN);  DWORD bytesWritten;  if (!WriteFile(file, result.c\_str(), result.size(), &bytesWritten, NULL)) {  cout << "Ошибка: Не удалось записать в файл." << endl;  }  SetEndOfFile(file);  cout << "\nВставлена строка: " << newRow;  delete[] buffer;  delete[] newRow;  CloseHandle(file);  ReleaseMutex(mutex);  return TRUE;  } |

Результат работы функции:

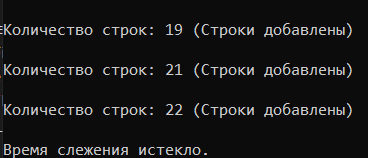


Задание 4

Код функции printWatchRowFileTxt

|  |
| --- |
| BOOL printWatchRowFileTxt(LPWSTR FileName, DWORD mlsec, DWORD maxDuration)  {  LARGE\_INTEGER fileSize = { 0 };  int rowCount = 0;  try {  HANDLE notif = FindFirstChangeNotification(FOLDER\_PATH, FALSE, FILE\_NOTIFY\_CHANGE\_LAST\_WRITE);  if (notif == INVALID\_HANDLE\_VALUE) {  cout << "Ошибка при инициализации уведомления о изменениях." << endl;  return FALSE;  }  DWORD dwWaitStatus;  DWORD startTime = GetTickCount();  while (true) {  if ((GetTickCount() - startTime) >= maxDuration) {  cout << "\nВремя слежения истекло." << endl;  break;  }  dwWaitStatus = WaitForSingleObject(notif, mlsec);  switch (dwWaitStatus) {  case WAIT\_OBJECT\_0: {  if (FindNextChangeNotification(notif) == FALSE) {  break;  }  WaitForSingleObject(mutex, INFINITE);  HANDLE of = CreateFile(  FileName,  GENERIC\_READ,  FILE\_SHARE\_READ,  NULL,  OPEN\_EXISTING,  FILE\_ATTRIBUTE\_NORMAL,  NULL  );  if (of == INVALID\_HANDLE\_VALUE) {  DWORD dwError = GetLastError();  cout << "Ошибка: не удалось открыть файл. Код ошибки: " << dwError << endl;  }  else if (GetFileSizeEx(of, &fileSize)) {  char\* buf = new char[(fileSize.QuadPart + 1) \* sizeof(char)];  ZeroMemory(buf, (fileSize.QuadPart + 1) \* sizeof(char));  DWORD n = 0;  if (ReadFile(of, buf, fileSize.QuadPart, &n, NULL)) {  int position = 0;  rowCount = 0;  bool lastLineIsEmpty = false;  while (buf[position] != '\0') {  if (buf[position] == '\n') {  rowCount++;  lastLineIsEmpty = false;  }  else {  lastLineIsEmpty = true;  }  position++;  }  if (lastLineIsEmpty && position > 0 && buf[position - 1] != '\n') {  rowCount++;  }  }  delete[] buf;  CloseHandle(of);  }  else {  cout << "Ошибка при чтении файла." << endl;  }  ReleaseMutex(mutex);  if (rowC != rowCount) {  cout << "\nКоличество строк: " << rowCount;  if (rowCount > rowC) {  cout << " (Строки добавлены)" << endl;  }  else if (rowCount < rowC) {  cout << " (Строки удалены)" << endl;  }  rowC = rowCount;  }  break;  }  default:  break;  }  }  CloseHandle(notif);  }  catch (const char\* err) {  cout << "Ошибка: " << err << "\n";  return false;  }  return true;  } |

Результат работы программы:



Задание 5

Код программы:

|  |
| --- |
| #include <iostream>  #include <fstream>  #include <string>  int countLinesInFile(const std::string& filePath) {  std::ifstream file(filePath);  if (!file.is\_open()) {  std::cerr << "Ошибка: Не удалось открыть файл " << filePath << std::endl;  return -1;  }  int lineCount = 0;  std::string line;  while (std::getline(file, line)) {  ++lineCount;  }  file.close();  return lineCount;  }  int main() {  std::string filePath = "OS09\_05.txt";  int lineCount = countLinesInFile(filePath);  if (lineCount >= 0) {  std::cout << "Количество строк в файле \"" << filePath << "\": " << lineCount << std::endl;  }  return 0;  } |

Результат выполнения:

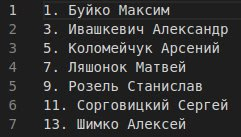


Задание 6

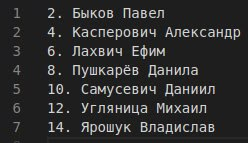
Код программы:

|  |
| --- |
| #include <iostream>  #include <fstream>  #include <string>  #include <cstdlib>  void processFile(const std::string& inputFile, const std::string& outputFile, bool odd) {  std::ifstream input(inputFile);  if (!input.is\_open()) {  std::cerr << "Ошибка: Не удалось открыть файл " << inputFile << std::endl;  return;  }  std::ofstream output(outputFile);  if (!output.is\_open()) {  std::cerr << "Ошибка: Не удалось создать файл " << outputFile << std::endl;  input.close();  return;  }  std::string line;  int lineNumber = 1;  while (std::getline(input, line)) {  if ((odd && lineNumber % 2 != 0) || (!odd && lineNumber % 2 == 0)) {  output << line << std::endl;  }  ++lineNumber;  }  input.close();  output.close();  std::cout << "Файл \"" << outputFile << "\" успешно создан." << std::endl;  }  int main(int argc, char\* argv[]) {  if (argc != 2) {  std::cerr << "Использование: " << argv[0] << " <целое число>" << std::endl;  return 1;  }  int number = std::atoi(argv[1]);  std::string inputFile = "OS09\_05.txt";  std::string outputFile = (number % 2 == 0) ? "OS09\_06\_2.txt" : "OS09\_06\_1.txt";  bool odd = (number % 2 != 0);  processFile(inputFile, outputFile, odd);  return 0;  } |









Задание 7

|  |
| --- |
| #include <iostream>  #include <fcntl.h>  #include <unistd.h>  #include <cstring>  int main() {  const char\* fileName = "example.txt";  int fd = open(fileName, O\_RDONLY);  if (fd == -1) {  std::cerr << "Ошибка открытия файла: " << strerror(errno) << std::endl;  return 1;  }  off\_t position = 15;  if (lseek(fd, position, SEEK\_SET) == -1) {  std::cerr << "Ошибка перемещения указателя: " << strerror(errno) << std::endl;  close(fd);  return 1;  }  char buffer[10] = {0};  ssize\_t bytesRead = read(fd, buffer, 7);  if (bytesRead == -1) {  std::cerr << "Ошибка чтения файла: " << strerror(errno) << std::endl;  close(fd);  return 1;  }  std::cout << "Прочитанная строка: \"" << buffer << "\"" << std::endl;  close(fd);  return 0;  } |

