Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Операционные системы

Студент: Трубач Д.С.

ФИТ 3 курс 5 группа

Преподаватель: Савельева М.Г.

Минск 2023

**Лабораторная работа №9**

Задание 1. С помощью Notepad (Notepad+) создайте на дисковом устройстве текстовый файл **OS09\_01.txt.** Заполните его 10 строками из списка студентов вашей подгруппы.



Разработайте приложение **OS09\_01**.

Приложение **OS09\_01** вызывает функцию **printFileInfo**, имеющую следующий прототип.



Функция **printFileInfo** выводит в стандартный поток вывода следующую информацию:

- имя файла;

- тип файла;

- размер файла;

- дата и время создания файла;

- дата и время последнего обновления.

Приложение **OS09\_01** вызывает функцию **printFileTxt**, имеющую следующий прототип.



Функция **printFileTxt** выводит в стандартный поток вывода содержимое файла.

При вызове функции укажите в качестве параметра имя текстового файла, созданного в п.2.

Продемонстрируйте работоспособность приложения **OS09\_01**.

|  |  |
| --- | --- |
| Код программы OS09\_01 | #include <Windows.h>  #include <iostream>  using namespace std;  #define FILE\_PATH L"E:/3course/5sem/OS/Lab9/OS09\_01.txt"  #define READ\_BYTES 450  string getFileName(wchar\_t\* filePath)  {  wstring ws(filePath);  string filename(ws.begin(), ws.end());  const size\_t last\_slash\_idx = filename.find\_last\_of("\\/");  if (string::npos != last\_slash\_idx)  filename.erase(0, last\_slash\_idx + 1);  return filename;  }  LPCWSTR getFileType(HANDLE file)  {  switch (GetFileType(file))  {  case FILE\_TYPE\_UNKNOWN:  return L"FILE\_TYPE\_UNKNOWN";  case FILE\_TYPE\_DISK:  return L"FILE\_TYPE\_DISK";  case FILE\_TYPE\_CHAR:  return L"FILE\_TYPE\_CHAR";  case FILE\_TYPE\_PIPE:  return L"FILE\_TYPE\_PIPE";  case FILE\_TYPE\_REMOTE:  return L"FILE\_TYPE\_REMOTE";  default:  return L"[ERROR]: WRITE FILE TYPE";  }  }  BOOL printFileInfo(LPWSTR path)  {  HANDLE file = CreateFile(  path,  GENERIC\_READ,  NULL,  NULL,  OPEN\_ALWAYS,  FILE\_ATTRIBUTE\_NORMAL,  NULL);  SYSTEMTIME sysTime;  FILETIME localTime;  BY\_HANDLE\_FILE\_INFORMATION fi;  BOOL fResult = GetFileInformationByHandle(file, &fi);  if (fResult)  {  cout << "File name:\t" << getFileName((wchar\_t\*)FILE\_PATH);  wcout << "\nFile type:\t" << getFileType(file);  cout << "\nFile size:\t" << fi.nFileSizeLow << " bytes";  FileTimeToLocalFileTime(&fi.ftCreationTime, &localTime);  FileTimeToSystemTime(&localTime, &sysTime);  printf("\nCreate time:\t%02d.%02d.%04d %02dh %02dm %02ds", sysTime.wDay, sysTime.wMonth, sysTime.wYear, sysTime.wHour % 24, sysTime.wMinute, sysTime.wSecond);  FileTimeToLocalFileTime(&fi.ftLastWriteTime, &localTime);  FileTimeToSystemTime(&localTime, &sysTime);  printf("\nUpdate time:\t%02d.%02d.%04d %02dh %02dm %02ds", sysTime.wDay, sysTime.wMonth, sysTime.wYear, sysTime.wHour % 24, sysTime.wMinute, sysTime.wSecond);  }  CloseHandle(file);  return true;  }  BOOL printFileTxt(LPWSTR path)  {  HANDLE file = CreateFile(  path,  GENERIC\_READ,  NULL,  NULL,  OPEN\_ALWAYS,  FILE\_ATTRIBUTE\_NORMAL,  NULL);  char buf[1024];  DWORD countBytes = NULL;  ZeroMemory(buf, sizeof(buf));  BOOL b = ReadFile(  file,  &buf,  READ\_BYTES,  &countBytes,  NULL);  if (!b) throw "[ERROR] Readfile throwed exception.";  cout << "\n\n\tPrint file (" << countBytes << " bytes)\n" << buf << '\n';  }  int main()  {  setlocale(LC\_ALL, "ru");  LPWSTR path = (LPWSTR)FILE\_PATH;  cout << "\n======================================\n\n";  printFileInfo(path);  cout << "\n\n======================================";  printFileTxt(path);  cout << "\n======================================\n";  } |
| Скриншот выполнения |  |

Задание 2. Разработайте приложение **OS09\_02.**

Приложение **OS09\_02** вызывает функцию **delRowFileTxt**, имеющую следующий прототип.



Функция применяется к файлу **OS09\_01.txt** (п.2) и вызывается последовательно 4 раза, с row = 1,3,8,10. Результат выполнения продемонстрируйте с помощью функции **printFileTxt** (п.6).

Продемонстрируйте работоспособность приложения **OS09\_02**.

|  |  |
| --- | --- |
| Код программы OS09\_02 | #define \_CRT\_SECURE\_NO\_WARNINGS  #include <Windows.h>  #include <iostream>  using namespace std;  #define FILE\_PATH L"E:/3course/5sem/OS/Lab9/OS09\_01.txt"  #define READ\_BYTES 500  BOOL printFileText(LPWSTR fileName)  {  try  {  cout << "\n\n\t====== RESULT ======\n";  HANDLE hf = CreateFile(fileName, GENERIC\_READ, NULL, NULL, OPEN\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, NULL);  if (hf == INVALID\_HANDLE\_VALUE) throw "[ERROR] Create or open file failed.";  DWORD n = NULL;  char buf[1024];  ZeroMemory(buf, sizeof(buf));  BOOL b = ReadFile(hf, &buf, READ\_BYTES, &n, NULL);  if (!b) throw "[ERROR] Read file failed";  cout << buf << endl;  CloseHandle(hf);  return true;  }  catch (const char\* em)  {  cout << em << endl;  return false;  }  }  BOOL delRowFileTxt(LPWSTR fileName, DWORD row)  {  char filepath[20];  filepath[19] = '\0';  wcstombs(filepath, fileName, 20);  cout << "\n====== Delete row: " << row << "\n\n";  try  {  HANDLE hf = CreateFile(fileName, GENERIC\_READ, NULL, NULL, OPEN\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, NULL);  if (hf == INVALID\_HANDLE\_VALUE)  {  CloseHandle(hf);  throw "[ERROR] Create or open file failed";  }  DWORD n = NULL;  char buf[1024];  BOOL b;  ZeroMemory(buf, sizeof(buf));  b = ReadFile(hf, &buf, sizeof(buf), &n, NULL);  if (!b)  {  CloseHandle(hf);  throw ("[ERROR] Read file unsuccessful");  }  cout << "\t\tBEFORE:\n";  cout << buf << endl;  CloseHandle(hf);  HANDLE hAppend = CreateFile(fileName, GENERIC\_WRITE, NULL, NULL, OPEN\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, NULL);  char editedBuf[1024];  ZeroMemory(editedBuf, sizeof(editedBuf));  int line = 1;  int j = 0;  bool rowFound = false;  for (int i = 0; i < n; i++)  {  if (line == row)  rowFound = true;  else  {  editedBuf[j] = buf[i];  j++;  }  if (buf[i] == '\n')  line++;  }  if (!rowFound)  {  CloseHandle(hAppend);  throw ("[ERROR] Can't find this row.\n");  }  b = WriteFile(hAppend, editedBuf, n, &n, NULL);  if (!b)  {  CloseHandle(hAppend);  throw ("[ERROR] Write file unsuccessful\n");  }  cout << "\n\t\tAFTER:\n";  cout << editedBuf << endl;  CloseHandle(hAppend);  cout << "\n==========================================\n";  return true;  }  catch (const char\* em)  {  cout << em << " \n";  cout << "==========================================\n";  return false;  }  }  int main()  {  setlocale(LC\_ALL, "ru");  LPWSTR file = (LPWSTR)FILE\_PATH;  delRowFileTxt(file, 1);  delRowFileTxt(file, 3);  delRowFileTxt(file, 8);  delRowFileTxt(file, 10);  printFileText(file);  return 0;  } |
| Скриншот запуска |  |

Задание 3. Разработайте приложение **OS09\_03.**

Приложение **OS09\_03** вызывает функцию **insRowFileTxt**, имеющую следующий прототип.



Функция применяется к файлу **OS09\_01.txt** (п.2) и вызывается последовательно 4 раза, с row = 0,-1,5,7. Результат выполнения продемонстрируйте с помощью функции **printFileTxt** (п.6).

Продемонстрируйте работоспособность приложения **OS09\_03**.

|  |  |
| --- | --- |
| Код программы OS09\_03 | #define \_CRT\_SECURE\_NO\_WARNINGS  #include <Windows.h>  #include <iostream>  #include <fileapi.h>  using namespace std;  #define FILE\_PATH L"E:/3course/5sem/OS/Lab9/OS09\_01.txt"  #define READ\_BYTES 1000  BOOL printFileText(LPWSTR fileName)  {  try  {  cout << "\n\n\t====== RESULT ======\n";  HANDLE hf = CreateFile(fileName, GENERIC\_READ, NULL, NULL, OPEN\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, NULL);  if (hf == INVALID\_HANDLE\_VALUE) throw "[ERROR] Create or open file failed.";  DWORD n = NULL;  char buf[1024];  ZeroMemory(buf, sizeof(buf));  BOOL b = ReadFile(hf, &buf, READ\_BYTES, &n, NULL);  if (!b) throw "[ERROR] Read file failed";  cout << buf << endl;  CloseHandle(hf);  return true;  }  catch (const char\* em)  {  cout << "[ERROR] " << em << endl;  return false;  }  }  BOOL insRowFileTxt(LPWSTR fileName, LPWSTR str, DWORD row)  {  char filepath[20];  wcstombs(filepath, fileName, 20);  char stringToInsert[50];  wcstombs(stringToInsert, str, 50);  cout << "\n====== Insert row: " << row << "\n\n";  try  {  HANDLE hf = CreateFile(fileName, GENERIC\_READ, NULL, NULL, OPEN\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, NULL);  if (hf == INVALID\_HANDLE\_VALUE)  {  CloseHandle(hf);  throw "[ERROR] Create or open file failed";  }  DWORD n = NULL;  char buf[1024];  BOOL b;  ZeroMemory(buf, sizeof(buf));  b = ReadFile(hf, &buf, sizeof(buf), &n, NULL);  if (!b)  {  CloseHandle(hf);  throw ("[ERROR] Read file unsuccessful");  }  if (!b)  {  cout << "Read file unsuccessful.\n";  CloseHandle(hf);  return false;  }  cout << "\t\tBEFORE:\n";  cout << buf << endl;  CloseHandle(hf);  HANDLE hAppend = CreateFile(fileName, GENERIC\_WRITE, NULL, NULL, OPEN\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, NULL);  char editedBuf[1024];  ZeroMemory(editedBuf, sizeof(editedBuf));  int line = 1;  int j = 0;  int k = 0;  for (int i = 0; i < n; i++)  {  if (line == row)  {  for (int k = 0; k < sizeof(stringToInsert); k++)  {  editedBuf[j] = str[k];  j++;  if (str[k + 1] == '\0')  {  editedBuf[j] = '\r';  j++;  editedBuf[j] = '\n';  j++;  row = 0;  break;  }  }  i--;  }  else  {  editedBuf[j] = buf[i];  j++;  }  if (buf[i] == '\n')  line++;  if (buf[i + 1] == '\0' && row == -1)  {  for (int k = 0; k < sizeof(stringToInsert); k++)  {  editedBuf[j] = str[k];  j++;  if (str[k + 1] == '\0')  {  editedBuf[j] = '\r';  j++;  editedBuf[j] = '\n';  j++;  row = 0;  break;  }  }  }  }  b = WriteFile(hAppend, editedBuf, j, &n, NULL);  if (!b)  {  CloseHandle(hAppend);  throw ("[ERROR] Write file unsuccessful\n");  }  cout << "\t\tAFTER:\n";  cout << editedBuf << endl;  CloseHandle(hAppend);  cout << "\n==========================================\n";  return true;  }  catch (const char\* em)  {  cout << em << " \n";  cout << "==========================================\n";  return false;  }  }  int main()  {  setlocale(0, "ru");  LPWSTR file = (LPWSTR)FILE\_PATH;  char str[] = "NEW STROKE IN FILE :)";  wchar\_t wStr[50];  mbstowcs(wStr, str, strlen(str) + 1);  LPWSTR strToIns = wStr;  insRowFileTxt(file, strToIns, 1);  insRowFileTxt(file, strToIns, -1);  insRowFileTxt(file, strToIns, 5);  insRowFileTxt(file, strToIns, 7);  printFileText(file);  } |
| Скриншот запуска |  |

Задание 4. Разработайте приложение **OS09\_04.**

Приложение **OS09\_04** вызывает функцию **printWathRowFileTxt**, имеющую следующий прототип.



Функция применяется к файлу **OS09\_01.txt** (п.2), следит (***используйте функцию наблюдения за файлами в каталоге***) за изменением количества строк в файле в течении mlsec и выводит информацию об изменениях в стандартный поток вывода.

Продемонстрируйте работоспособность приложения **OS09\_04** совместно с приложениями **OS09\_03** и **OS09\_04**.

|  |  |
| --- | --- |
| Код программы OS09\_04 | #define \_CRT\_NON\_CONFORMING\_WCSTOK  #define \_CRT\_SECURE\_NO\_WARNINGS  #include <iostream>  #include <cstdlib>  #include "Windows.h"  using namespace std;  #define FILE\_PATH L"E:/3course/5sem/OS/Lab9/OS09\_01.txt"  #define DIR\_PATH L"E:/3course/5sem/OS/Lab9"  int rowC = 0;  BOOL printWatchRowFileTxt(LPWSTR FileName, DWORD mlsec)  {  PLARGE\_INTEGER fileSize = new LARGE\_INTEGER();  LPWSTR path = (LPWSTR)DIR\_PATH;  char\* cFileName = new char[wcslen(FileName) \* sizeof(char) + 1];  wcstombs(cFileName, FileName, strlen(cFileName));  int i = 0;  int lenName = 0;  try  {  HANDLE notif = FindFirstChangeNotification(path, false, FILE\_NOTIFY\_CHANGE\_LAST\_WRITE);  DWORD err = GetLastError();  clock\_t t1 = clock();  clock\_t t2 = clock();  DWORD dwWaitStatus;  cout << "\n==================================================\n";  printf("\tStarted filewatch (timestamp %d)", t1);  while (true)  {  dwWaitStatus = WaitForSingleObject(notif, mlsec);  switch (dwWaitStatus)  {  case WAIT\_OBJECT\_0:  {  if (FindNextChangeNotification(notif) == FALSE)  break;  else  {  int position = 0;  int rowCount = 0;  HANDLE of = CreateFile(  FileName,  GENERIC\_READ,  FILE\_SHARE\_READ,  NULL,  OPEN\_ALWAYS,  FILE\_ATTRIBUTE\_NORMAL,  NULL);  if (of == INVALID\_HANDLE\_VALUE)  {  CloseHandle(of);  throw "[ERROR] Open file failed";  }  if (GetFileSizeEx(of, fileSize))  {  char\* buf = new char[(fileSize->QuadPart + 1) \* sizeof(char)];  ZeroMemory(buf, (fileSize->QuadPart + 1) \* sizeof(char));  DWORD n = NULL;  if (ReadFile(of, buf, fileSize->QuadPart, &n, NULL))  while (buf[position++] != '\0')  if (buf[position] == '\n')  rowCount++;  }  if (rowC != rowCount)  {  printf("\n\t\t Rows: %d", rowCount);  rowC = rowCount;  }  CloseHandle(of);  }  }  }  t2 = clock();  if (t2 - t1 > mlsec)  break;  }  CloseHandle(notif);  printf("\n\tEnded filewatch (timestamp %d)\n", t2);  }  catch (const char\* err)  {  cout << "[ERROR] " << err << "\n";  return false;  }  return true;  }  int main()  {  setlocale(LC\_ALL, "ru");  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  LPWSTR fileName = (LPWSTR)FILE\_PATH;  printWatchRowFileTxt(fileName, 30000);  } |
| Скриншот запуска |  |

Задание 5. Создайте текстовый файл **OS09\_05.txt,** аналогичный файлу **OS09\_01.txt** (п.2).

Разработайте приложение **OS09\_05,** подсчитывающее количество строк и выводящее это значение в стандартный поток.

Продемонстрируйте работоспособность приложения **OS09\_05**.

|  |  |
| --- | --- |
| Код программы OS09\_05 | #include <stdio.h>  #include <stdlib.h>  #include <sys/types.h>  #include <time.h>  #include <unistd.h>  #include <sys/stat.h>  #include <fcntl.h>  #include <locale.h>  int main()  {  char \*locale = setlocale(LC\_ALL, "ru");  char c;  int rowNumber = 0;  int in = open("./OS09\_05.txt", O\_RDONLY);  int bytesRead = 0;  int isNewLine = 1; // Флаг для отслеживания новой строки  while (read(in, &c, 1) == 1) {  bytesRead++;  if (c == '\n') {  rowNumber++;  isNewLine = 1;  } else if (c != '\n' && isNewLine) {  isNewLine = 0;  }  }  // Добавляем проверку последнего символа в файле  if (bytesRead > 0 && !isNewLine) {  rowNumber++;  }  printf("Number of lines: %d\n", rowNumber);  return 0;  } |
| Скриншот запуска |  |

Задание 6. Разработайте приложение **OS09\_06,** принимающее 1 параметр, принимающее целочисленное числовое значение.

Если числовое значение принимает нечетное значение, то приложение создает новый файл **OS09\_06\_1.txt**, содержащий только нечетные строки из файла **OS09\_05.txt**.

Если числовое значение принимает четное значение, то приложение создает новый файл **OS09\_06\_2.txt**, содержащий только четные строки из файла **OS09\_05.txt**.

Продемонстрируйте работоспособность приложения **OS09\_06**.

|  |  |
| --- | --- |
| Код программы OS09\_06 | #include <stdio.h>  #include <stdlib.h>  #include <sys/types.h>  #include <time.h>  #include <unistd.h>  #include <sys/stat.h>  #include <fcntl.h>  #include <locale.h>  #include <stdbool.h>  int main(int argc, char \*argv[])  {  char \*locale = setlocale(LC\_ALL, "");  char c;  int rowNumber = 0;  int in = open("./OS09\_05.txt", O\_RDONLY);  int out;  int num;  bool yaBool = 0;    if (argc != 2) {  printf("Please, provide an integer parameter.\n");  exit(1);  }  num = atoi(argv[1]);    if (num % 2 != 0)  out = open("./OS09\_06\_1.txt", O\_CREAT | O\_WRONLY, 0777);  else {  out = open("./OS09\_06\_2.txt", O\_CREAT | O\_WRONLY, 0777);  yaBool = 1;  }    while (read(in, &c, 1) == 1) {  if (!yaBool)  write(out, &c, 1);  if (c == '\n')  yaBool = !yaBool;  }  close(out);  close(in);  return 0;  } |
| Скриншот запуска |  |

Задание 7. Разработайте приложение **OS09\_07,** демонстрирующее возможности функции **lseek.**

Поясните назначение и принцип работы функции **lseek**.

|  |  |
| --- | --- |
| Код программы OS09\_07 | #include <stdio.h>  #include <unistd.h>  #include <fcntl.h>  int main() {  int filedesc = open("OS09\_05.txt", O\_RDONLY);  if (filedesc < 0) {  return -1;  }  // Перемещаемся на 10 байт от начала файла  lseek(filedesc, 10, SEEK\_SET);  char buffer[11];  read(filedesc, buffer, 10);  buffer[10] = '\0';  printf("Символы 11-20 файла: %s\n", buffer);  close(filedesc);  return 0;  } |
| Скриншот запуска |  |