**Тема:     Работа с файловой системой в среде WIN32. Система оповещений.**

|  |
| --- |
| **Лабораторная работа. Работа с файловой системой в среде WIN32. Система оповещений.** |
|  |  |
| **Цель работы**: научиться создавать приложения в среде WIN32, выполняющие работу с файловой системой на уровне системных вызовов.    **Темы для предварительной проработки**: файловые системы, графический интерфейс пользователя, структура оконного приложения в среде WIN32, интерфейс прикладного программирования API WIN32.    **Теоретические сведения**    Для отслеживания изменений в файловой системе в операционной системе Windows используется система извещений. Для начала отслеживания изменений в каталогах необходимо создать дескриптор контроля изменений и установить начальные условия фильтрации извещений об изменениях: **FindFirstChangeNotification**.  В этой функции можно задать каталог для отслеживания, необходимость анализа подкаталогов и виды изменений, которые должны отслеживаться. Возвращаемое значение – дескриптор обнаруженного объекта контроля изменений, который используется при контроле изменений с помощью функций **WaitForSingleObject** или **WaitForMultipleObject**.  В случае необходимости повторного ожидания извещения используется функция **FindNextChangeNotification**.  Для прекращения отслеживания изменений для заданного дескриптора используется функция **FindCloseChangeNotification**.  Пример. Вывод содержимого каталога в случае изменения имен файлов в нем.    #include <windows.h>  #include <stdio.h>    HANDLE hFind;  HANDLE g\_hndl;  WIN32\_FIND\_DATA fd;  BOOL  bRet = TRUE;    int main()  {  //инициализируем дескриптор уведомлений  g\_hndl = FindFirstChangeNotification("D:\\TEMP",            FALSE, FILE\_NOTIFY\_CHANGE\_FILE\_NAME);  if(g\_hndl == INVALID\_HANDLE\_VALUE)     {     printf("Invalid Handle\n");     }  //ожидание появления уведомлений  if(WaitForSingleObject(g\_hndl, INFINITE) != WAIT\_FAILED)     {     printf("File Name Changed!");     //Вывод списка файлов в директории     hFind = FindFirstFile("D:\\TEMP\\\*.\*", &fd );     while ( hFind != INVALID\_HANDLE\_VALUE && bRet)        {        if ((fd.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY) == 0)           {           printf("%s\n",fd.cFileName);           }        bRet = FindNextFile( hFind, &fd );        }     FindClose( hFind );     //Эта часть не нужна в данной программе - приведена для примера     /\*\*/ if(FindNextChangeNotification(g\_hndl) == FALSE)     /\*\*/    {     /\*\*/    printf("Find Next failed\n");     /\*\*/    }     }  FindCloseChangeNotification(g\_hndl);  return 0;  }    **Постановка задачи**: разработать приложение в среде программирования Visual C, которое отслеживает изменения в файловой системе и выводит сообщение, в соответствии с заданным вариантом. Для хранения промежуточных результатов можно использовать средства WIN32 API для работы с памятью.    **Порядок выполнения работы**    1. Ознакомиться с постановкой задачи и исходными данными. В соответствии с номером по журналу определить вариант задачи.  2. Согласно рекомендациям, приведенным в исходных данных, сконструировать структуру программы.  3. Составить тексты программ и утвердить их у преподавателя.  4. Пункты 1 - 3 должны быть выполнены предварительно до проведения данной лабораторной работы. Утверждение преподавателем текстов программ является допуском к лабораторной работе.  5. Набрать текст программы.  6. Выполнить компиляцию программы.  7. Провести анализ и исправление обнаруженных синтаксических ошибок в тексте программы и повторить пункты 6. и 7. При устранении всех синтаксических ошибок перейти к выполнению пункта 8.  8. Получить решение и, в случае обнаружения логических ошибок, описать и устранить их. Продемонстрировать преподавателю окончательный вариант программы и ее работу.    **Содержание отчета**    1. Тема лабораторной работы.  2. Цель работы.  3. Индивидуальное задание.  4. Метод и алгоритм решения задачи.  5. Текст программы.  6. Результаты работы программы.  7. Выводы по работе.    **Индивидуальные задания**    15. Имя каталога вводится с клавиатуры. Подкаталоги анализируются. Выводить имена удаляемых фалов и время их удаления. | |