

**操作系统课程设计**

|  |  |
| --- | --- |
| **实验、进程控制** | **Experiment 5, Copy Files** |

学院：计算机学院

专业：计算机科学与技术

学生姓名：夏奇拉

学号：1820171025

班级：07111705

Table of Contents

[Purpose 3](#__RefHeading___Toc1766_1221548062)

[Problem Discussion 3](#__RefHeading___Toc1744_1221548062)

[Execution [Windows] 5](#__RefHeading___Toc1768_1221548062)

[Results and Analysis [Windows] 9](#__RefHeading___Toc1770_1221548062)

[Execution [Linux] 10](#__RefHeading___Toc604_992951594)

[Results and Analysis [Linux] 15](#__RefHeading___Toc1770_12215480621)

[Reference: 16](#__RefHeading___Toc606_992951594)

# Purpose

Complete a directory copy command mycp, including the files and subdirectories in the directory, and the running results are as follows:

|  |
| --- |
| [beta@bugs.com](mailto:beta@bugs.com) [~/]# ls -l sem |
| total 56  drwxr-xr-x 3 beta beta 4096 Dec 19 02:53 ./  drwxr-xr-x 8 beta beta 4096 Nov 27 08:49 ../   * rw-r--r-- 1 beta beta 128 Nov 27 09:31 Makefile * rwxr-xr-x 1 beta beta 5705 Nov 27 08:50 consumer\* * rw-r--r-- 1 beta beta 349 Nov 27 09:30 consumer.c   drwxr-xr-x 2 beta beta 4096 Dec 19 02:53 subdir/ |
| [beta@bugs.com](mailto:beta@bugs.com) [~/]# mycp sem target |
| [beta@bugs.com](mailto:beta@bugs.com) [~/]# ls -l target |
| total 56  drwxr-xr-x 3 beta beta 4096 Dec 19 02:53 ./  drwxr-xr-x 8 beta beta 4096 Nov 27 08:49 ../   * rw-r--r-- 1 beta beta 128 Nov 27 09:31 Makefile * rwxr-xr-x 1 beta beta 5705 Nov 27 08:50 consumer\* * rw-r--r-- 1 beta beta 349 Nov 27 09:30 consumer.c   drwxr-xr-x 2 beta beta 4096 Dec 19 02:53 subdir/ |

Description:

Linux: creat, read, write and other system calls

Windows: CreateFile(), ReadFile(), WriteFile(), CloseHandle() and other functions

Requires the ability to copy nested folders (there are folders in the folder), link files

# Problem Discussion

The problem asks that we

1. print the contents of the source directory in the command line
2. run the mycp command to copy the contents of the source directory to the target directory
3. print the contents of the target directory in the command line

# Execution [Windows]

<https://www.installsetupconfig.com/win32programming/windowsfileapis4_22.html>

## Results and Analysis [Windows]

# Execution [Linux]

**Step 1: List the files in the source directory**

In the terminal, list the files and folders in the source directory using the “ls -l” command which allows us to list files and directories in long listing format, one per line. The line shows the file or directory permission, owner and group name, file size, created/modified date and time, file/folder name.

|  |
| --- |
| $ ls -l source |

**Step 2: Create the mycp program**

In the current directory, create the mycp.c file by running the following command:

|  |
| --- |
| $ nano mycp.c |

In the file we write the following code:

|  |
| --- |
| // BIT 100073007 Operating Systems Course Lab 5: Copy File  #include <unistd.h> // read(), write() - read/write from/to a file  #include <string.h> // strcmp() - compare two strings  #include <stdio.h>  #include <dirent.h> // opendir(), readdir()  #include <sys/types.h> // opendir()  #include <sys/stat.h> // lstat(), chmod()  #include <fcntl.h> // open(), creat() = open and possibly create a  #include <utime.h> // utime() - change file last access and mod times  /\* buffer size is set to 4096 because thats the tyical sector size on a disk.  increasing the buffer size beyond 4096 has little positive effect. \*/  const int BUFFER = 4096;  /\* maximum path length limitation \*/  const int MAX\_PATH = 260;  void searchDirectory(char sourcePath[MAX\_PATH], char targetPath[MAX\_PATH]);  int main(int argc, char const \*argv[])  {  if (argc != 3)  {  printf("usage: ./mycp <source-directory> <targetDirectory>/n");  return -1;  }  /\* taking the contents of argv[] and adding them to character array source and target for clarity and ease\*/  char source[MAX\_PATH], target[MAX\_PATH];  /\* sprintf stands for “String print”. Instead of printing on console, it store output on char buffer which are specified in sprintf. \*/  sprintf(source, "%s", argv[1]);  sprintf(target, "%s", argv[2]);  /\*  int lstat(const char \*pathname, struct stat \*fileAttributeBuffer);  const char \*pathname // The path of the file or directory whose attributes need to be extracted  struct stat \*fileAttributeBuffer // The acquired attribute storage buffer  return int // Returns 0 successfully, otherwise -1  \*/  struct stat fileAttributeBuffer;  struct utimbuf timebuf;  // check if the source file exists and open the directory  DIR \*sourceDir;  DIR \*targetDir;  if ((sourceDir = opendir(source)) == NULL) // opendir - open a directory  {  printf("Error: Source directory not found/n");  }  // Create target directory (if there is no target directory )  if ((targetDir = opendir(target)) == NULL)  {  // Store the attributes of the source file into fileAttributeBuffer  stat(source, &fileAttributeBuffer); // stat() - get file attributes  // Create target file directory  // st\_mode = File type and mode  mkdir(target, fileAttributeBuffer.st\_mode); // mkdir() - create a directory  // Record the time of file access and modification  timebuf.actime = fileAttributeBuffer.st\_atime; // access time = time of last access  timebuf.modtime = fileAttributeBuffer.st\_mtime; // modification time = time of last modification  // Record the access and modification time of the target file  utime(target, &timebuf);  }  searchDirectory(source, target);  return 0;  }  void searchDirectory(char sourcePath[MAX\_PATH], char targetPath[MAX\_PATH])  {  char source[MAX\_PATH];  char target[MAX\_PATH];  strcpy(source, sourcePath); // strcpy - copy a string  strcpy(target, targetPath);  DIR \*sourceDir = opendir(source);  DIR \*targetDir = opendir(target);  /\* the dirent structure  struct dirent {  ino\_t d\_ino; // Inode number  off\_t d\_off; // Not an offset; see below  unsigned short d\_reclen; // Length of this record  unsigned char d\_type; // Type of file; not supported by all filesystem types  char d\_name[256]; // Null-terminated filename  };  \*/  struct dirent \*entry;  /\* The stat structure  struct stat {  struct timespec st\_atim; // Time of last access  struct timespec st\_mtim; // Time of last modification  struct timespec st\_ctim; // Time of last status change  };  \*/  struct stat fileAttributeBuffer;  /\* The utimbuf structure is:  struct utimbuf {  time\_t actime; // access time  time\_t modtime; // modification time  };  \*/  struct utimbuf timebuf;  char buffer[BUFFER];  while ((entry = readdir(sourceDir)) != NULL)  {  lstat(entry->d\_name, &fileAttributeBuffer);  if (strcmp(entry->d\_name, ".") == 0 || strcmp(entry->d\_name, "..") == 0)  {  continue;  }  else if (S\_ISDIR(fileAttributeBuffer.st\_mode)) // directory  {  // change path of source and target  strcat(source, "/");  strcat(source, entry->d\_name);  strcat(target, "/");  strcat(target, entry->d\_name);  stat(source, &fileAttributeBuffer);  mkdir(target, fileAttributeBuffer.st\_mode);  timebuf.actime = fileAttributeBuffer.st\_atime;  timebuf.modtime = fileAttributeBuffer.st\_mtime;  // int utime(const char \*filename, const struct utimbuf \*times);  utime(target, &timebuf); // change file last access and modification times  // call search functions  searchDirectory(source, target);  strcpy(source, sourcePath); // strcpy - copy a string  strcpy(target, targetPath);  }  else if (S\_ISREG(fileAttributeBuffer.st\_mode)) // normal file  {  // change path of source and target  strcat(source, "/");  strcat(source, entry->d\_name);  strcat(target, "/");  strcat(target, entry->d\_name);  // start copy file  int sourceData = open(source, O\_RDONLY);  if (sourceData == -1)  {  printf("Error: Failed to open file./n");  }  stat(source, &fileAttributeBuffer);  int targetData = creat(targetPath, fileAttributeBuffer.st\_mode);  if(targetData == -1)  {  printf("Error: File creation failed./n");  }  int wordbit;  while ((wordbit = read(sourceData, buffer, BUFFER)) > 0)  {  if (write(targetData, buffer, wordbit) != wordbit)  {  printf("Error: Write file failed./n");  }  timebuf.actime = fileAttributeBuffer.st\_atime;  timebuf.modtime = fileAttributeBuffer.st\_mtime;  close(sourceData);  close(targetData);  // end of copy file  strcpy(source, sourcePath);  strcpy(target, targetPath);  }  }  }  } |

**About the code:**

*Line 19*: The code accepts command line arguments through the declaration of main which accepts two arguments: the number of command line arguments and a full list of all the command line arguments.

The if statement checks that the mycp command is being used correctly with 3 arguments.

*Line 30:* sprintf which stands for String print, stores the output on a char buffer instead of printing to the console. In this case sprintf stores the contents of argv[] to source & target which are of type char. This is being done for clarity in the code.

*Line 47*: opens the source directory using opendir(). Opendir() exists in the sys/types.h library and opens a directory stream corresponding to the directory name, and returns a pointer to the directory stream.

*Line 53*: creates the target directory using opendir() if it doesn’t exist. The new directory will be a copy of the source directory. Stat() retrieves the attributes of the source directory and stores the in the statbuf structure which was declared on *line 142*. This information is used to make the new target directory. These attributes include the file type and mode, the access time and the modification time. Utime() is used to change the file last access and modification times of the target directory

*Line 70*: calls the searchDirectory() function and passes character arrays source and target to is.

*Line 75*; declares the searchDirectory() function.

*Line 80*: copies the contents of character array sourcePath which is an argument of searchDirectory() and pastes it in character array source.

*Line 83*: opens the source directory.

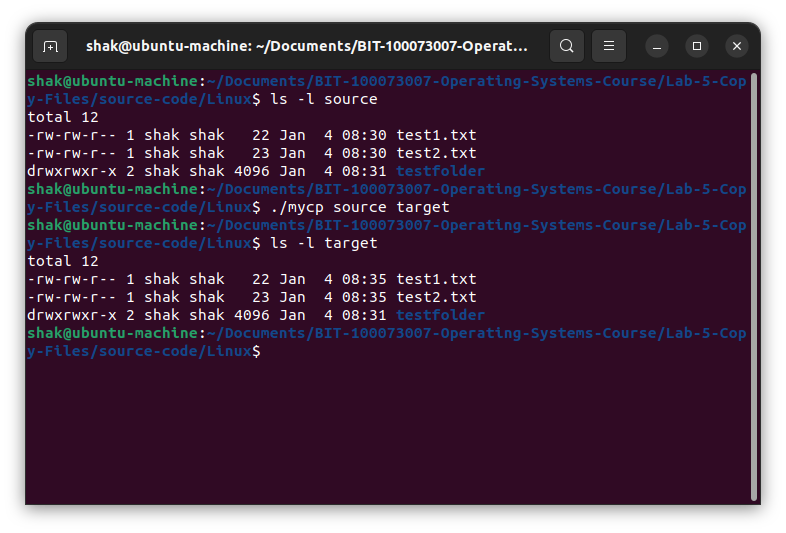
*Line 116*: reads the source directory using readdir(). The readdir() function returns a pointer to a dirent structure representing the next directory entry in the directory stream pointed to by dirp. It returns NULL on reaching the end of the directory stream or if an error occurred.

*Line 119*: uses the operator -> which is used to access the data elements of a structure that a pointer variable refers to. The syntax is: *(pointer variable) -> (variable) = value;*

*Line 123:* copies the source directory to the target directory attributes

*Line 146*: copies the files of the source directory to the target directory

## Results and Analysis [Linux]



# Reference:

* <https://learn.microsoft.com/en-us/windows/win32/api/sysinfoapi/nf-sysinfoapi-getsysteminfo>
* <https://learn.microsoft.com/en-us/windows/win32/api/sysinfoapi/ns-sysinfoapi-system_info>
* <https://learn.microsoft.com/en-gb/windows/win32/sysinfo/getting-hardware-information?redirectedfrom=MSDN>
* <https://www.installsetupconfig.com/win32programming/windowsvolumeapis1_6.html>
* <https://learn.microsoft.com/en-us/windows/win32/winprog64/virtual-address-space>
* <https://learn.microsoft.com/en-us/windows/win32/memory/memory-management>
* <https://www.tutorialspoint.com/operating_system/os_memory_management.htm>#
* [How Random Access Memory (RAM) affects performance | Dell US](https://www.dell.com/support/kbdoc/en-us/000129805/how-random-access-memory-ram-affects-performance)
* [Physical and Virtual Memory in Windows 10 - Microsoft Community](https://answers.microsoft.com/en-us/windows/forum/all/physical-and-virtual-memory-in-windows-10/e36fb5bc-9ac8-49af-951c-e7d39b979938)
* [Page State - Win32 apps | Microsoft Learn](https://learn.microsoft.com/en-us/windows/win32/Memory/page-state)
* [MEMORY\_BASIC\_INFORMATION (winnt.h) - Win32 apps | Microsoft Learn](https://learn.microsoft.com/en-us/windows/win32/api/winnt/ns-winnt-memory_basic_information)
* <https://linuxhint.com/what-does-ls-l-command-do-in-linux/>
* <https://linuxhint.com/linux_file_permissions/>
* <https://man7.org/linux/man-pages/man3/opendir.3.html>
* <https://man7.org/linux/man-pages/man2/lstat.2.html>
* <https://man7.org/linux/man-pages/man3/strcpy.3.html>
* <https://man7.org/linux/man-pages/man2/open.2.html>
* <https://man7.org/linux/man-pages/man2/write.2.html>