

Gebze Technical University
Computer Engineering
Department

CSE 344 – System Programming

Homework-5 Report

Muhammet Akkurt

1901042644

Differences and Improvements from HW4 code:

In the updated code, the barrier is defined and all threads, including the manager thread, are included and initialized.

Barrier ensures that all worker threads and the manager thread exit synchronously after completing their work. This guarantees that all threads complete at the same time and there are no synchronization issues in the process.

Cond. Variable and Barrier Usage:

Manager and worker threads perform their operations according to the state of the buffer using condition variables. Manager thread adds new files to the buffer, while worker threads remove files from the buffer.

When the operations are complete, all threads meet at the barrier and complete synchronously. This guarantees that the copying of all files is complete and all threads have finished their work.

Purpose and Details of the Assignment:

This assignment is a multi-threaded program designed to copy files from one directory to another. The program is built using POSIX threads to handle multiple file operations simultaneously.

The system architecture consists of a manager thread that populates a shared buffer with file metadata and

multiple worker threads that perform copying of files based on data retrieved from the buffer. Synchronization between these threads is achieved using mutexes and condition variables to manage access to the shared buffer and coordinate the producer-consumer relationship.

The program handles different types of files, including regular files and FIFOs. Regular files are directly copied, whereas FIFOs are recreated at the destination.

Directories are processed recursively, ensuring that all nested files and subdirectories are copied. The program replicates directory structures accurately at the destination.

Mutexes are used to ensure exclusive access to the shared buffer by one thread at a time, while condition variables manage thread synchronization, signaling threads to sleep or wake up based on the buffer's state of being empty or full.

Pseudocode:

Main Program:

```
Initialize resources (buffer, threads, barrier)  
Start timing  
Create manager thread with source and destination directories  
Create worker threads  
Wait for all threads to complete  
Stop timing  
Print performance statistics  
Clean up resources
```

Manager Thread:

```
For each file in the source directory:  
    Wait until there is space in the buffer  
    Lock the buffer  
    Add file to buffer  
    Signal worker threads that buffer is not empty  
    Unlock the buffer  
Wait at the barrier to synchronize termination with worker threads  
Indicate completion and exit
```

Worker Thread:

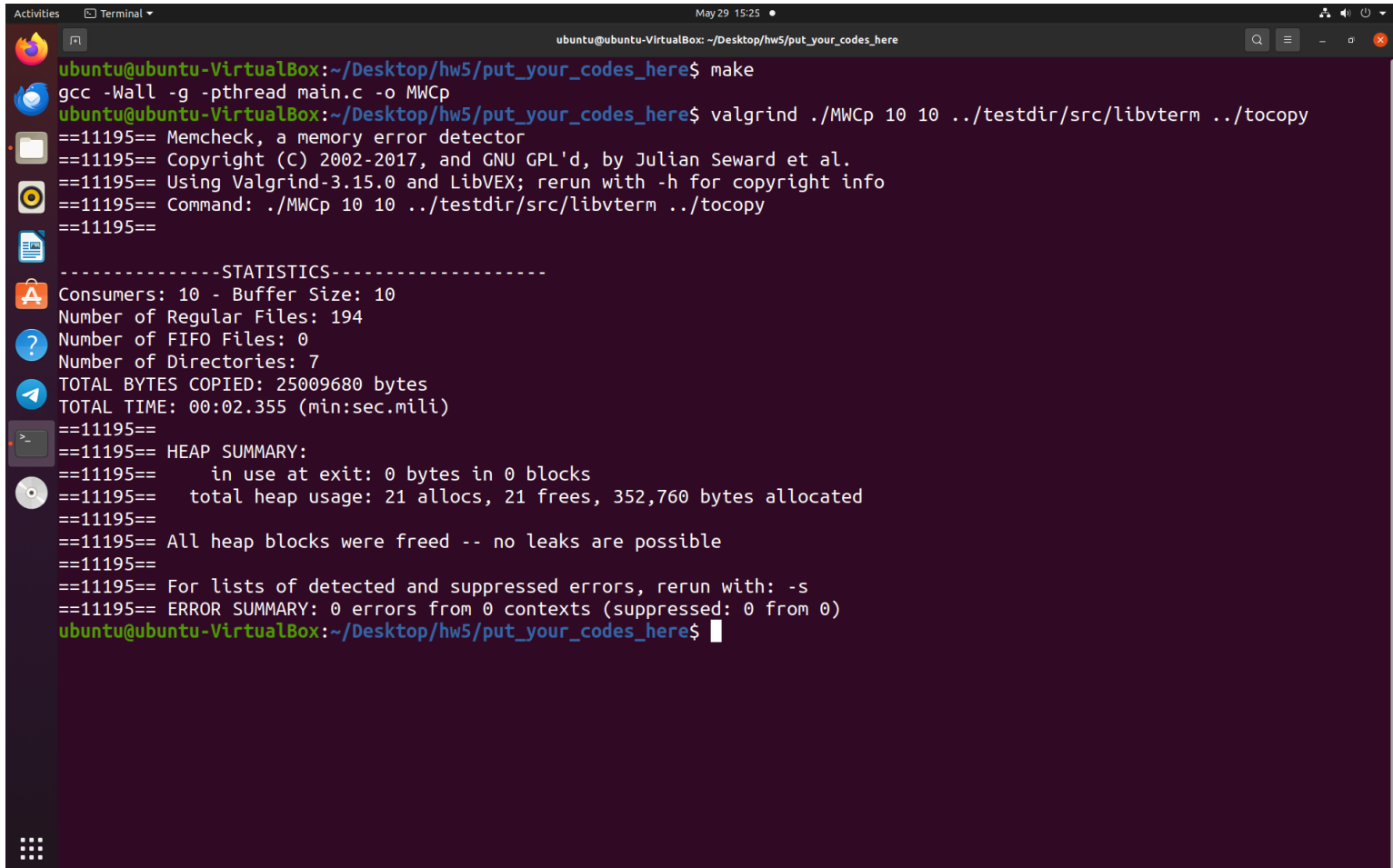
```
While not done:  
    Lock the buffer  
    Wait for buffer to be not empty  
    Retrieve file information from buffer  
    If file is a FIFO:  
        Create FIFO at destination  
    Else:  
        Copy file from source to destination  
    Update copied bytes and file count  
    Unlock the buffer  
Wait at the barrier to synchronize termination with other threads  
Exit thread
```

Signal Handling:

```
On SIGINT or SIGTERM:  
    Set termination flag  
    Signal all conditions to ensure no thread is left waiting
```

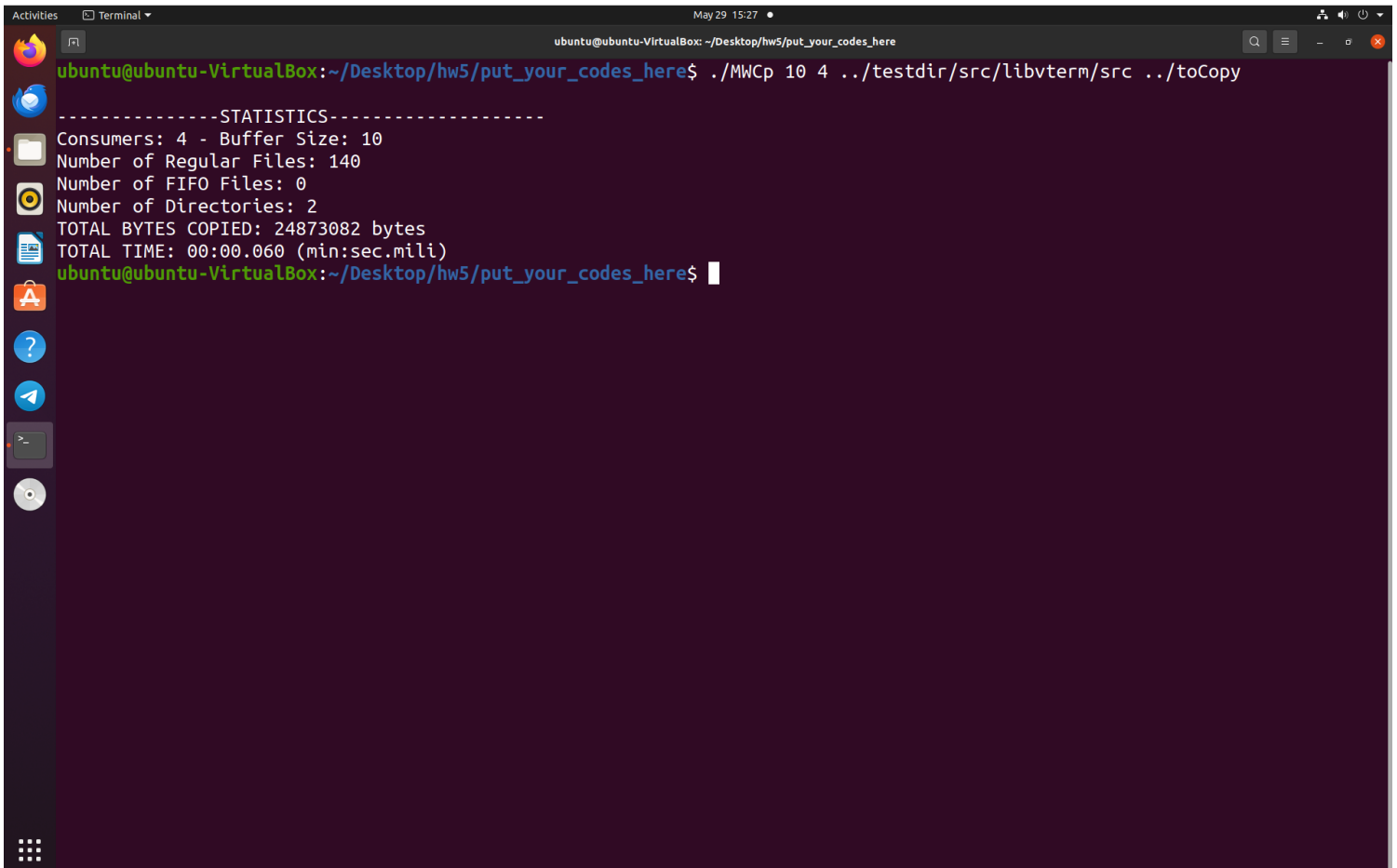
Test Scenarios:

- Test1: valgrind ./MWCp 10 10 ../testdir/src/libvterm ../tocopy



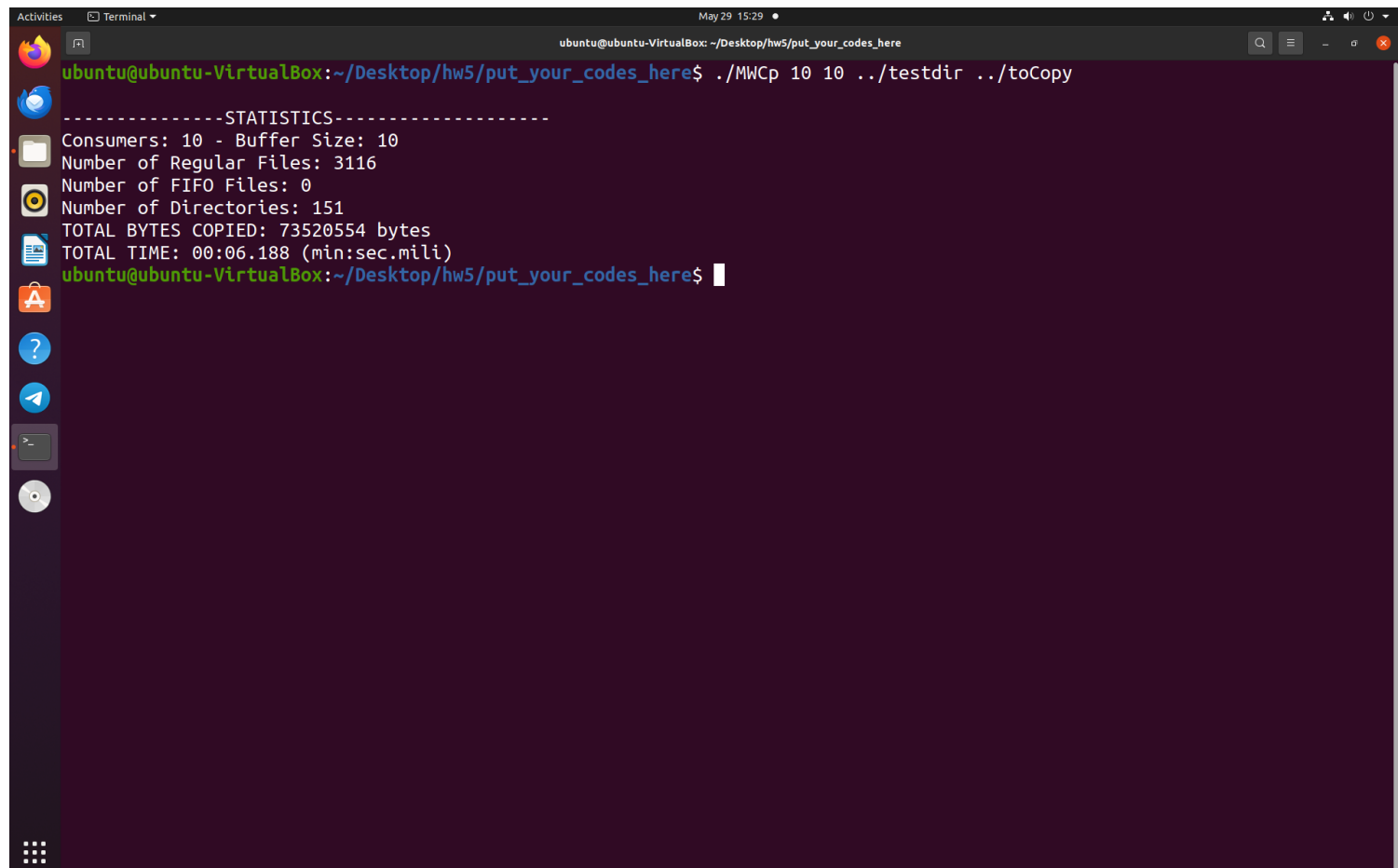
```
ubuntu@ubuntu-VirtualBox: ~/Desktop/hw5/put_your_codes_here
ubuntu@ubuntu-VirtualBox:~/Desktop/hw5/put_your_codes_here$ make
gcc -Wall -g -pthread main.c -o MWCp
ubuntu@ubuntu-VirtualBox:~/Desktop/hw5/put_your_codes_here$ valgrind ./MWCp 10 10 ../testdir/src/libvterm ../tocopy
==11195== Memcheck, a memory error detector
==11195== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==11195== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==11195== Command: ./MWCp 10 10 ../testdir/src/libvterm ../tocopy
==11195==
-----STATISTICS-----
Consumers: 10 - Buffer Size: 10
Number of Regular Files: 194
Number of FIFO Files: 0
Number of Directories: 7
TOTAL BYTES COPIED: 25009680 bytes
TOTAL TIME: 00:02.355 (min:sec.mili)
==11195==
==11195== HEAP SUMMARY:
==11195==     in use at exit: 0 bytes in 0 blocks
==11195==   total heap usage: 21 allocs, 21 frees, 352,760 bytes allocated
==11195==
==11195== All heap blocks were freed -- no leaks are possible
==11195==
==11195== For lists of detected and suppressed errors, rerun with: -s
==11195== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
ubuntu@ubuntu-VirtualBox:~/Desktop/hw5/put_your_codes_here$
```

- Test2: `./MWCp 10 4 ../testdir/src/libvterm/src ../toCopy`

A terminal window titled 'Terminal' showing the execution of the command `./MWCp 10 4 ../testdir/src/libvterm/src ../toCopy`. The output displays statistics for the copy operation. The terminal has a dark purple background and a sidebar on the left with various application icons.

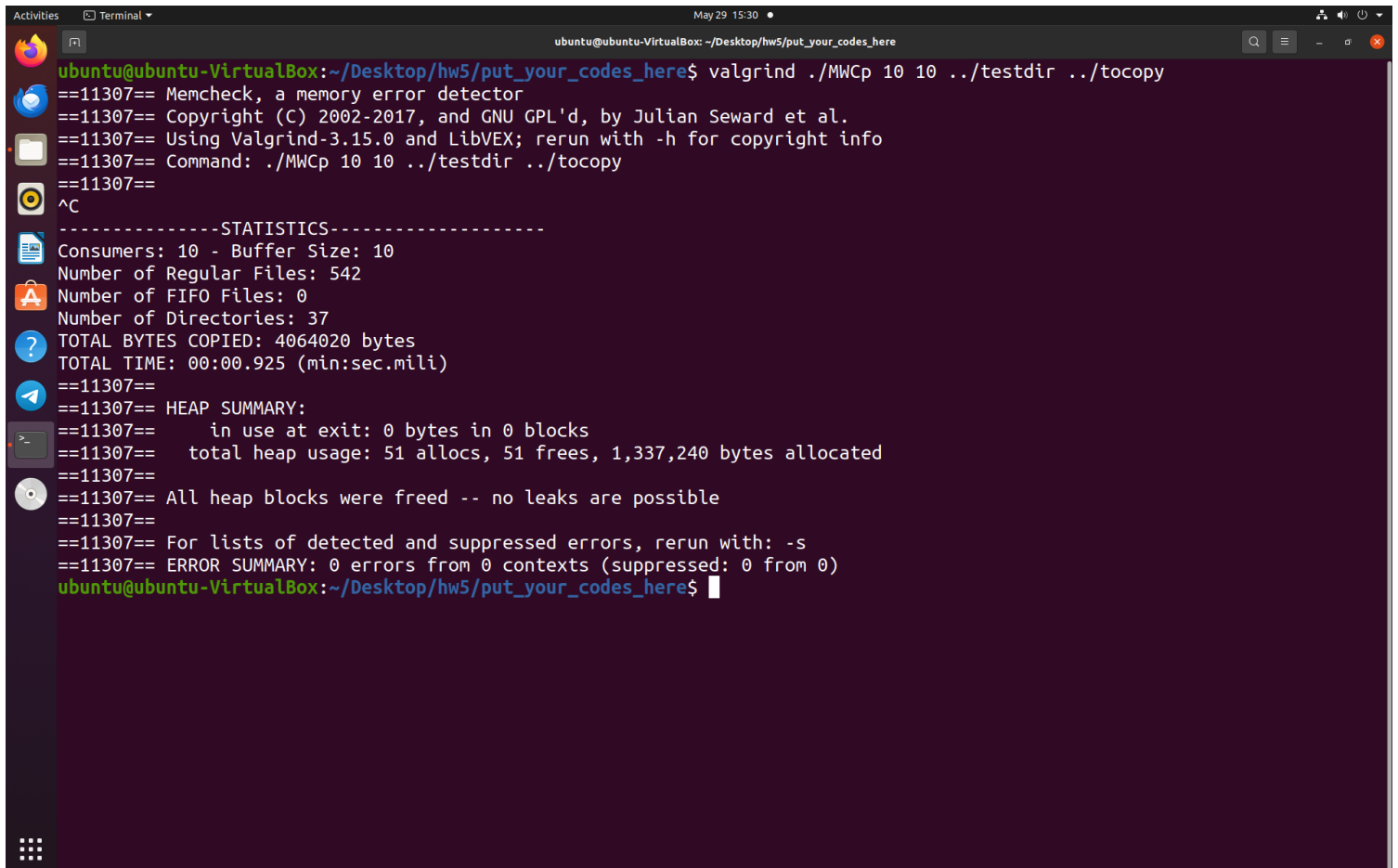
```
ubuntu@ubuntu-VirtualBox: ~/Desktop/hw5/put_your_codes_here
ubuntu@ubuntu-VirtualBox:~/Desktop/hw5/put_your_codes_here$ ./MWCp 10 4 ../testdir/src/libvterm/src ../toCopy
-----STATISTICS-----
Consumers: 4 - Buffer Size: 10
Number of Regular Files: 140
Number of FIFO Files: 0
Number of Directories: 2
TOTAL BYTES COPIED: 24873082 bytes
TOTAL TIME: 00:00.060 (min:sec.mili)
ubuntu@ubuntu-VirtualBox:~/Desktop/hw5/put_your_codes_here$
```

- Test3: `./MWCp 10 10 ../testdir ../toCopy`

A terminal window titled 'Terminal' showing the execution of the command `./MWCp 10 10 ../testdir ../toCopy`. The output displays statistics for the copy operation. The terminal has a dark purple background and a sidebar on the left with various application icons.

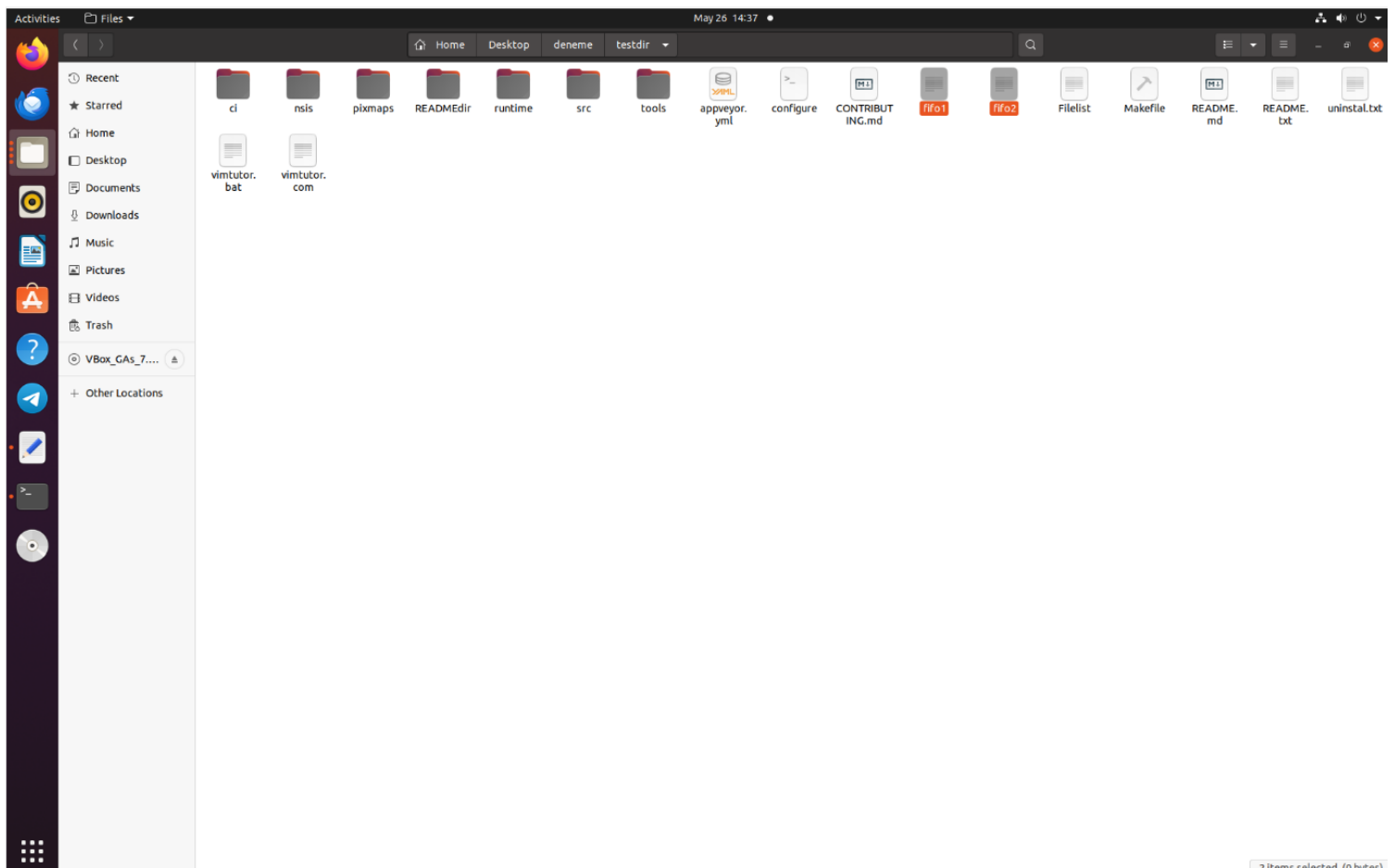
```
ubuntu@ubuntu-VirtualBox: ~/Desktop/hw5/put_your_codes_here
ubuntu@ubuntu-VirtualBox:~/Desktop/hw5/put_your_codes_here$ ./MWCp 10 10 ../testdir ../toCopy
-----STATISTICS-----
Consumers: 10 - Buffer Size: 10
Number of Regular Files: 3116
Number of FIFO Files: 0
Number of Directories: 151
TOTAL BYTES COPIED: 73520554 bytes
TOTAL TIME: 00:06.188 (min:sec.mili)
ubuntu@ubuntu-VirtualBox:~/Desktop/hw5/put_your_codes_here$
```

- Test4: valgrind ./MWCp 10 10 ../testdir ../tocopy
(Signal Handling CTRL+C)



```
ubuntu@ubuntu-VirtualBox: ~/Desktop/hw5/put_your_codes_here$ valgrind ./MWCp 10 10 ../testdir ../tocopy
==11307== Memcheck, a memory error detector
==11307== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==11307== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==11307== Command: ./MWCp 10 10 ../testdir ../tocopy
==11307==
^C
-----STATISTICS-----
Consumers: 10 - Buffer Size: 10
Number of Regular Files: 542
Number of FIFO Files: 0
Number of Directories: 37
TOTAL BYTES COPIED: 4064020 bytes
TOTAL TIME: 00:00.925 (min:sec.mili)
==11307==
==11307== HEAP SUMMARY:
==11307==    in use at exit: 0 bytes in 0 blocks
==11307==   total heap usage: 51 allocs, 51 frees, 1,337,240 bytes allocated
==11307==
==11307== All heap blocks were freed -- no leaks are possible
==11307==
==11307== For lists of detected and suppressed errors, rerun with: -s
==11307== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
ubuntu@ubuntu-VirtualBox: ~/Desktop/hw5/put_your_codes_here$
```

- add and test two FIFO files in the “testdir” folder



Test5: valgrind ./MWCp 4 4 ../testdir ../tocopy

```
Activities Terminal May 29 15:32 ubuntu@ubuntu-VirtualBox: ~/Desktop/hw5/put_your_codes_here
ubuntu@ubuntu-VirtualBox:~/Desktop/hw5/put_your_codes_here$ valgrind ./MWCp 4 4 ../testdir ../tocopy
==11332== Memcheck, a memory error detector
==11332== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==11332== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==11332== Command: ./MWCp 4 4 ../testdir ../tocopy
==11332==
-----STATISTICS-----
Consumers: 4 - Buffer Size: 4
Number of Regular Files: 3116
Number of FIFO Files: 2
Number of Directories: 151
TOTAL BYTES COPIED: 73520554 bytes
TOTAL TIME: 00:02.841 (min:sec.mili)
==11332==
==11332== HEAP SUMMARY:
==11332==    in use at exit: 0 bytes in 0 blocks
==11332==   total heap usage: 159 allocs, 159 frees, 5,024,288 bytes allocated
==11332==
==11332== All heap blocks were freed -- no leaks are possible
==11332==
==11332== For lists of detected and suppressed errors, rerun with: -s
==11332== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
ubuntu@ubuntu-VirtualBox:~/Desktop/hw5/put_your_codes_here$
```