

## Assignment – 4

### COL331 : Operating Systems

#### **PART A:**

##### **Task 1 :-**

Initially before implementing any system call the program would print the lines showing that the pintos started booting and then it identifies our program t1 and since process\_wait is not yet implemented, so it exits from the program and power off the OS.

##### **Task 2 :-**

In syscall.c, pintos provides us 4 syscall (syscall1, syscall2, syscall3, syscall4) macros which are used to generate the assembly code for trapping into the kernel to use. The syscall macros store the function\_number and the arguments are pushed onto the stack in reverse order. Ex. The open function uses the macros syscall2 which first pushes the function\_number of open onto the stack and then it pushes the argument which we give to the open function.

##### **Task 3 :-**

When we insert an infinite loop in the wait function, at that time the error message of “System call not implemented” is printed confirming the execution of the syscall\_handler.

##### **Task 4 :-**

###### **1. Syscall\_handler :**

The syscall\_handler switches the function by looking at the syscall\_number of the function which is basically the value stored at the bottom of the stack to which f->esp points. Then to take the arguments of the relevant function I made structs as given in the pdf, so in each struct the id of the system call and the argument of it are stored.

###### **2. Create :**

In this at first the char\* filename is validated using the already given function validate\_user\_addr\_range() and then file is created using filesys\_create function if the addr is valid.

###### **3. Remove :**

In this at first the char\* filename is validated using the already given function validate\_user\_addr\_range() and then file is removed using filesys\_remove function if the addr is valid.

###### **4. Open :**

For this parts and some other parts, I have implemented a struct for the file\_descriptor which basically stored the fid(file id), file, and list\_elem to use the list operations efficiently.

In the thread data structure we also included a list of file\_descriptors as well.

Now, first the char\* file is validated and then an initially file\_desc is given space using the function palloc\_get\_page() and then if we get a valid file\_desc then the file\_desc should be given the file given as the argument and while giving it the id we will see if the list of the file descriptors is empty or not; if it is empty means this is the first file in this thread hence it is given a value 3, because 0,1,2 are reserved for stdin, stdout, stderr and finally this id is returned

###### **5. Close :**

It searches in the file descriptor list for the file\_desc object having the same id as given in the arguments and then it closes the file of the object using file\_close and removes that file descriptor from the list.

###### **6. Filesize :**

It searches in the file descriptor list for the file\_desc object having the same id as given in the arguments and then finds the size of the file of that object using file\_length.

###### **7. Read :**

It validates the buffer and then checks if the fd corresponds to stdin or not, if it is yes then it reads from the stdin else it reads from the file using file\_read and return the number of characters read.

###### **8. Write :**

It validates the buffer and then checks if the fd corresponds to stdout or not, if it is yes then it writes on the stdout else it writes on the file using file\_write and return the number of characters wrote.

**9. Seek :**

It searches in the file descriptor list for the file\_desc object having the same id as given in the arguments and then seeks the position of the file of that object using file\_seek.

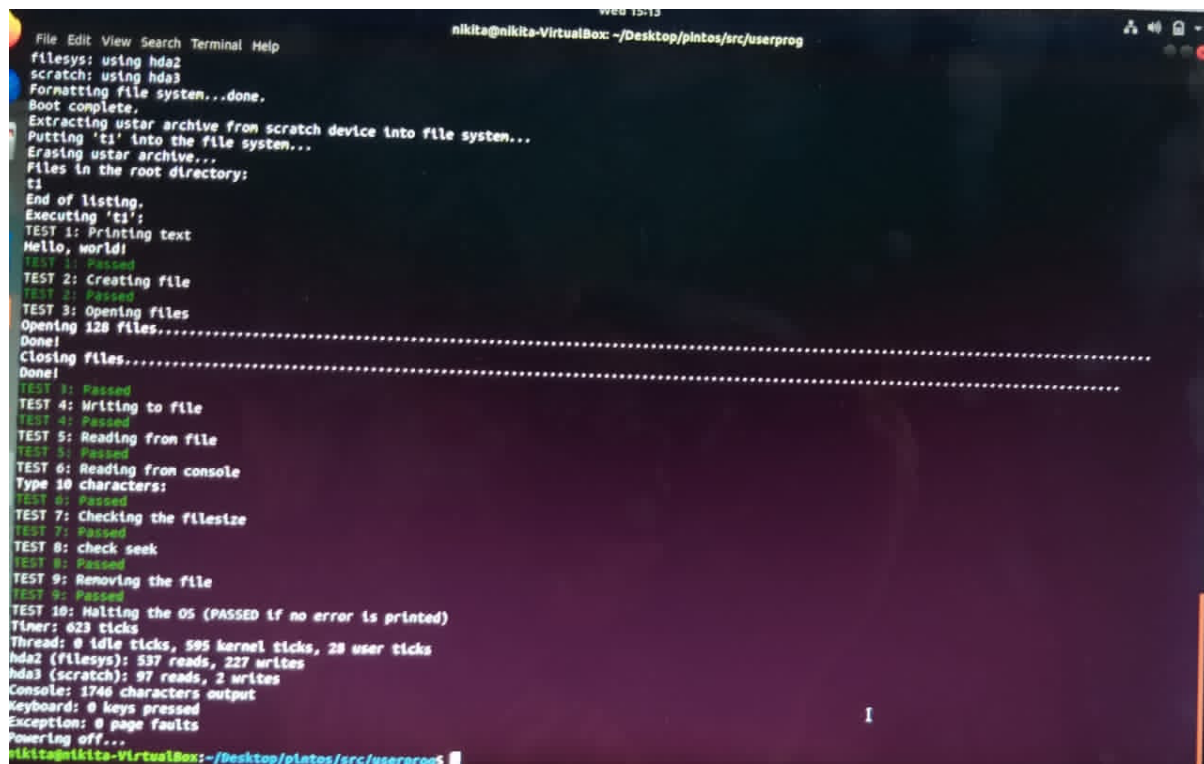
**10. Halt :**

It shutdown the OS using the function in shutdown.h

**Task 5 :-**

The test are written for seek, remove, filesize and halt in testsycalls.c

**Final output :-**



```
File Edit View Search Terminal Help
nikita@nikita-VirtualBox: ~/Desktop/pintos/src/userprog
fileys: using hda2
scratch: using hda3
Formatting file system...done.
Boot complete.
Extracting ustar archive from scratch device into file system...
Putting 't1' into the file system...
Erasing ustar archive...
Files in the root directory:
t1
End of listing.
Executing 't1':
TEST 1: Printing text
Hello, world!
TEST 1: Passed
TEST 2: Creating file
TEST 2: Passed
TEST 3: Opening files
Opening 128 files.....
Done!
Closing files.....
Done!
TEST 3: Passed
TEST 4: Writing to file
TEST 4: Passed
TEST 5: Reading from file
TEST 5: Passed
TEST 6: Reading from console
Type 10 characters:
TEST 6: Passed
TEST 7: Checking the filesize
TEST 7: Passed
TEST 8: check seek
TEST 8: Passed
TEST 9: Removing the file
TEST 9: Passed
TEST 10: Halting the OS (PASSED if no error is printed)
Timer: 623 ticks
Thread: 0 idle ticks, 595 kernel ticks, 20 user ticks
hda2 (fileys): 537 reads, 227 writes
hda3 (scratch): 97 reads, 2 writes
Console: 1746 characters output
Keyboard: 0 keys pressed
Exception: 0 page faults
Powering off...
nikita@nikita-VirtualBox:~/Desktop/pintos/src/userprog$
```

**Submitted By :-**

**Nikita Bhamu**  
**2018CS50413**