**Operating System Lab**

**Lab Exercise #2**



**Session: 2021**

**Submitted by:**

**Wali Muhammad 2021-SE-39**

**Submitted to:**

**Mr. Waqas Ali**

Department of Computer Science, New Campus **University of Engineering and Technology Lahore, Pakistan**

**Command Line Arguments**

It is possible to pass some values from the command line to your C/C++ programs when they are executed. These values are called command line arguments and many times they are important for your program especially when you want to control your program from outside instead of hard coding those values inside the code. The command line arguments are handled using main () function arguments using argc and argv [].

**Task 1**

**Description**

You are required to create a program, which will receive the name of a text file via command line arguments, and it will read that file and print the contents of the file on console. You can assume that the text file to be read is placed in the same folder as your executable program. Program should be able to process multiple file name given to it by command line and read & print its content.

**Solution**

**Code**

#include<iostream>

#include<fstream>

#include<string>

using namespace std;

int main(int argc,char \*name[])

{

fstream file;

string data;

string n;

for(int i=1;i<argc;i++)

{

file.open(name[i]);

while(getline(file,data))

{

cout<<data<<endl;

}

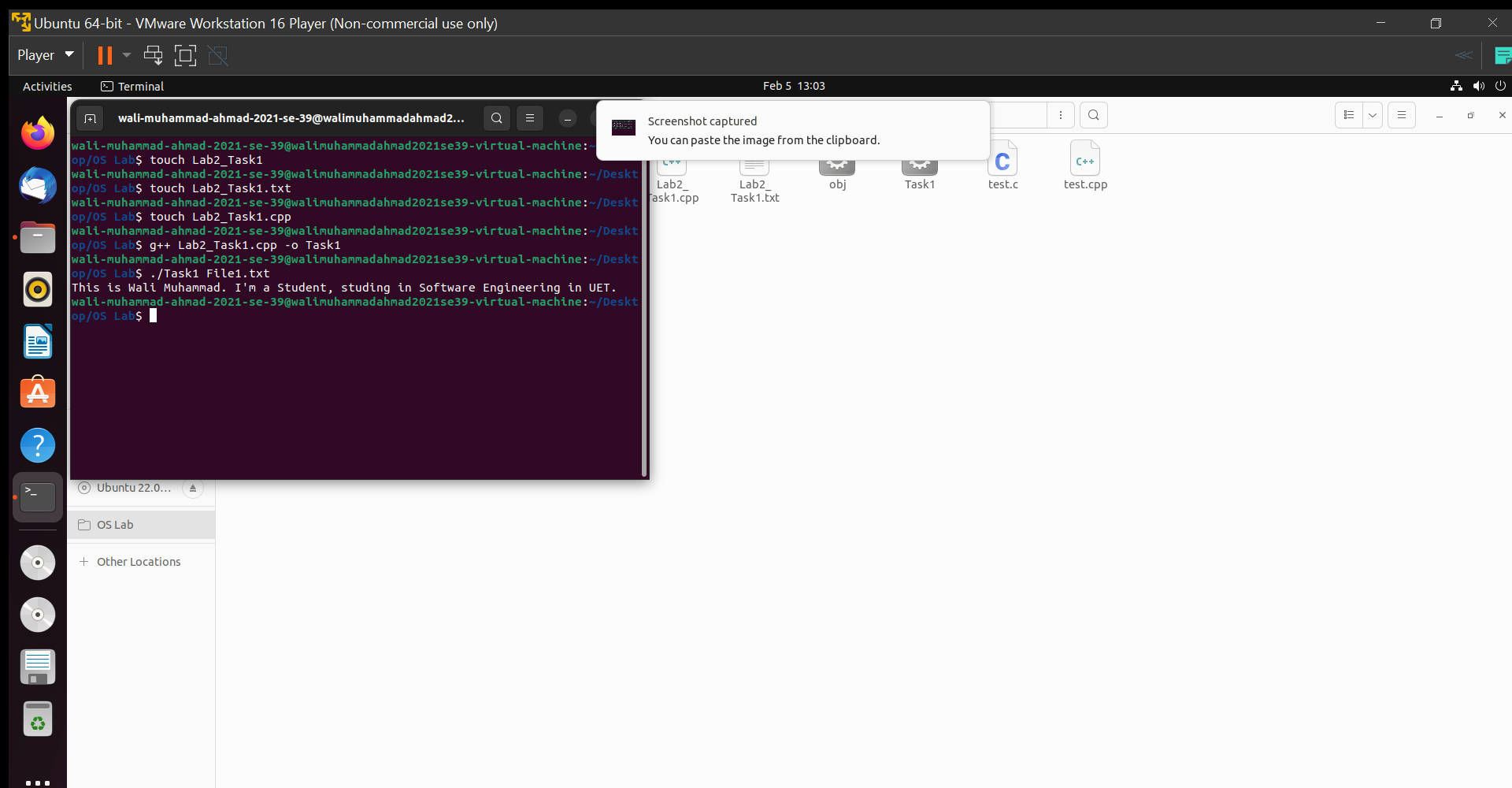
file.close();

}

return 0;

}

**Code Execution Screenshot**

****

**Task 2**

**Description**

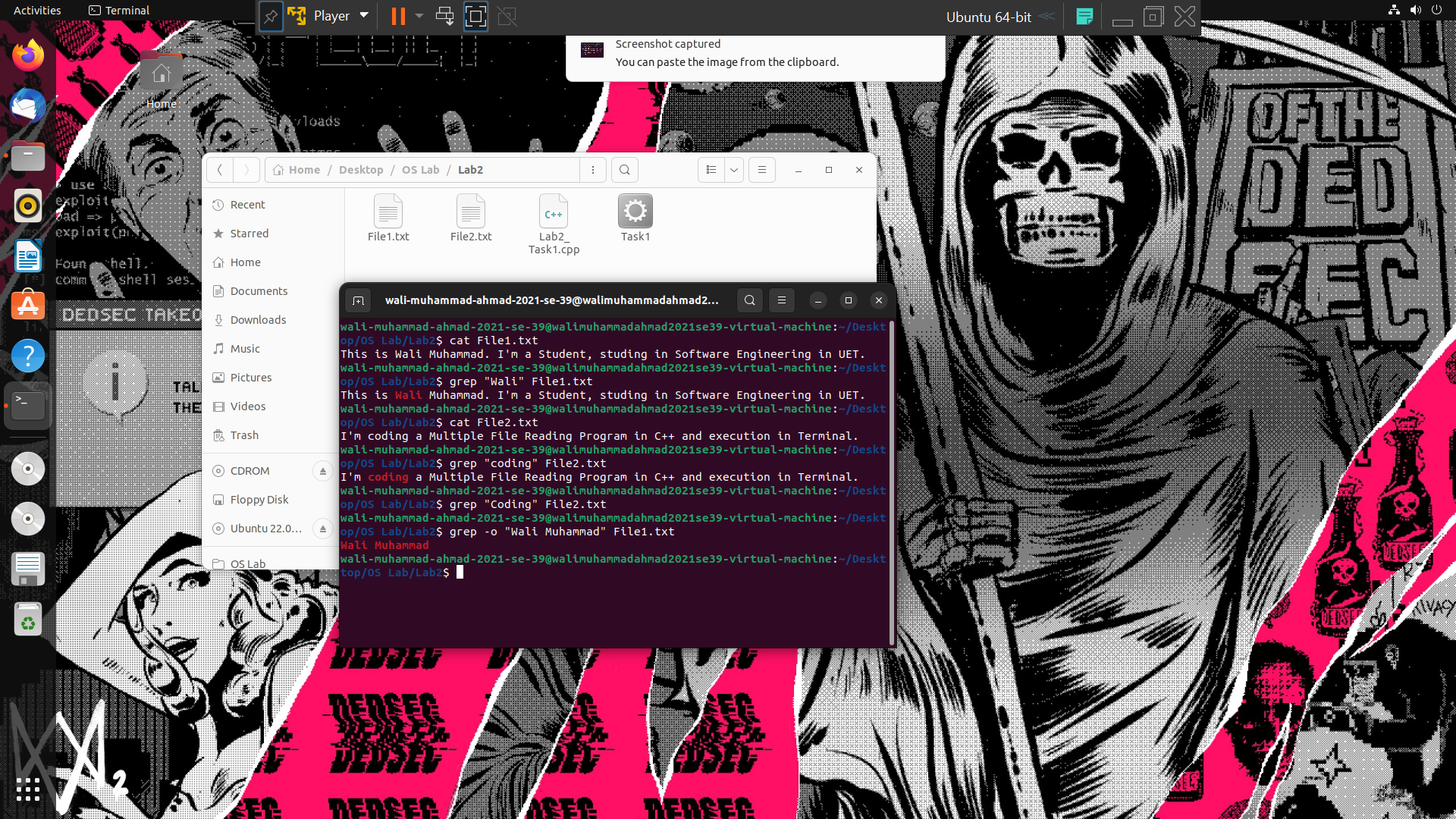
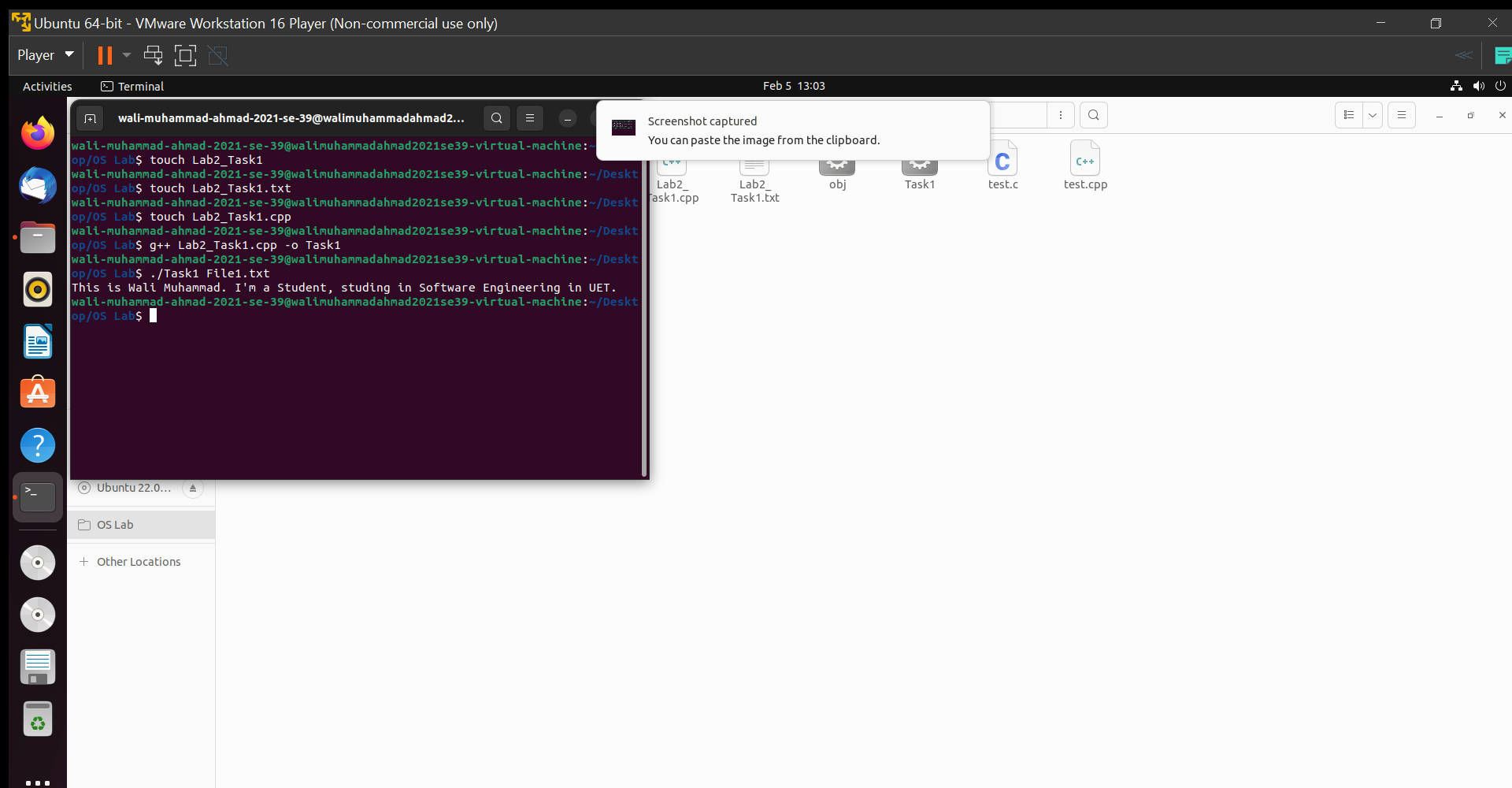
To simulate grep command using UNIX system call.

**Algorithm**

1. Get filename and search string as command-line argument
2. Open the file in read-only mode using open system call.
3. If file does not exist, then stop.
4. Let length of the search string be n.
5. Read line-by-line until end-of-file. Check to find out the occurrence of the search string in a line by examining characters. If search string exists, then print the line
6. Close the file using close system call.
7. Stop.

**Solution**

1. **Using grep command**

****

1. **Using C++ code**

**Code**

#include<iostream>

#include<fstream>

#include<string>

using namespace std;

int main(int argc,char \*name[])

{

fstream file;

string data;

string n=name[1];

string word;

for(int i=2;i<argc;i++)

{

file.open(name[i],fstream::in);

while(getline(file,data))

{

for(int j=data.find(name[1]),k=0;k<n.size();j++,k++)

{

word+= data[j];

}

if(word==name[1])

{

cout<<data<<endl;

}

word="\0";

}

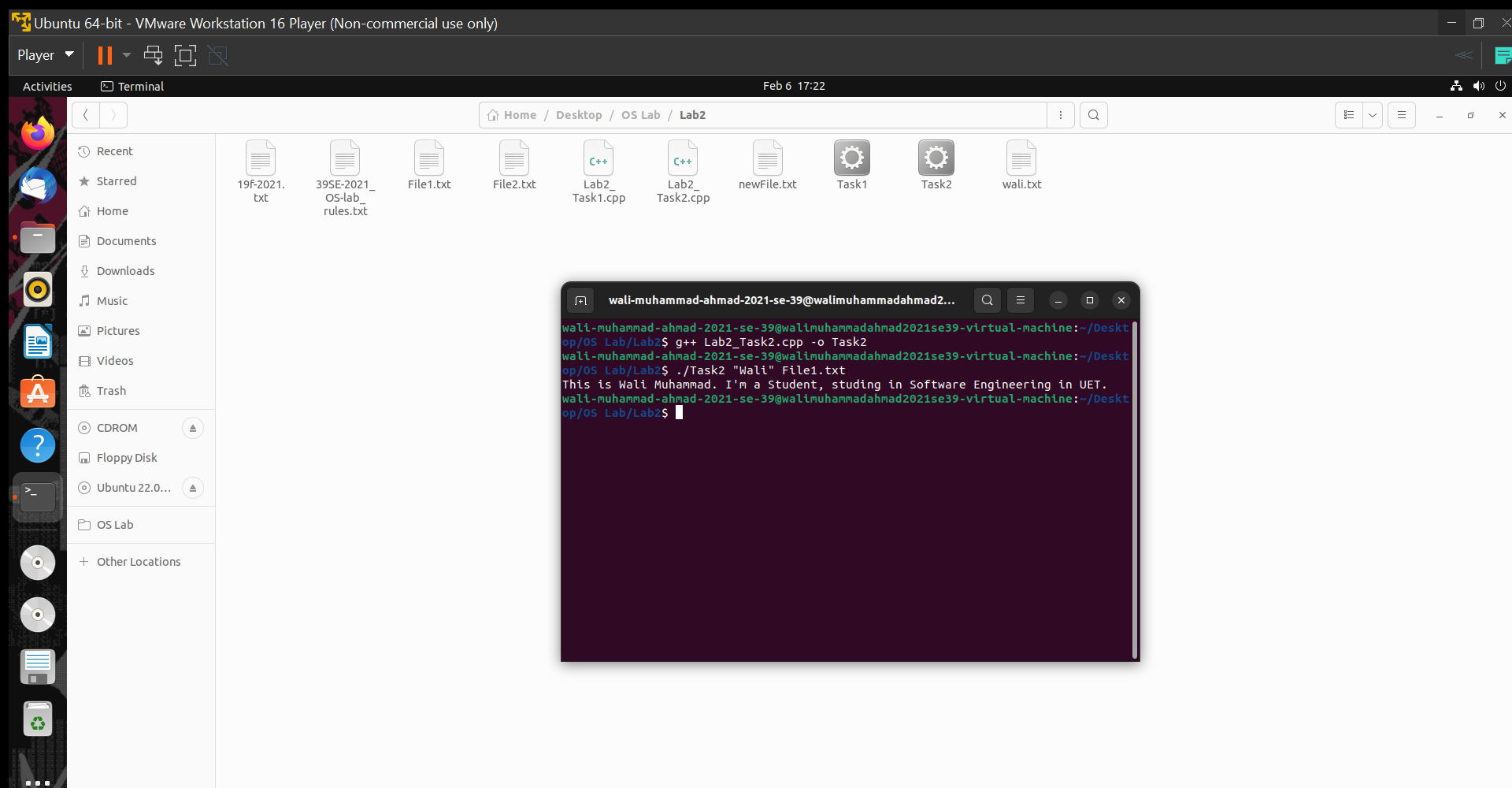
file.close();

}

return 0;

}

**Code Execution Screenshot**



**Task 3**

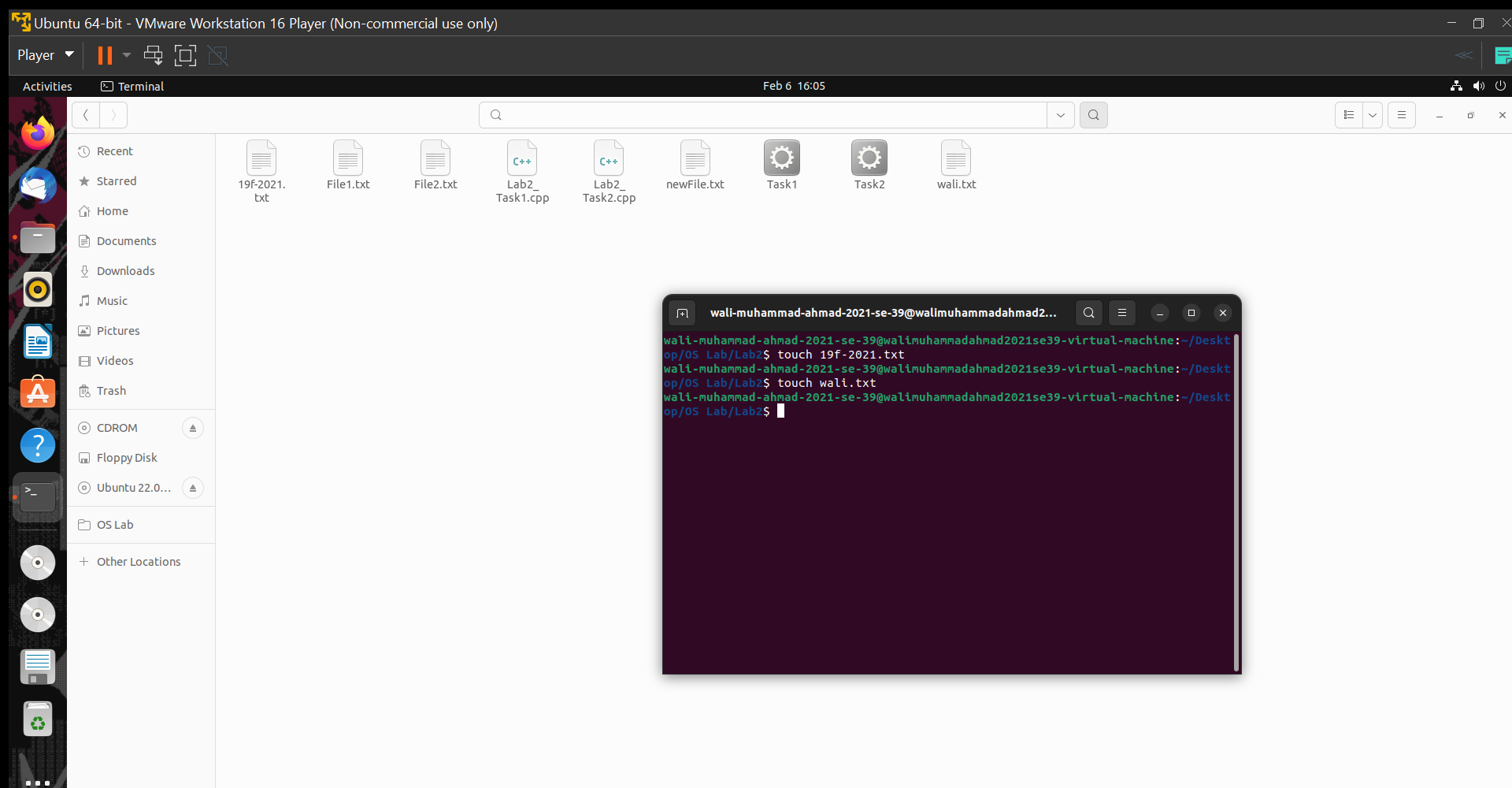
**Description**

You are required to help a new Ubuntu user in performing Ubuntu commands. Mr. Tom is a new Ubuntu user. He wants guidance regarding basic Ubuntu utilities to perform following tasks:

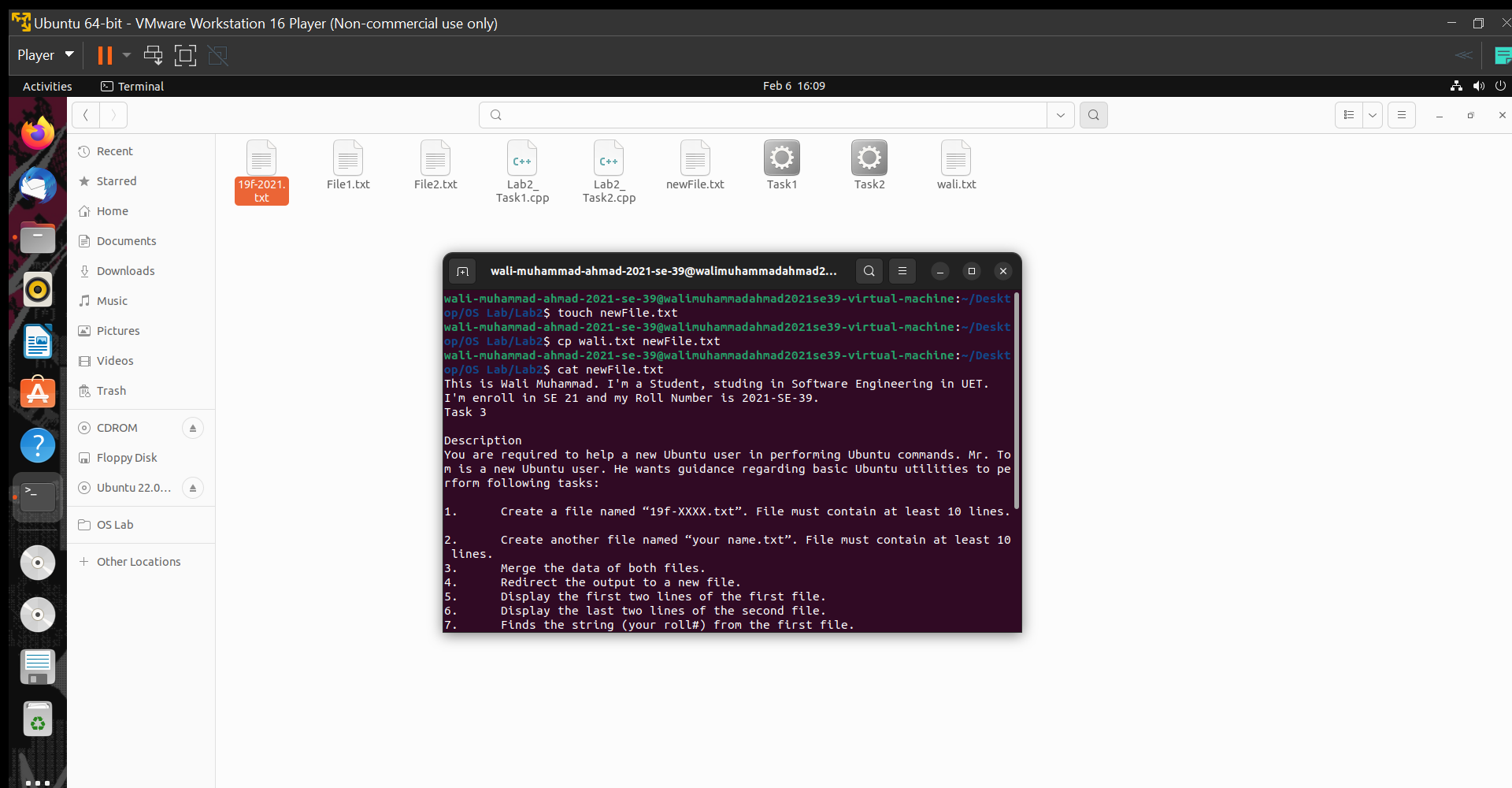
1. Create a file named “19f-XXXX.txt”. File must contain at least 10 lines.
2. Create another file named “your name.txt”. File must contain at least 10 lines.
3. Merge the data of both files.
4. Redirect the output to a new file.
5. Display the first two lines of the first file.
6. Display the last two lines of the second file.
7. Finds the string (your roll#) from the first file.
8. Grant the execute permission of the second file to the group.
9. Remove the write permission for the owner.
10. Now Mr. Tom suddenly lost track of his current location. Help him find his location.
11. He wants the list of all files present in the Desktop directory.
12. Now he wanted to create a folder of his personal files and pictures named as your roll#.
13. Display the current time.
14. He is done with the task and he is happy with your work. He wants to display a thankyou message.

**Solution**

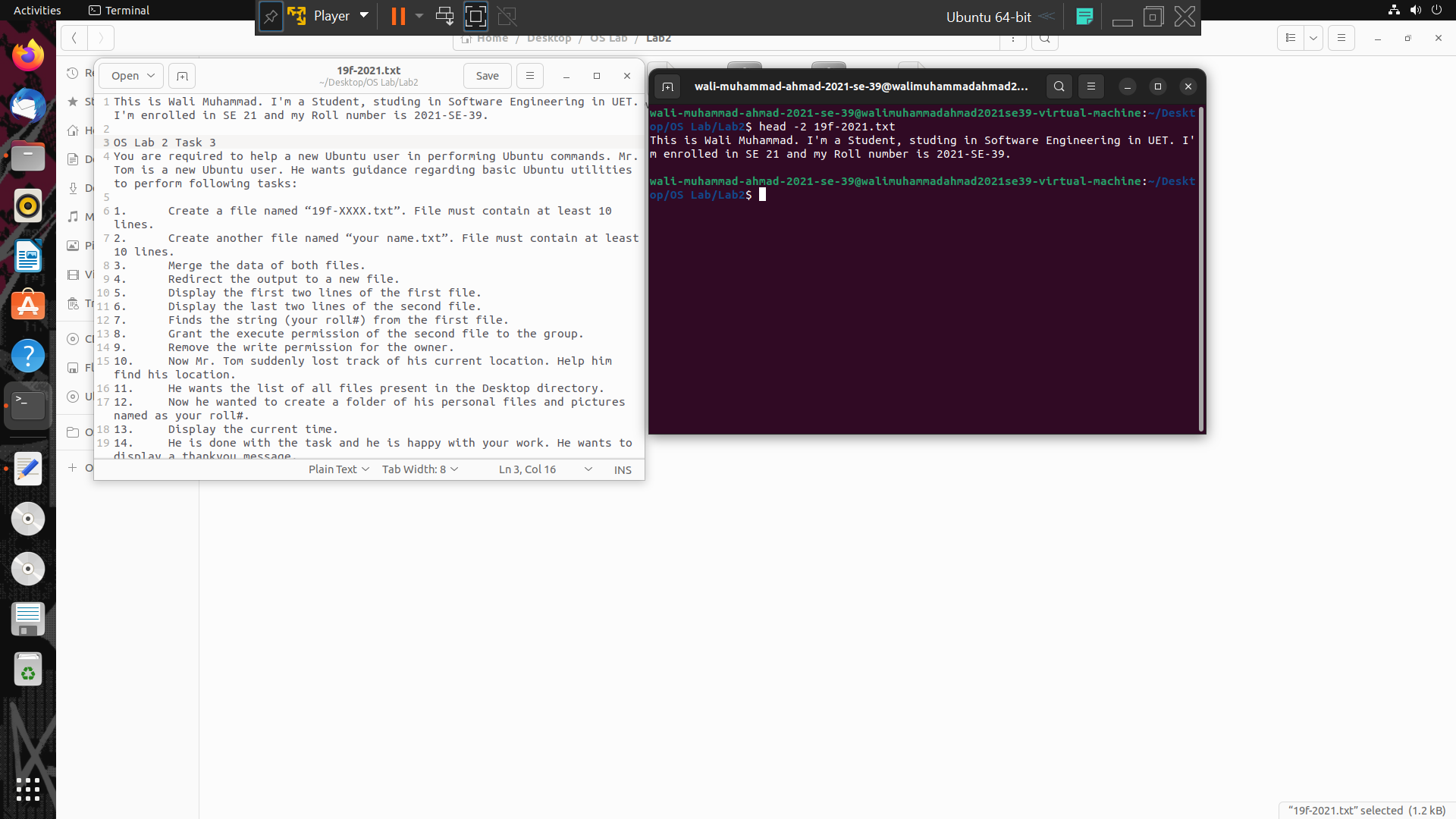
**Step 1,2**

****

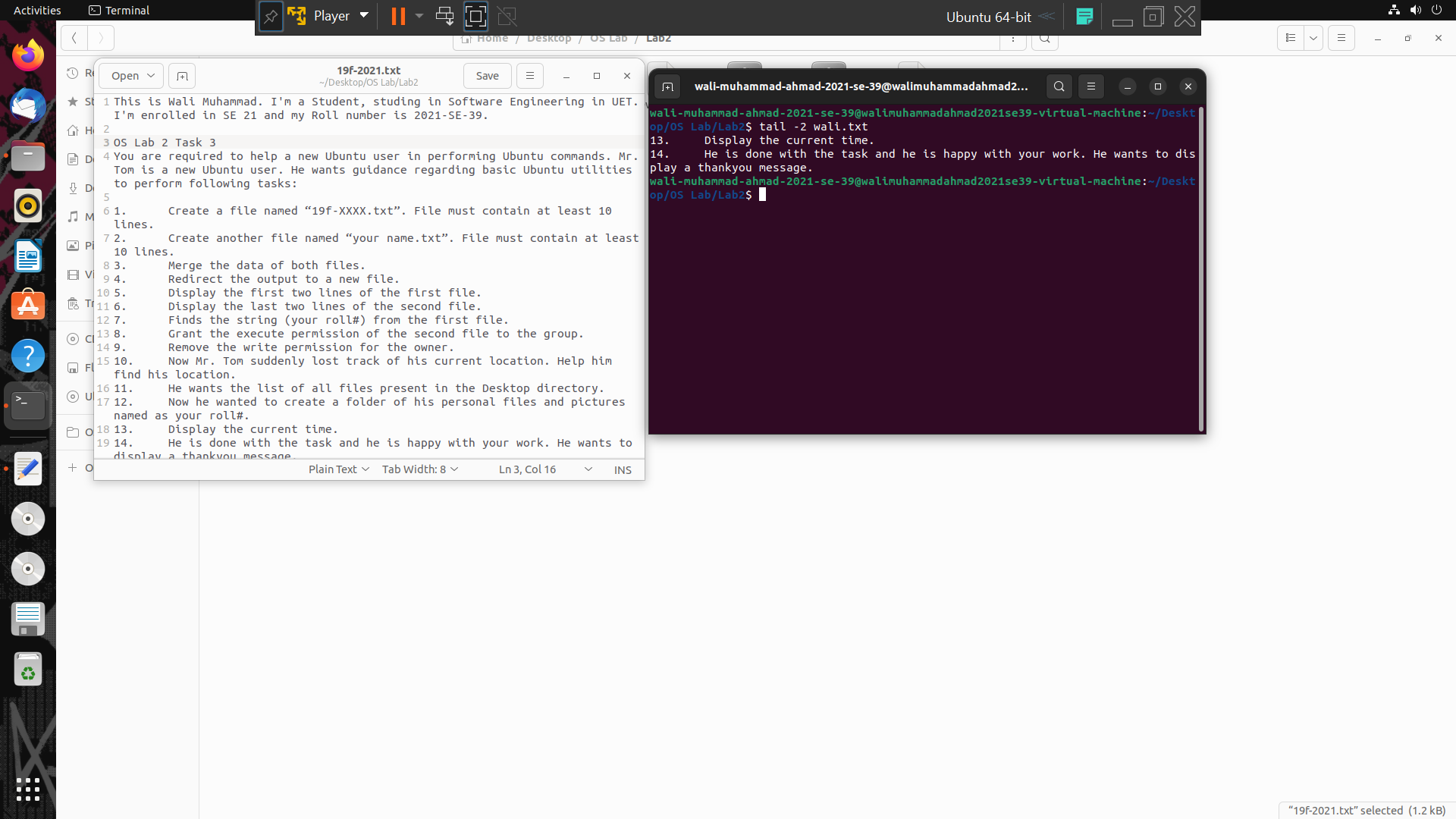
**Step 3,4**

****

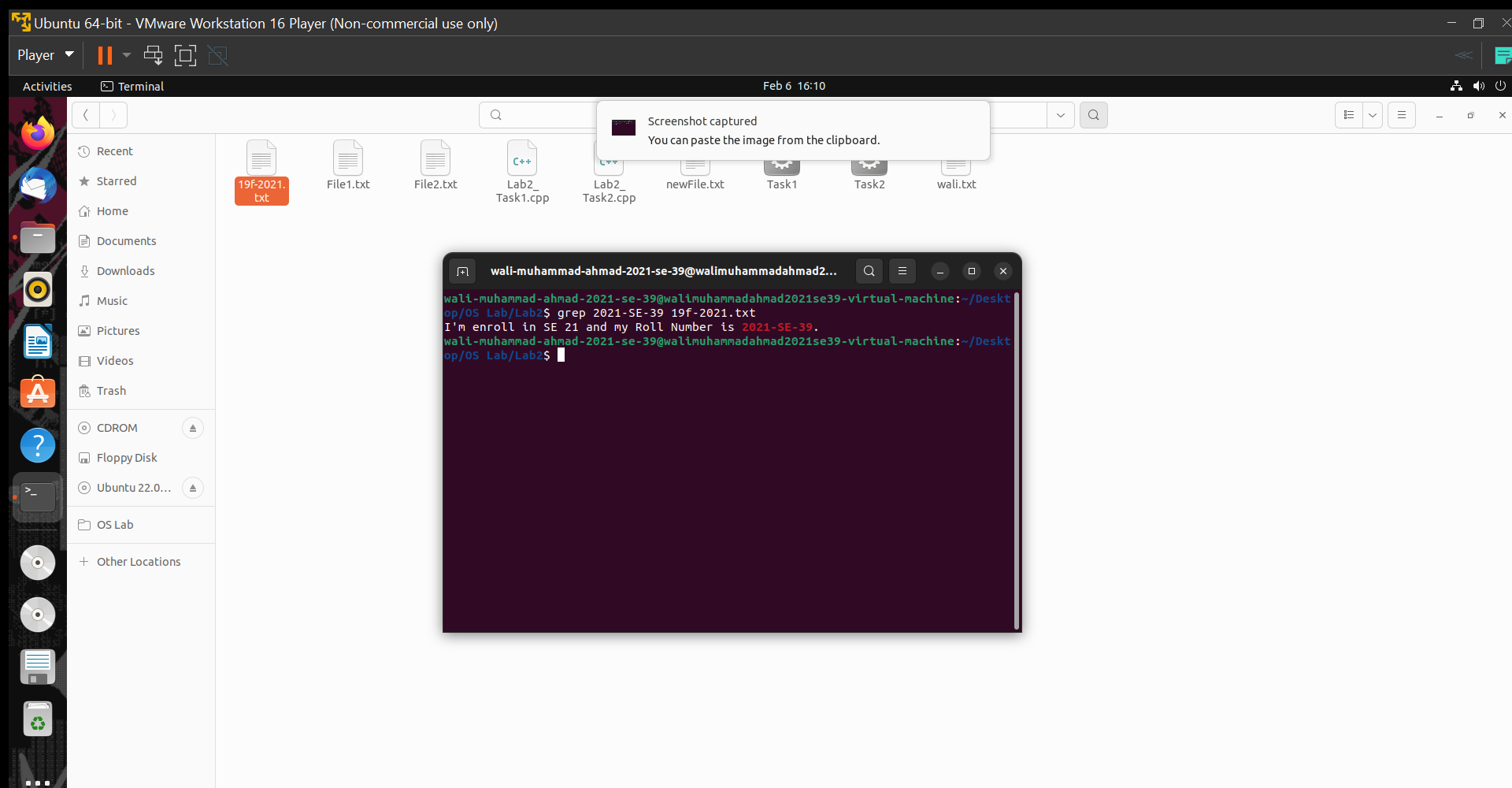
**Step 5**

****

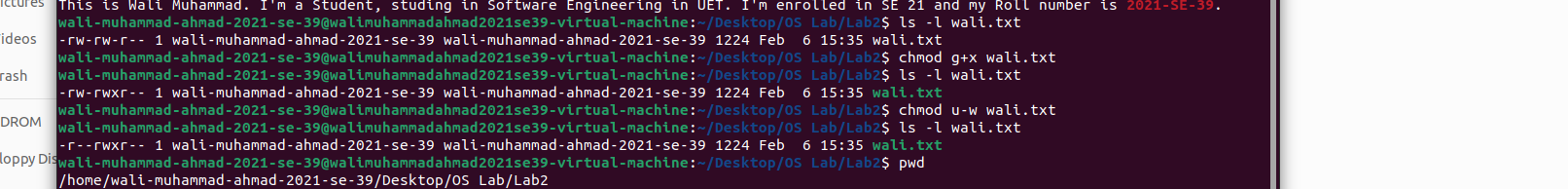
**Step 6**

****

**Step 7**

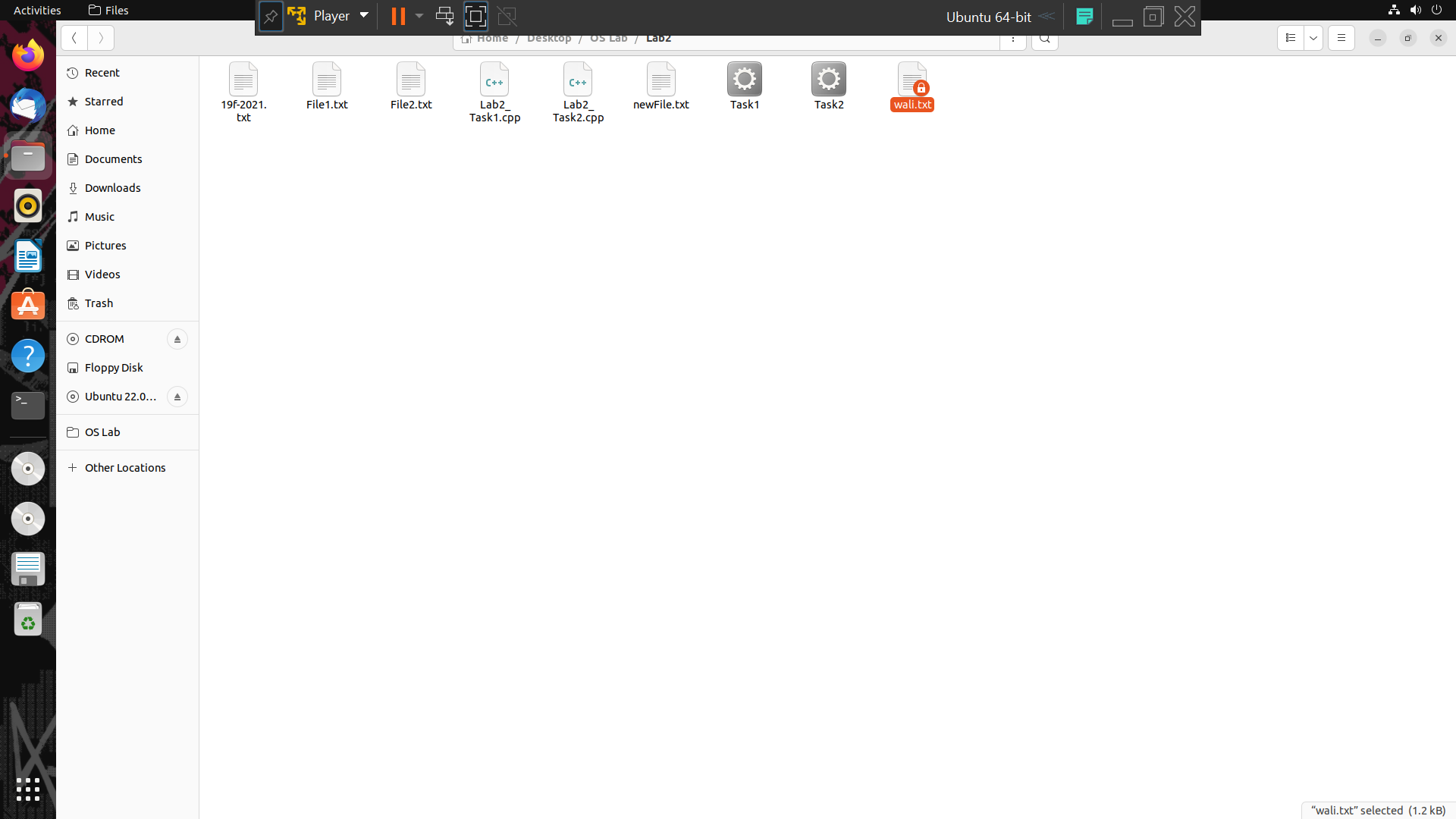
****

**Step 8,9**

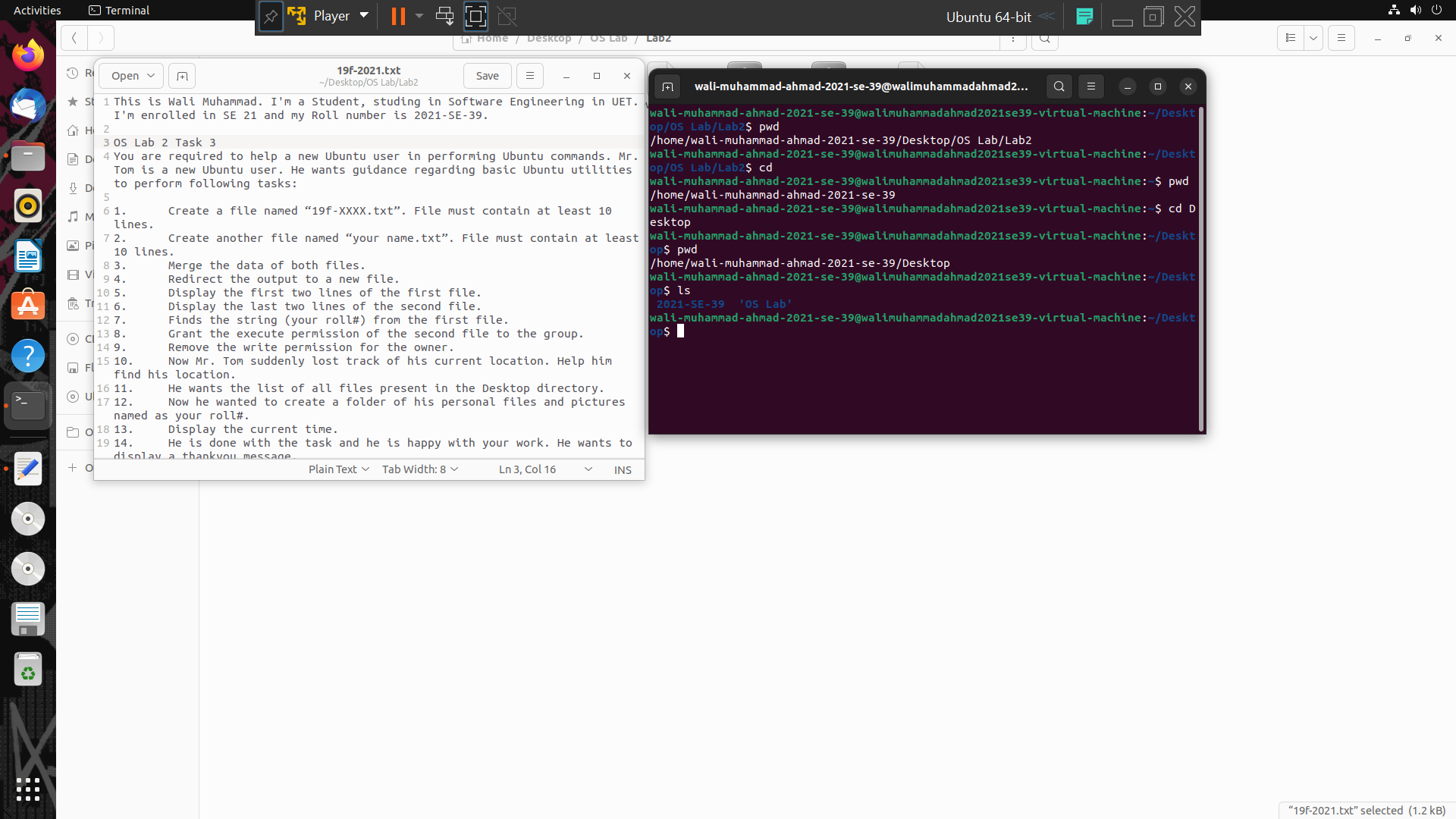
****

**Step 9**

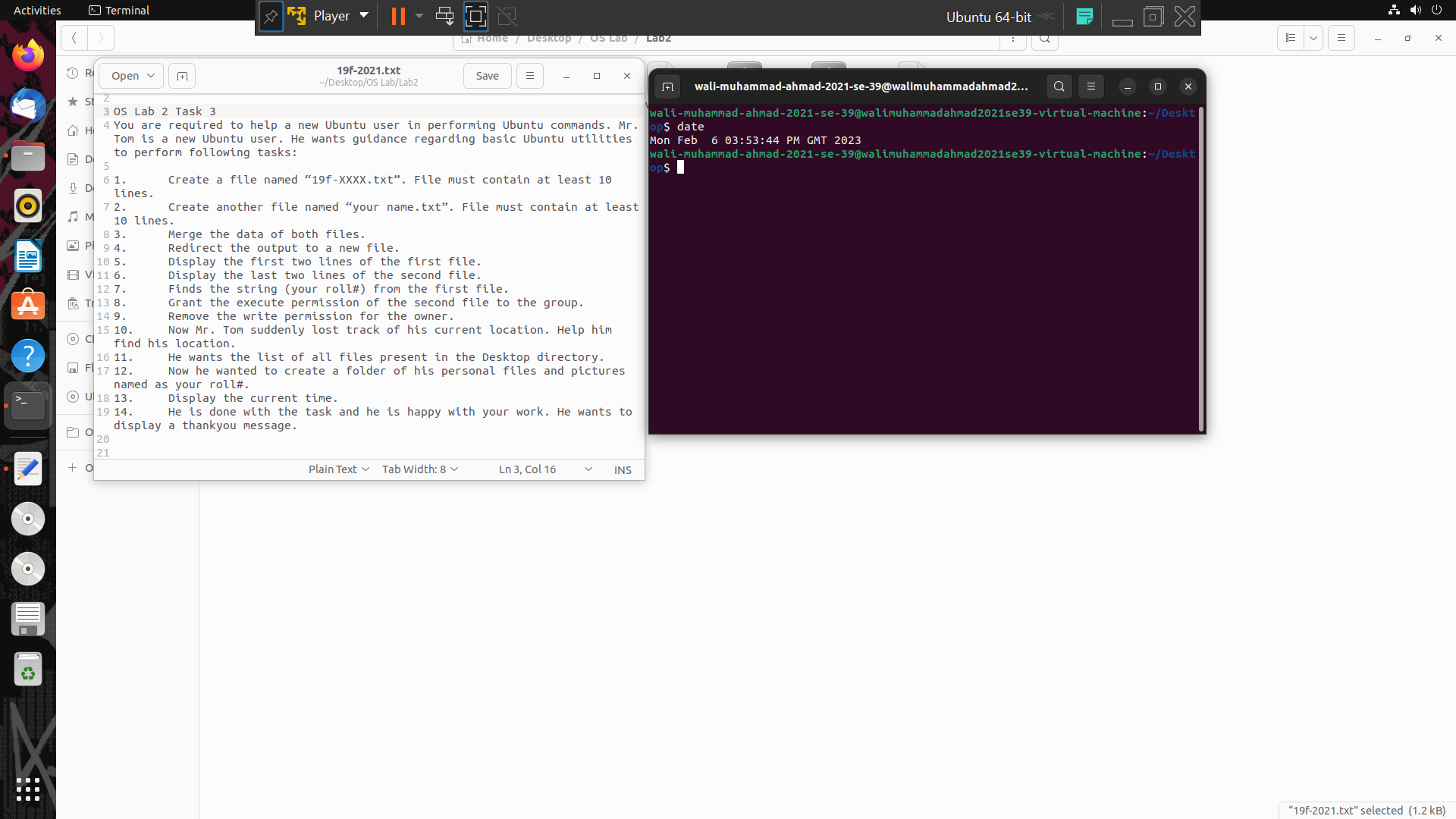
After removing the write permission for the owner.

****

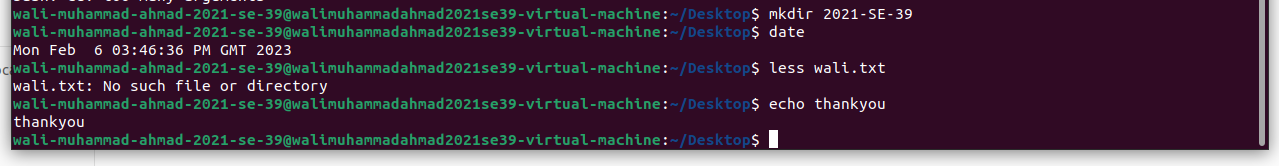
**Step 10,11**

****

**Step 13**

****

**Step 12,13,14**

****

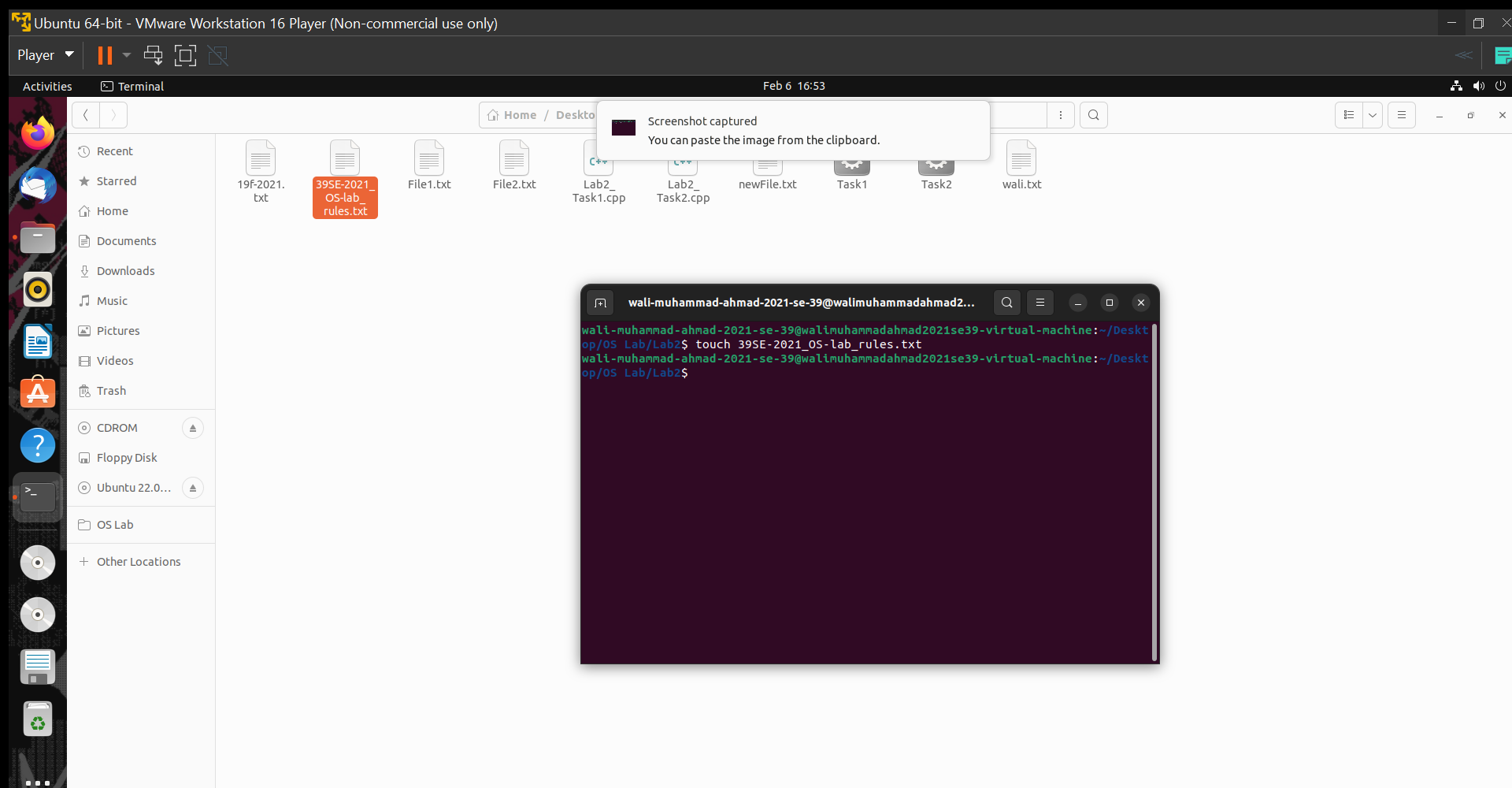
**Task 4**

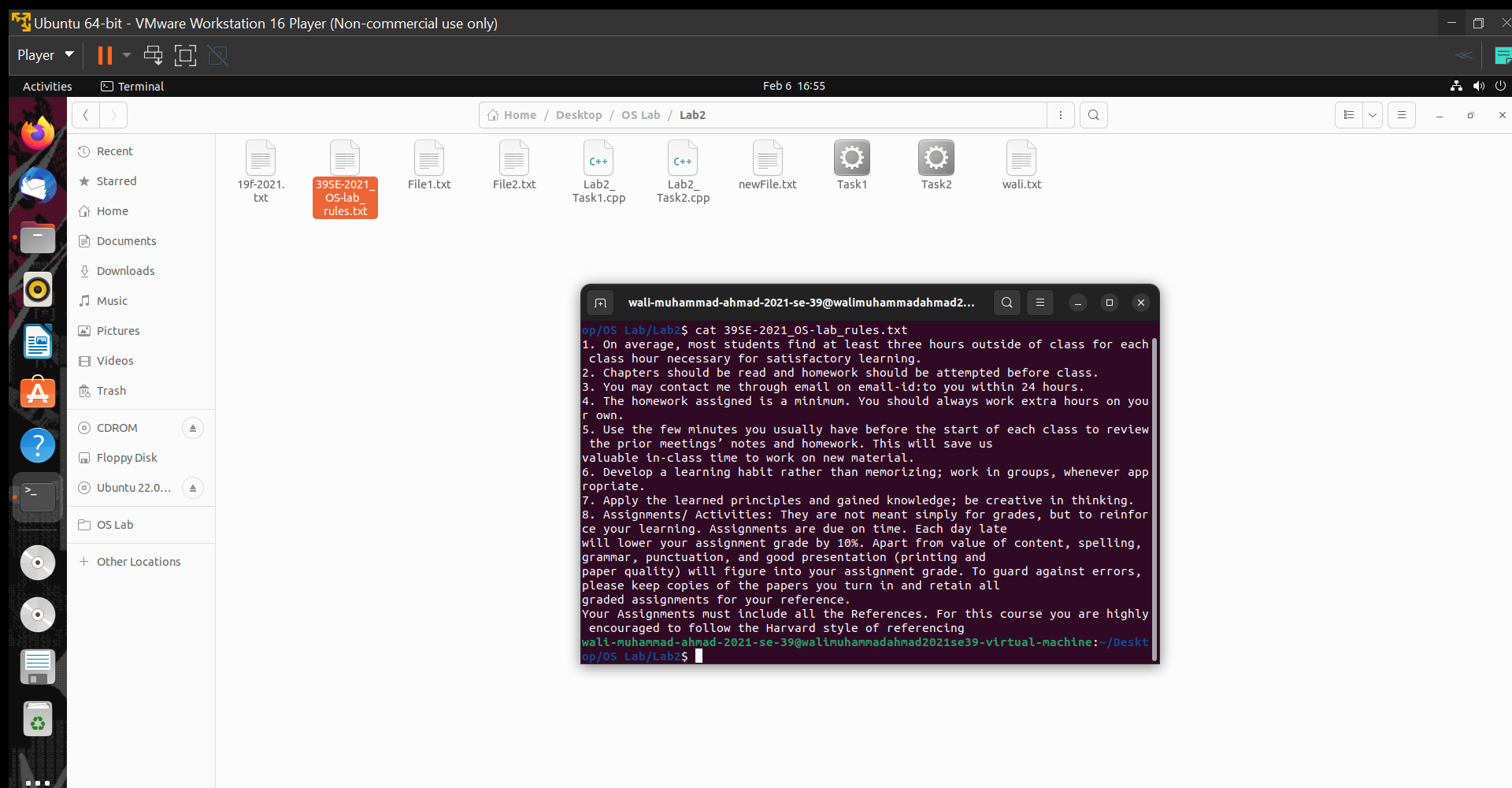
**Description**

1. Create a file named “19f-XXXX\_OS-lab\_rules.txt” using Linux commands. The file must contain all lab rules covered in the first lab.
2. You want to set the rights of the created file to this (- rwx r-x r--). For these rights, you are required to convert the given rights in numeric format using binary number system procedure covered in lecture. Show complete working.
3. Now use the derived number to change the permission of a file using the chmod command.
4. Append the output of ls command to the created file.

**Solution**

**Step 1**

****

****

**Step 2**

You want to set the rights of the created file to this (- rwx r-x r--). For these rights, you are required to convert the given rights in numeric format using binary number system procedure covered in lecture.

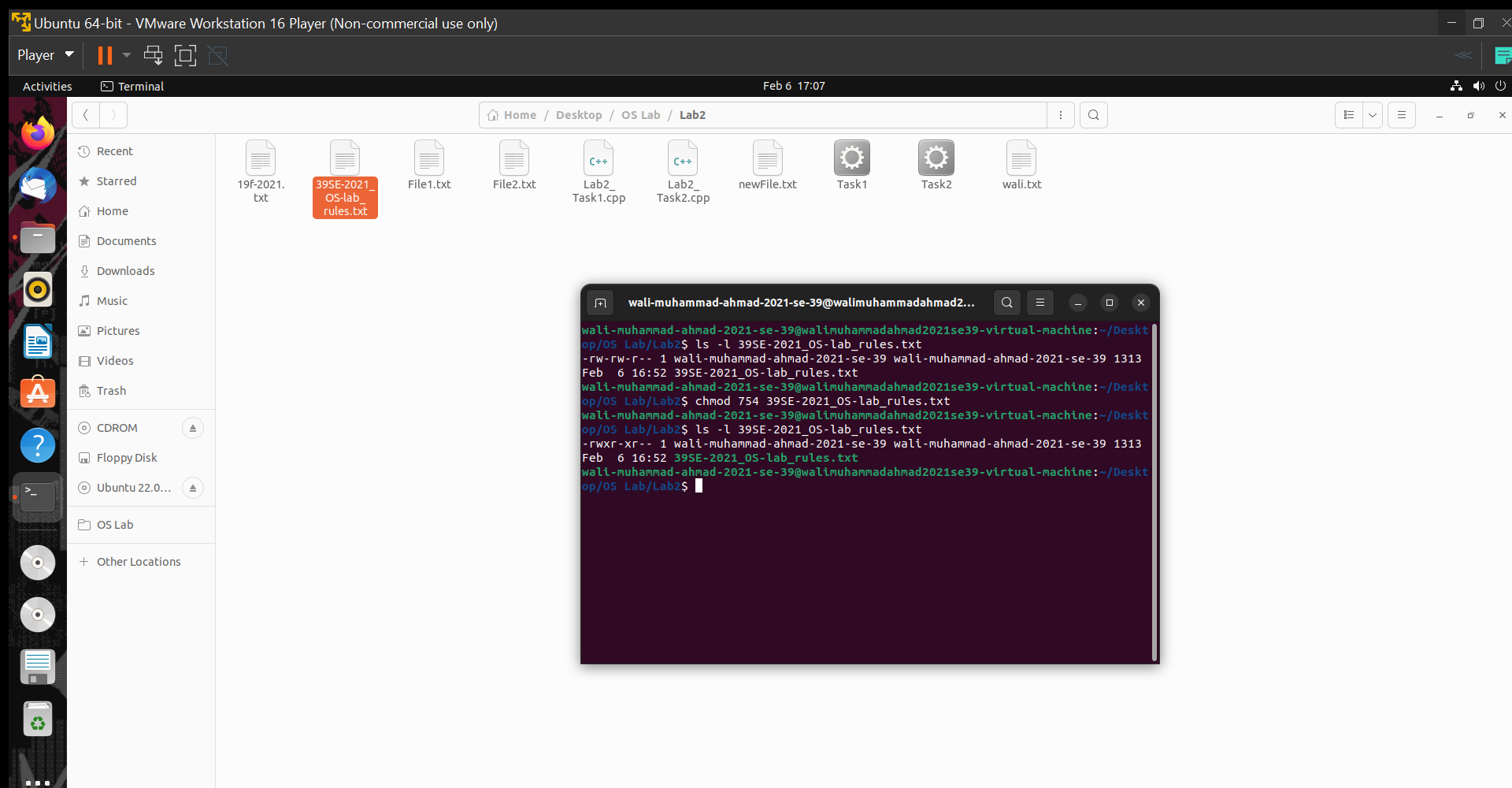
For this step we need to divide the rights into three steps and convert them into binary numbers. For this purpose, the given table in OS LAB 2 Slides is used.

**According to table in provided slides**

1. **(rws) = read, write and execute = 7**
2. **(r-x) = read and execute = 5**
3. **(r--) = read only = 4**

So, the (- rwx r-x r--) will be 754 in decimal and 111 101 100 in binary number system.

**Step 3**

****

**THE END**