**Institute of Computer Technology**

**B. Tech. Computer Science and Engineering**

**Semester: IV**

**Sub: Operating System**

**Course Code: 2CSE402**

**Practical Number:1**

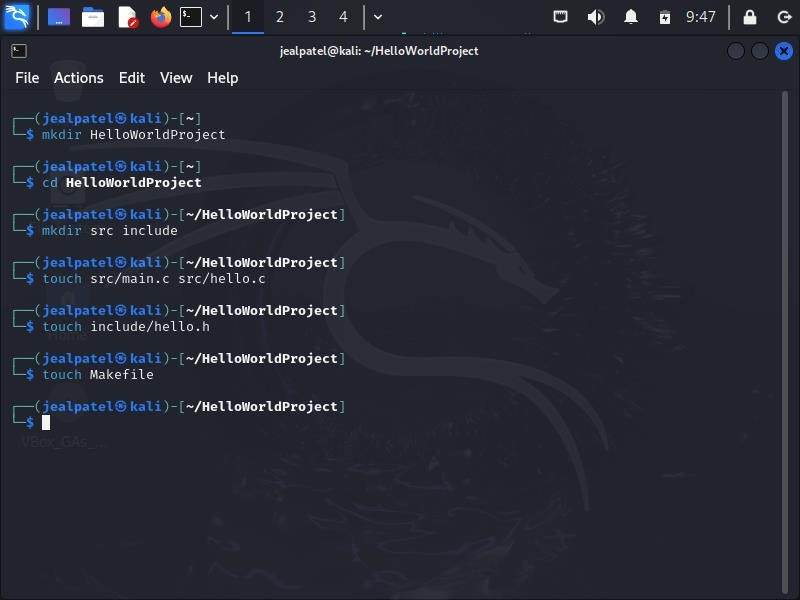
**Objective:**

*Basic Linux Command Practice.*

Q.1: Makefiles are used to help decide which parts of a large program need to be recompiled. In the vast majority of cases, C or C++ files are compiled. Other languages typically have their own tools that serve a similar purpose as Make. It can be used beyond programs too, when you need a series of instructions to run depending on what files have changed. This experiment focus on the C/C++ compilation use case attached herewith. Here's an example dependency graph that you might build with Make. If any file's dependencies changes, then the file will get recompiled.  
  
  
in short, A makefile is a special file that lists a set of rules for compiling a project. These rules include , which can be an action make needs to take (eg. "clean" or "build") or the files/objects make will need to build (eg. .o files or an executable), and the commands that need to be run in order to build that target. When you call the program make, it runs through each of these targets in your Makefile and executes them.    
  
  
Sample Code:  
  
  
blah.o: blah.c  
gcc -c blah.c -o blah.o # Runs second  
  
blah.c:  
  
echo "int main() { return 0; }" > blah.c # Runs first  
  
Tasks to be done by students:  
  
1. Write a simple “Hello World” program. Move the part that outputs the text (i.e., the printf() function call), into a separate file hello.c. Edit an appropriate interface file hello.h in order to use the subroutine in the main.c program. Write a Makefile that maintains the program structure. Use a variable CC in the makefile to define the compiler, e.g., CC=gcc.  
  
2. Place the header files and C/C++ files  in separate directory and compile and execute those files using Makefile Utility.

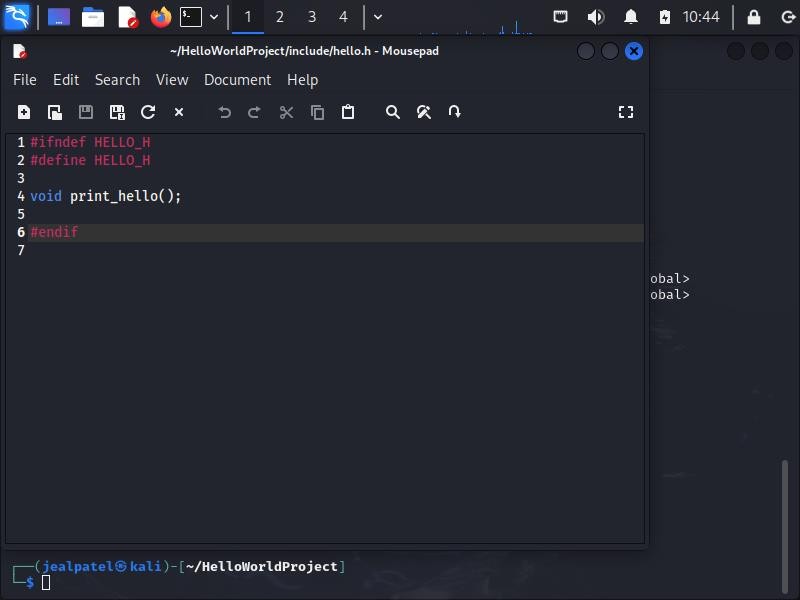
**Steps :**

*Step 1:Create Files and Directories*

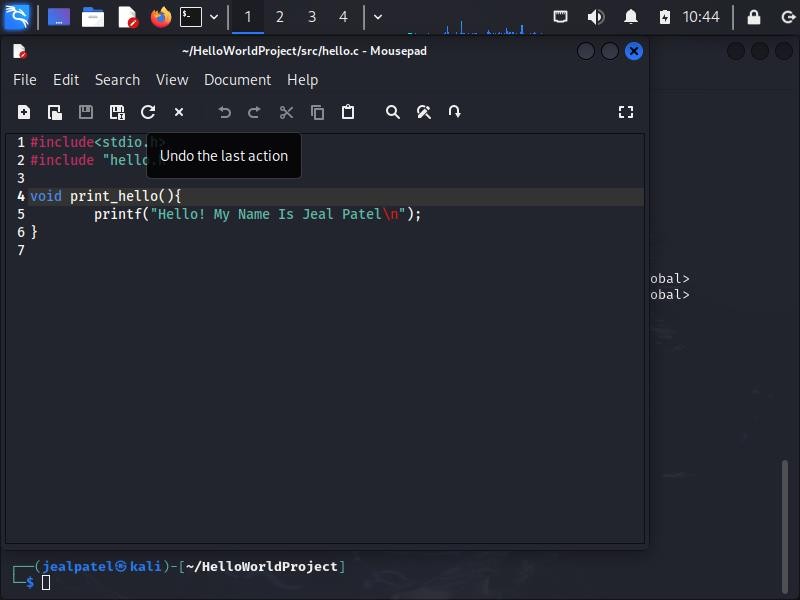
**

*Step 2:Write Code Into Specific Files*

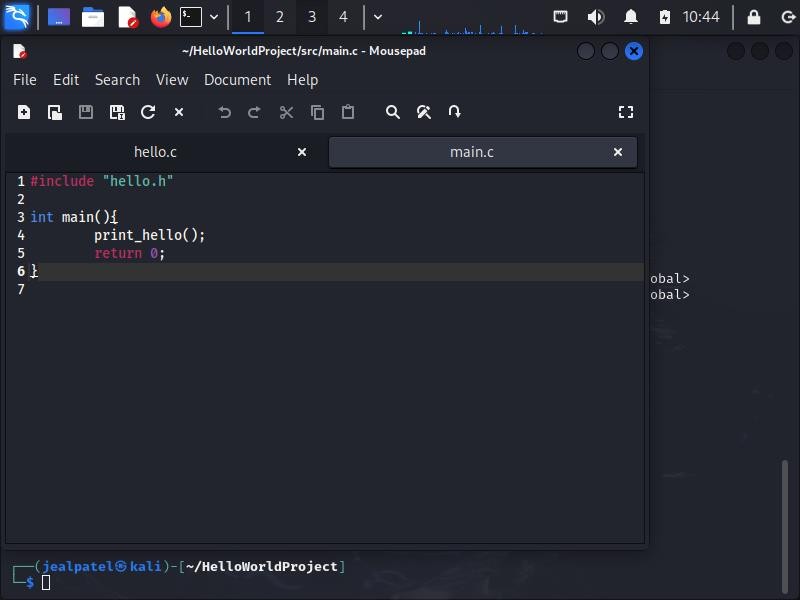
1. *nano include/hello.h*

**

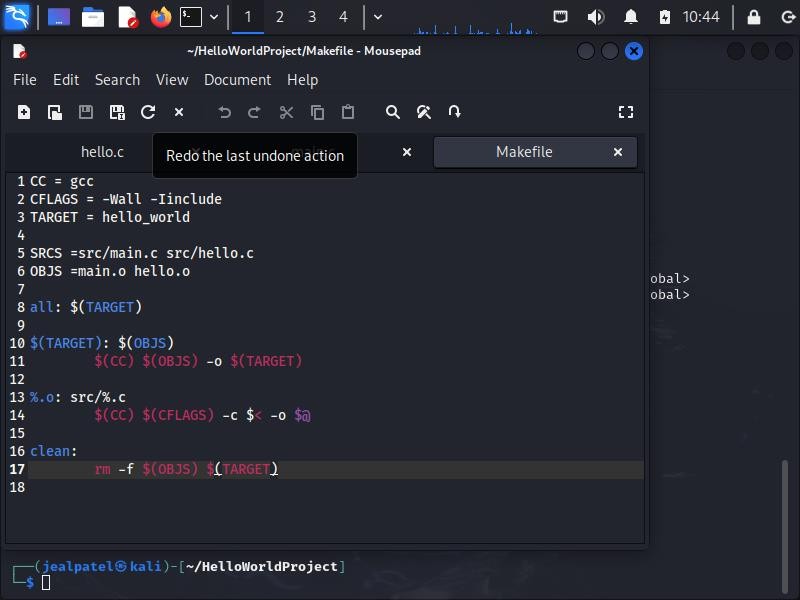
1. *nano src/hello.c*

**

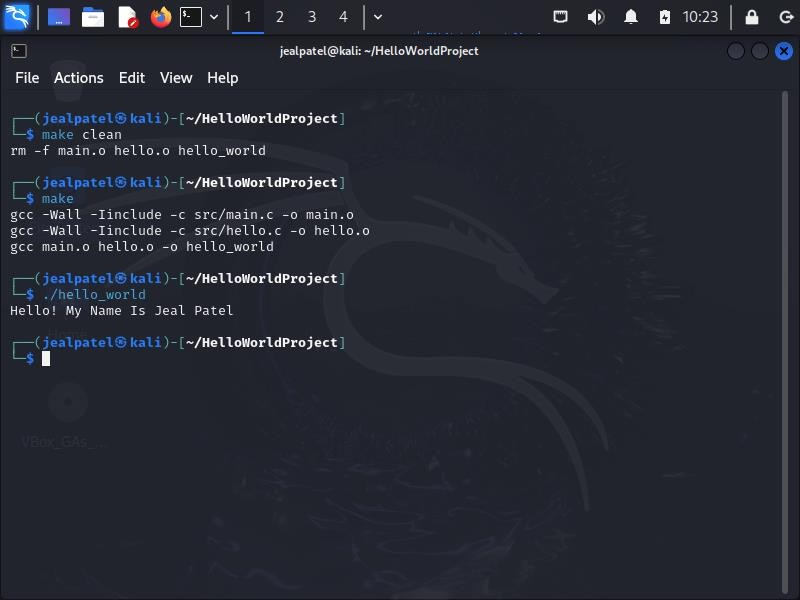
1. *nano src/main.c*

**

1. *nano Makefile*

**

*Step 3:Compile the program*

**