Name: Nivedita Londhe
PRN: 22420003
Batch: EN-4
Practical No. 5

Title: Use of Makefile

Procedure:

1. **Install *make***

   Installing make using package manager:

   ```
   sudo apt-get install make
   ```

2. **Creating *Makefile* in project directory**

   Project Directory: `cd /RTOS/Lab`

   Create the Makefile: `touch Makefile`

3. **Structure of the *Makefile***

   Targets: What we want to built (e.g. executable files)

   Dependencies: Files that the target depends on

   Commands: Actions to execute

4. **Running *Makefile***

   Project Directory: `cd /RTOS/Lab`

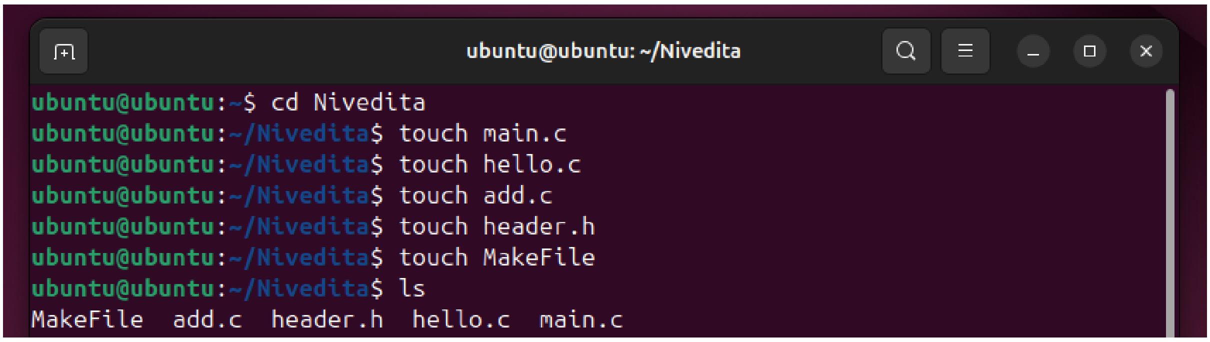
   Run make to build the target specified in the Makefile: `make`

5. **Run specific target**

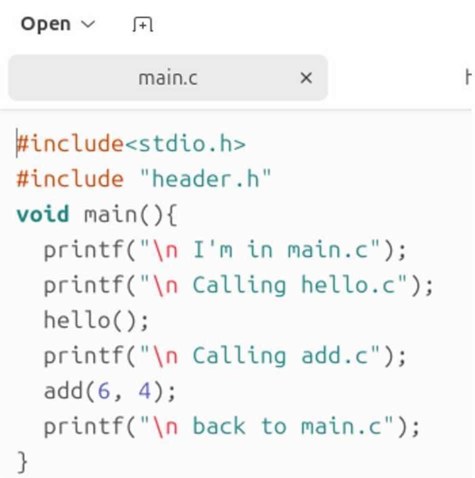
   To clean the build files (remove object files and the executable):

   ```
   make clean
   ```

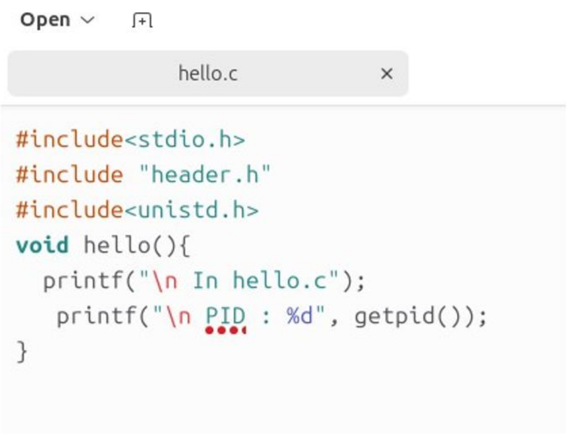
Screenshots:



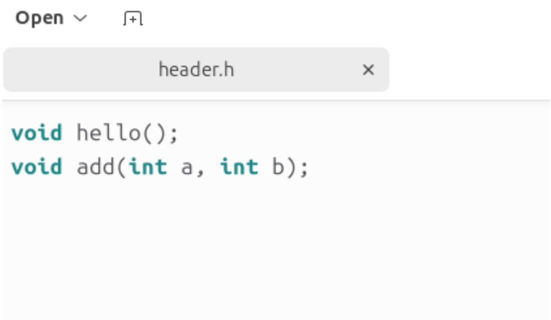
```
ubuntu@ubuntu:~$ cd Nivedita
ubuntu@ubuntu:~/Nivedita$ touch main.c
ubuntu@ubuntu:~/Nivedita$ touch hello.c
ubuntu@ubuntu:~/Nivedita$ touch add.c
ubuntu@ubuntu:~/Nivedita$ touch header.h
ubuntu@ubuntu:~/Nivedita$ touch MakeFile
ubuntu@ubuntu:~/Nivedita$ ls
MakeFile  add.c  header.h  hello.c  main.c
```



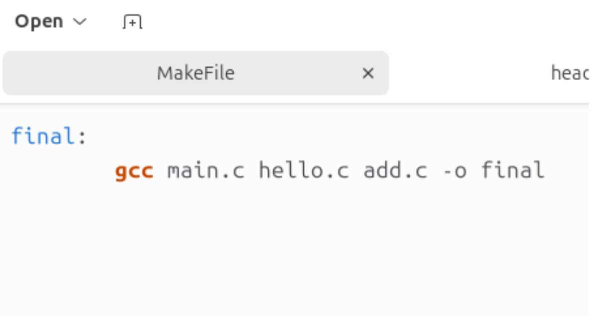
```c
#include<stdio.h>
#include "header.h"
void main(){
  printf("\n I'm in main.c");
  printf("\n Calling hello.c");
  hello();
  printf("\n Calling add.c");
  add(6, 4);
  printf("\n back to main.c");
}
```



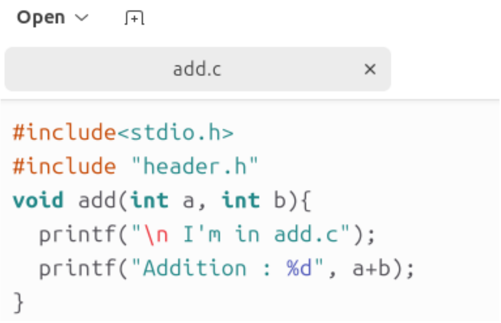
```c
#include<stdio.h>
#include "header.h"
#include<unistd.h>
void hello(){
  printf("\n In hello.c");
   printf("\n PID : %d", getpid());
}
```



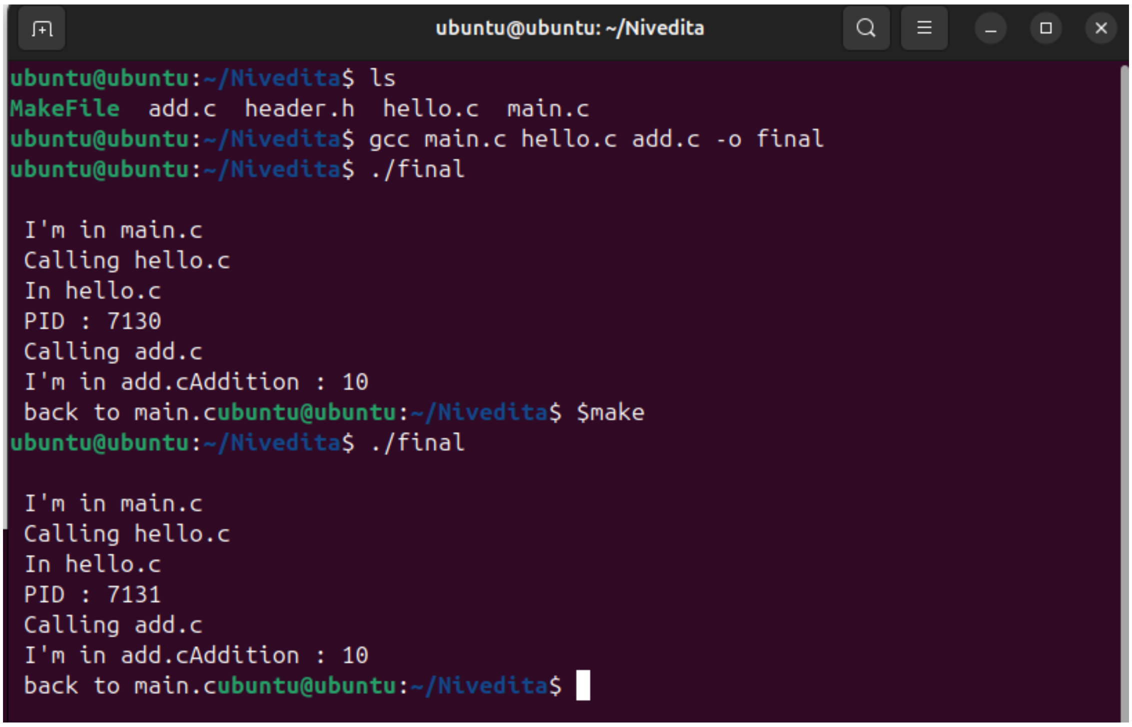
```c
void hello();
void add(int a, int b);
```



```
final:
        gcc main.c hello.c add.c -o final
```



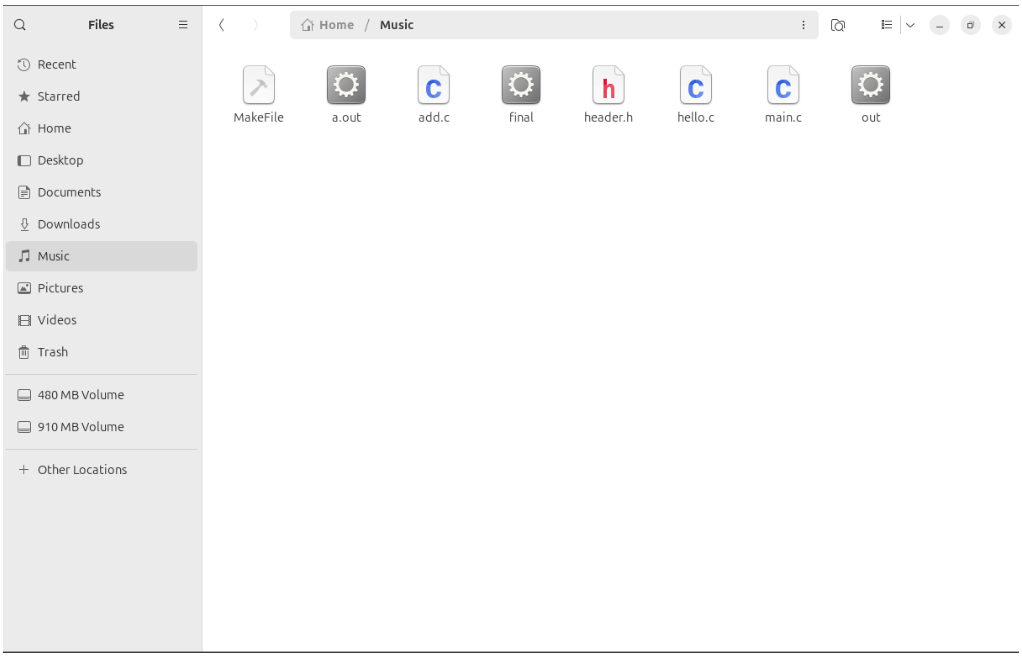
```c
#include<stdio.h>
#include "header.h"
void add(int a, int b){
  printf("\n I'm in add.c");
  printf("Addition : %d", a+b);
}
```



```
ubuntu@ubuntu:~/Nivedita$ ls
MakeFile  add.c  header.h  hello.c  main.c
ubuntu@ubuntu:~/Nivedita$ gcc main.c hello.c add.c -o final
ubuntu@ubuntu:~/Nivedita$ ./final

 I'm in main.c
 Calling hello.c
 In hello.c
 PID : 7130
 Calling add.c
 I'm in add.cAddition : 10
 back to main.cubuntu@ubuntu:~/Nivedita$ $make
ubuntu@ubuntu:~/Nivedita$ ./final

 I'm in main.c
 Calling hello.c
 In hello.c
 PID : 7131
 Calling add.c
 I'm in add.cAddition : 10
 back to main.cubuntu@ubuntu:~/Nivedita$
```



Significance of Using a Makefile:

- **Automation of Tasks:** A Makefile automates repetitive tasks, such as compiling, linking, and cleaning up files, ensuring consistent and efficient project builds. This reduces manual effort and streamlines the build process.
- **Dependency Management:** It manages dependencies between source files. Only the files that have changed are recompiled, which saves time and ensures that the build process is efficient**.**
- **Build Process Customization:** Makefiles allow for customization of the build process. Users can define specific rules for different actions such as compiling, linking, and cleaning, providing flexibility in handling different tasks.
- **Cross-Platform Development:** Makefiles are widely used in cross-platform development. They offer a consistent build process across various operating systems, ensuring that the same build commands work in different environments.
- **Simplifies Complex Projects:** In large projects with many files, a Makefile consolidates all the instructions required to build the project into a single file. This simplifies the process of managing and building large codebases.
- **Error Reduction:** By automating tasks and handling dependencies, Makefiles reduce the likelihood of human error, particularly in complex, multi-step build processes where manual execution could lead to mistakes.