UTS
UNIVERSITY OF TECHNOLOGY SYDNEY

LAB00 TASK 2

ED ENVIRONMENT, CONFIGURATIONS, HELLO WORLD PROGRAM

BEESHANGA ABEWARDANA JAYAWICKRAMA

UTS:
ENGINEERING AND INFORMATION TECHNOLOGY

STEPS IN CREATING A C PROGRAM

When implementing a C program we go through 3 steps.

1. Write a C program
2. Compile the program with GCC
3. Run it on the terminal

We use Ed as the C programming environment. Click on the Ed link on UTSOnline to access the Ed environment.

Following steps will guide you on how to develop your first C program in Ed environment.

WRITE – CREATE A NEW WORKSPACE

<u>STEP 1:</u> after logging into **Ed** selecting **48430 Fundamentals of C Programming** from the Dashboard, click on the **Workspaces** icon from the top menu bar.

<u>STEP 2:</u> click on the 'New Workspace' button and choose an appropriate workspace name.

The workspace name should have alphanumerical characters and underscores ONLY (NO spaces).

Suggested workspace name `48430`.

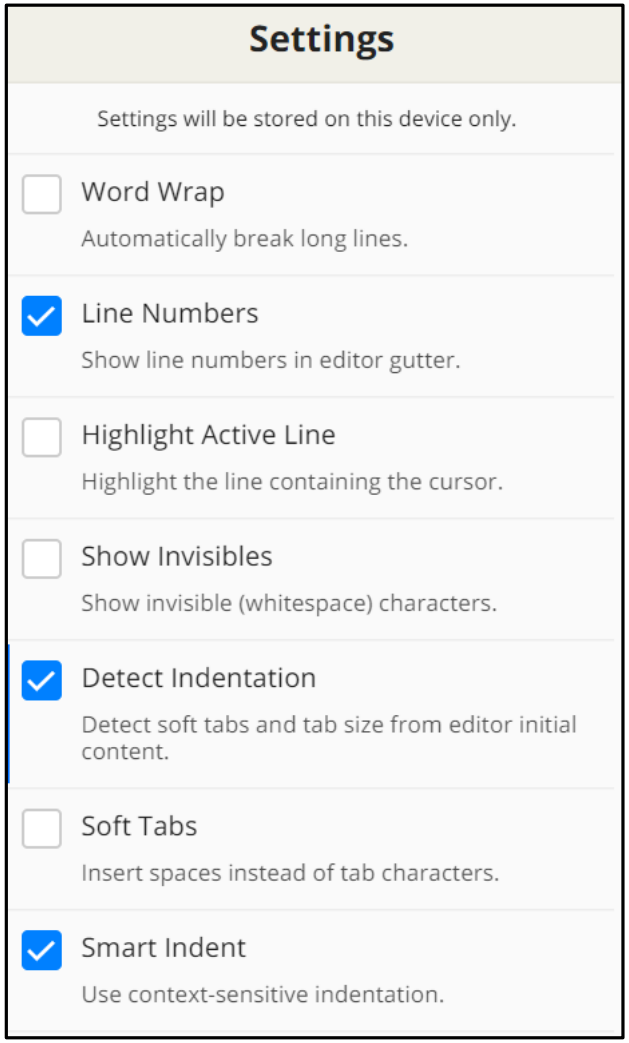
Click on the created workspace to access it.

WRITE – CONFIGURE THE EDITOR

STEP 3: turn on the following configurations in the **Settings**:

- Line numbers
- Detect Indentation
- Smart Indent

STEP 4: Set Tab Size to 4

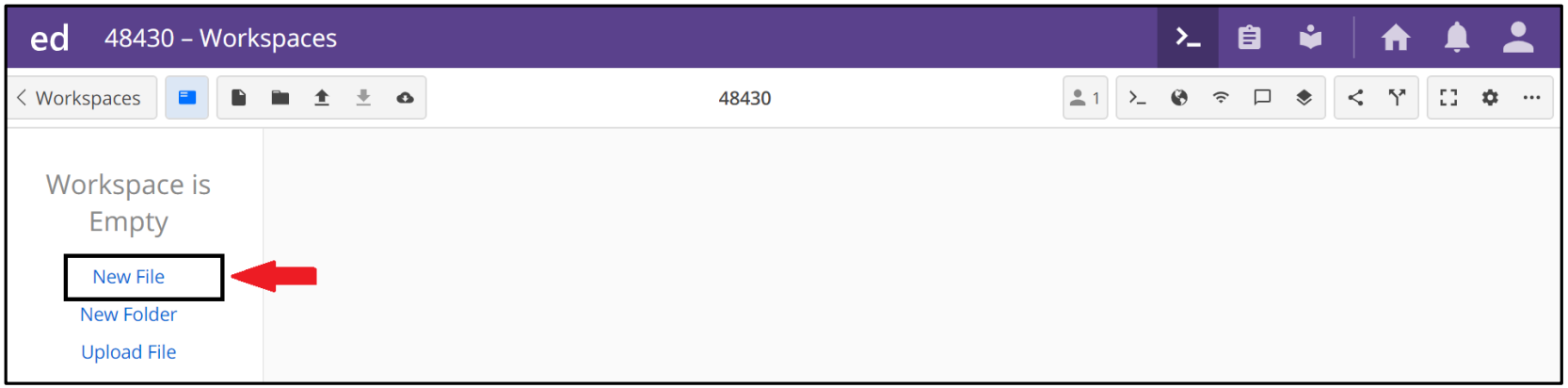


WRITE – CREATE A NEW C FILE

STEP 5: Click on **'New File'** from left hand side menu pane. Select an appropriate file name.

Suggested filename is `helloworld.c` and .c in the end is important.

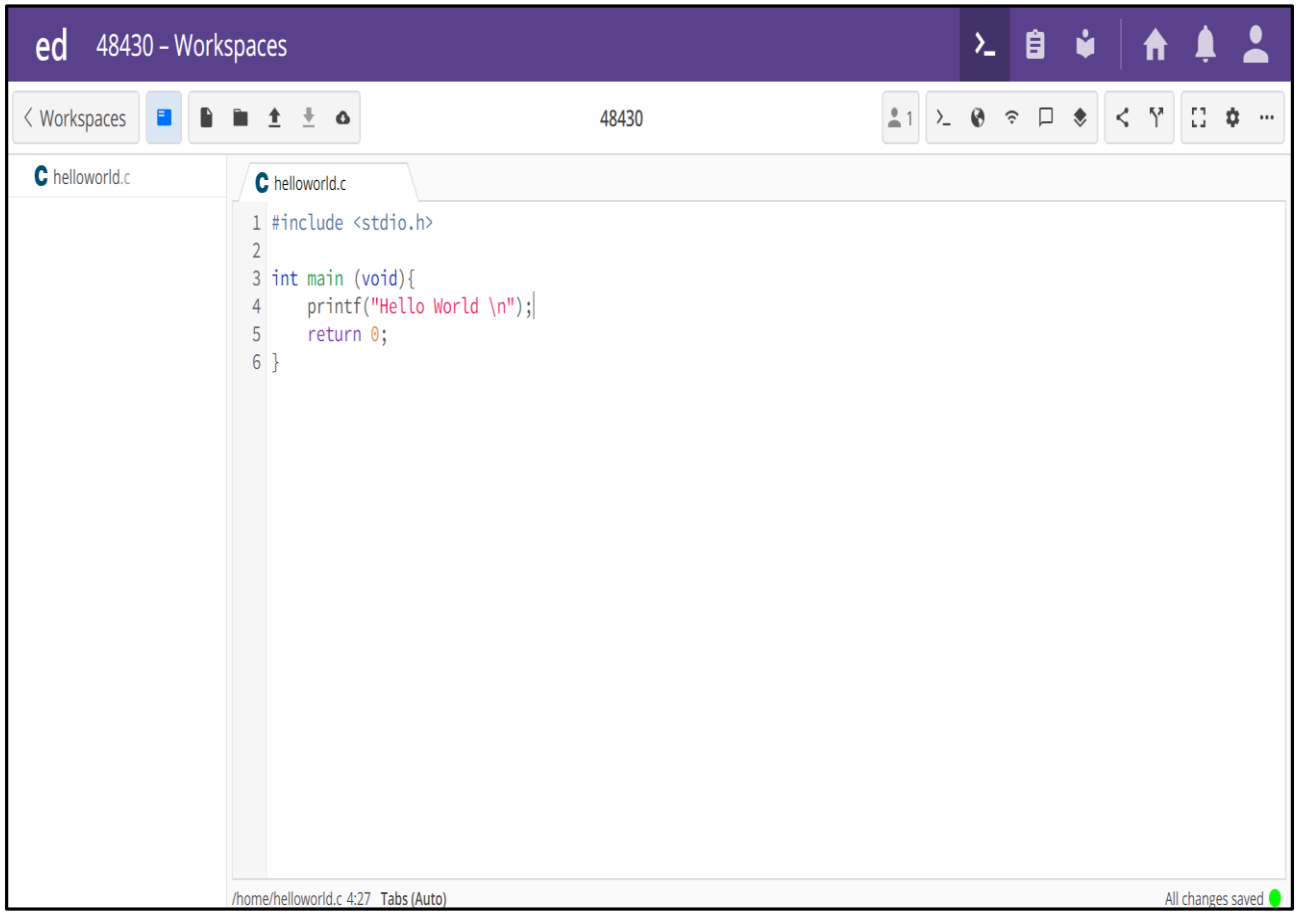
The filename should have alphanumerical characters and underscores ONLY (NO spaces).



WRITE – CREATE A NEW C FILE

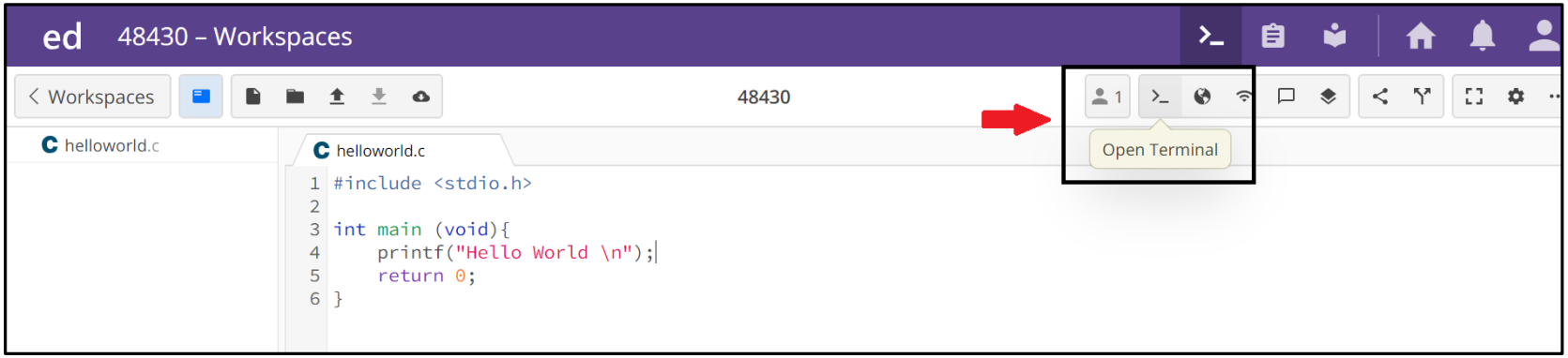
STEP 6: type in the following C program.

Manually enter the program (no copy paste).



COMPILE – OPEN A TERMINAL

STEP 7: open a new terminal.



COMPILE – COMPILE AND LINK

STEP 8 Compile and link the program as follows

```
gcc -Wall -Werror -ansi -o helloworld.out helloworld.c -lm
```

If you typed the program correctly you should NOT see any error messages. If there are any error messages go to step 6.

```
>_ user@sahara:~
[user@sahara ~]$ gcc -Wall -Werror -ansi -o helloworld.out helloworld.c -lm
[user@sahara ~]$
```

RUN

STEP 9: Run the program as follows.

```
./helloworld.out
```

The `./` in front is important.

If the program runs successfully, it should print "Hello World"

```
user@sahara:~
[user@sahara ~]$ gcc -Wall -Werror -ansi -o helloworld.out helloworld.c -lm
[user@sahara ~]$ ./helloworld.out
Hello World
[user@sahara ~]$
```

PLAY TIME

While you have both text editor and terminal open, go and change the helloworld.c file to print a different message, come back to the terminal to compile and run it.

You only have to repeat Step 6, 8 and 9 (no need to close and reopen the .c file/terminal every time you make a change).

helloworld.c

```c
#include <stdio.h>

int main (void){
    printf("Hello World \n");
    return 0;
}
```

user@sahara:~

```
[user@sahara ~]$ gcc -Wall -Werror -ansi -o helloworld.out helloworld.c -lm
[user@sahara ~]$ ./helloworld.out
Hello World
[user@sahara ~]$
```

EXERCISE I

Research more about the command we typed to compile and link the source code.

```
gcc -Wall -Werror -ansi -o helloworld.out helloworld.c -lm
```

- What do the above terms mean?
- What happens if you type:

  `gcc -Wall -Werror -ansi helloworld.c -lm`

- Why do we use:

  `-Wall -Werror -ansi` and `-lm`

- Write your own program to do further testing.

EXERCISE II

1. Understand the significance of each line in the helloworld program given in Step 6 (do research, read the textbook).

2. Understand the significance of each term in compilation syntax in Step 8 (do research, read the textbook).

3. Experiment. Change the program to print your name in a separate line after "Hello World" (do research, read the textbook, experiment).

ADVANCED WORKSPACE FUNCTIONALITIES

You can share your workspace with your friends to work on a programme together.

This will be come when you work on the group project later in the semester.

Read about ‘Sharing’ and ‘Forking’ workspaces and give it a try.