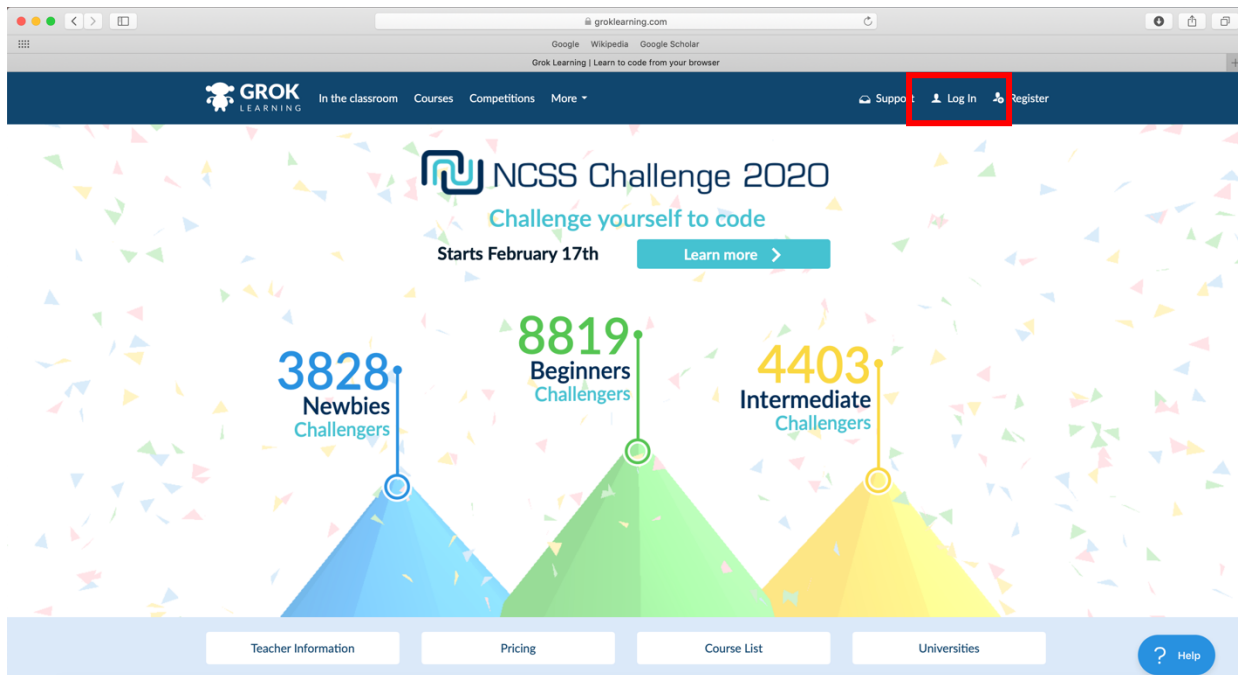


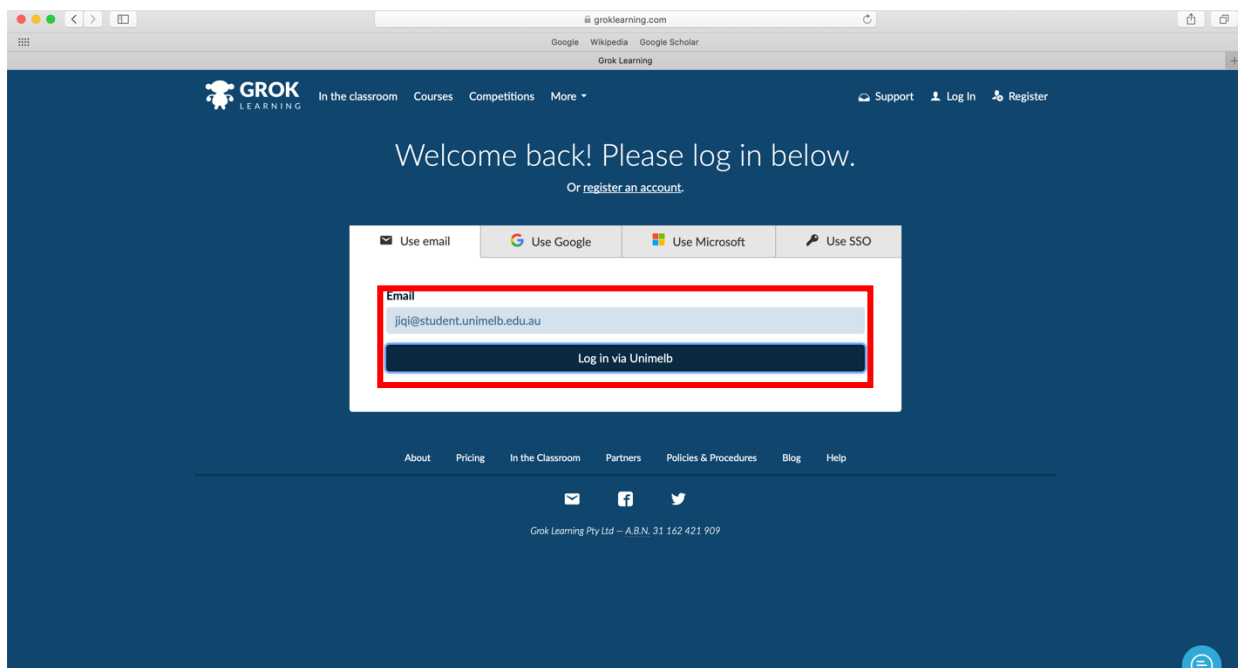
School of Computing and Information Systems
The University of Melbourne
Working with Grok

This guide illustrates the process to access and write C programs in the Grok learning system. The screenshots were based on the 2020 version of the site. The process has remained unchanged so far.

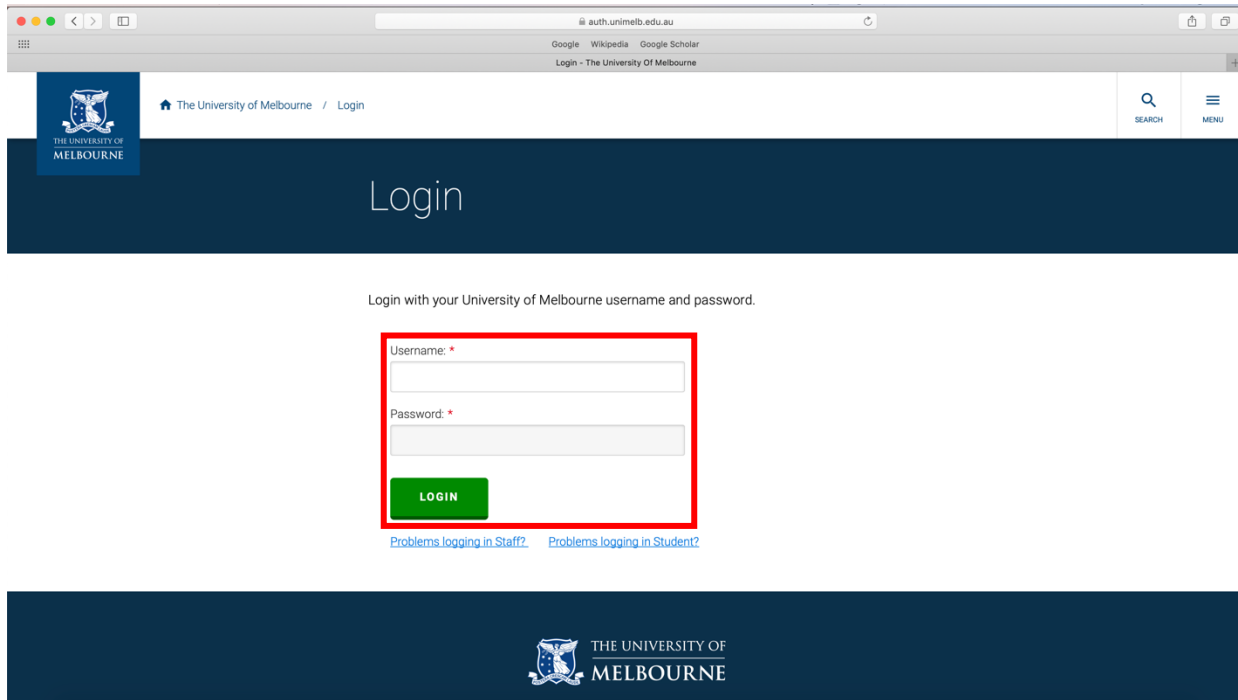
1. Navigate to the Grok website at <https://groklearning.com>; click on “Log In”.



2. Enter your unimelb student email; click on “Log in via Unimelb”.



3. Enter your unimelb username and password; click on “LOGIN”.

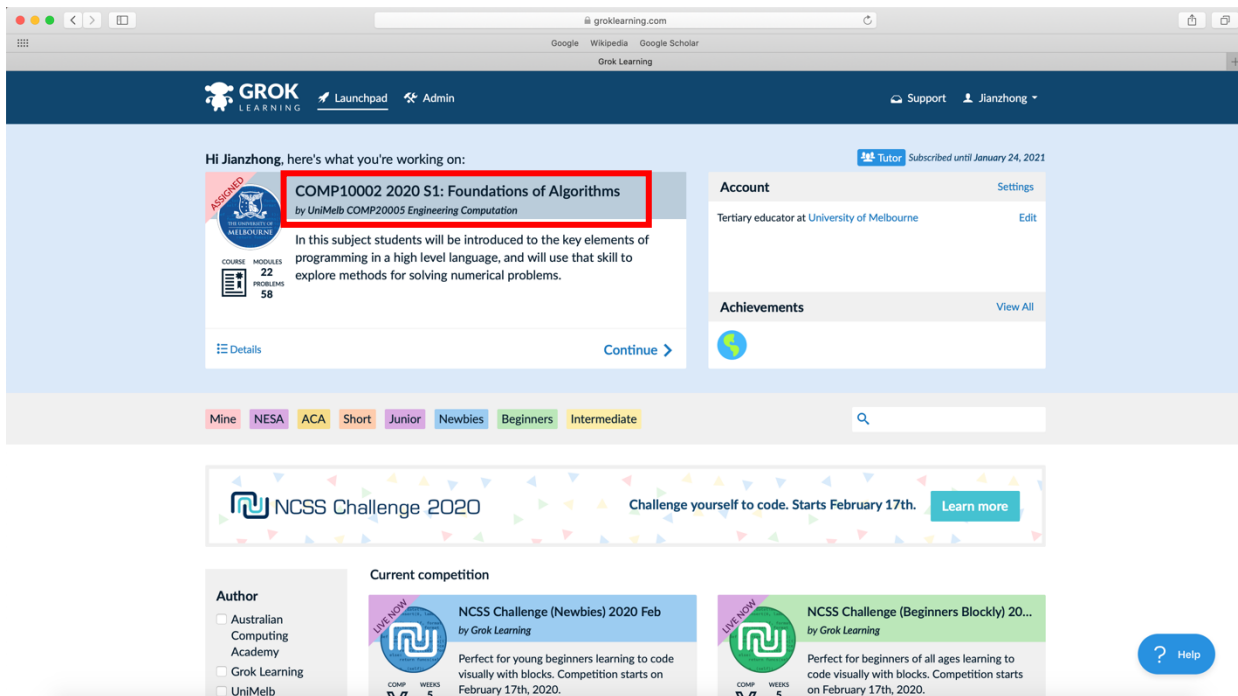


The screenshot shows the login page of the University of Melbourne. The browser address bar displays `auth.unimelb.edu.au`. The page header includes the University of Melbourne logo and navigation links. The main heading is "Login". Below it, a message states: "Login with your University of Melbourne username and password." A red rectangular box highlights the login form, which contains the following fields and elements:

- Username:** * (text input field)
- Password:** * (password input field)
- LOGIN** (green button)

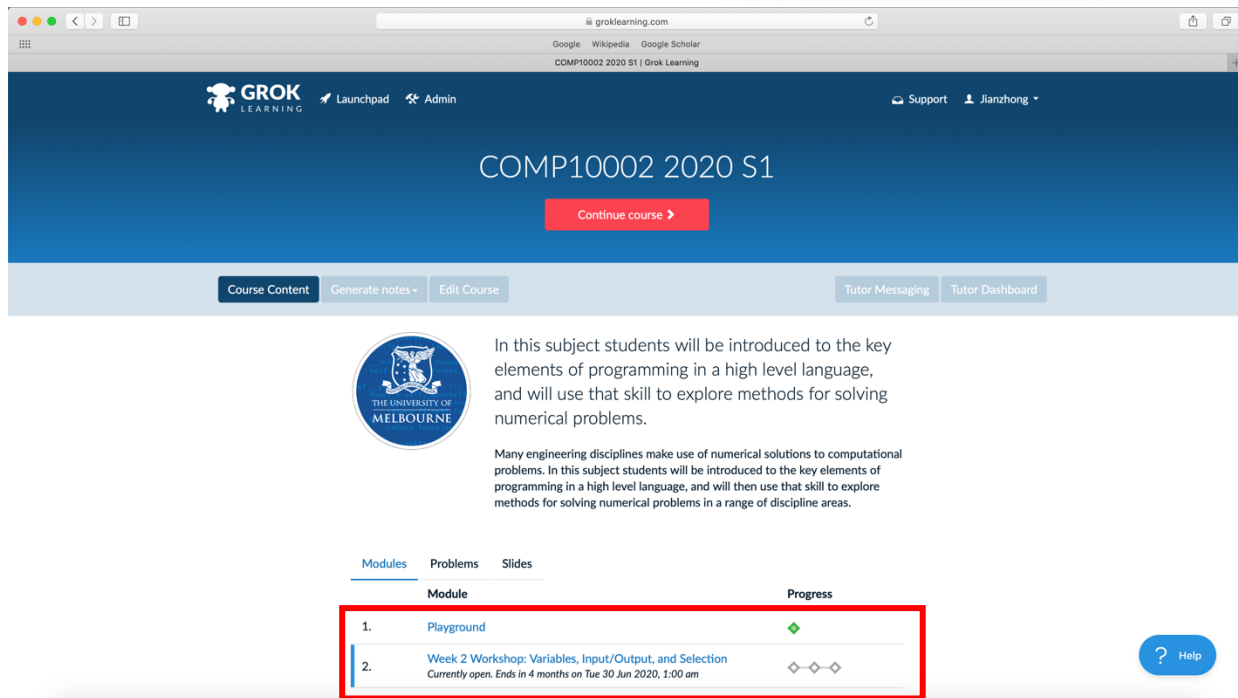
Below the login form, there are two links: [Problems logging in Staff?](#) and [Problems logging in Student?](#). At the bottom of the page, the University of Melbourne logo and name are displayed.

4. Click on “COMP10002 2021 S1: Foundations of Algorithms”.

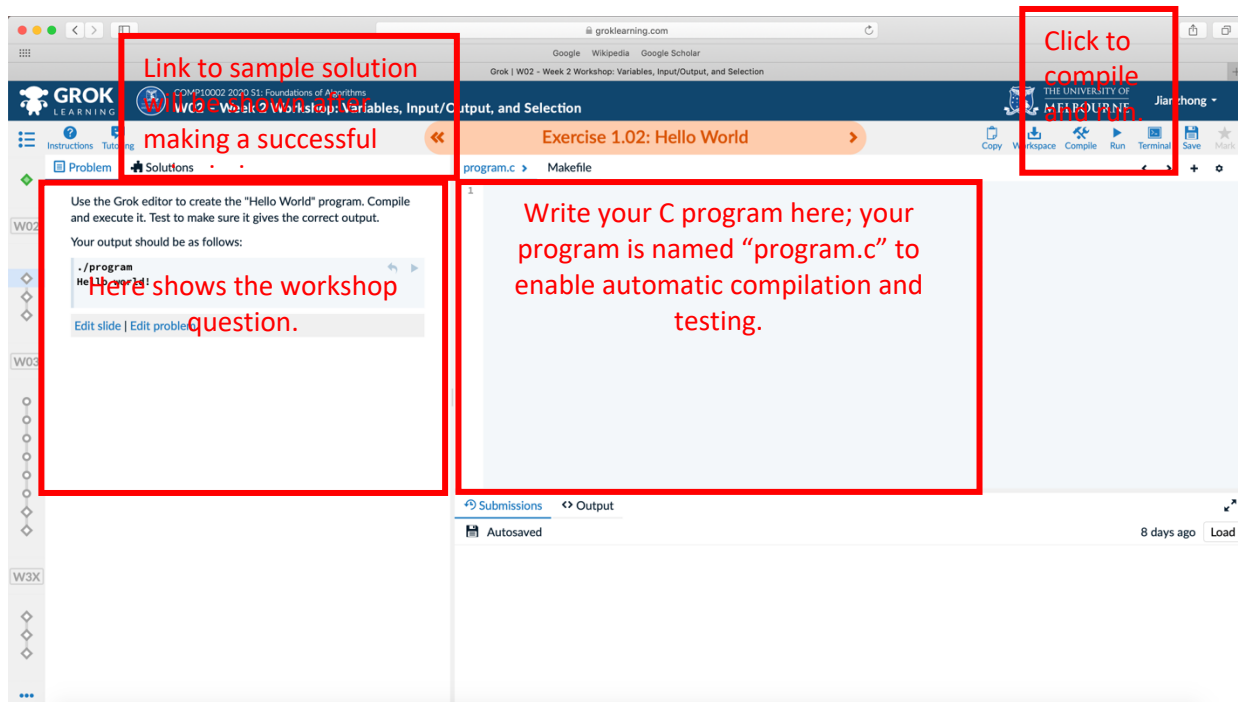


The screenshot shows the Grok Learning dashboard. The browser address bar displays `grollearning.com`. The dashboard header includes the Grok Learning logo, navigation links (Launchpad, Admin), and user information (Support, Jianzhong). The main content area displays a message: "Hi Jianzhong, here's what you're working on:". Below this message, a red rectangular box highlights the course "COMP10002 2020 S1: Foundations of Algorithms" by UniMelb COMP20005 Engineering Computation. The course description states: "In this subject students will be introduced to the key elements of programming in a high level language, and will use that skill to explore methods for solving numerical problems." The course details show 22 modules and 58 problems. A "Continue" button is visible. To the right of the course details, there is an "Account" section with a "Settings" link, and an "Achievements" section with a "View All" link. Below the course details, there is a "Current competition" section with two entries: "NCSS Challenge (Newbies) 2020 Feb" and "NCSS Challenge (Beginners Blockly) 20...". The "Newbies" challenge is described as "Perfect for young beginners learning to code visually with blocks. Competition starts on February 17th, 2020." The "Beginners Blockly" challenge is described as "Perfect for beginners of all ages learning to code visually with blocks. Competition starts on February 17th, 2020." A "Help" button is visible in the bottom right corner.

5. Click on “Week 2 Workshop: Variables, Input/Output, and Selection”. Note:
 - a. The diamonds will turn green when you have made submissions successfully.
 - b. Playground is an additional module created for you to practise C programming. You may write, compile, and run any C programs in this module.



6. Now you may write, compile, run, and submit a solution for a workshop exercise.



7. Write your solution.

Use the Grok editor to create the "Hello World" program. Compile and execute it. Test to make sure it gives the correct output.

Your output should be as follows:

```
./program
Hello world!
```

Edit slide | Edit problem

```
1 /* A first C program. Just writes a message, then exits.
2  Alistair Moffat, amoffat@unimelb.edu.au, July 2002.
3  */
4 #include <stdio.h>
5
6 int
7 main(int argc, char *argv[]) {
8     printf("Hello world!\n");
9     return 0;
10 }
11
12 /* =====
13  Program written by Alistair Moffat, as an example for the book
14  "Programming, Problem Solving, and Abstraction with C", Pearson
15  Custom Books, Sydney, Australia, 2002; revised edition 2012,
16  ISBN 9781486018974.
17
18  See http://people.eng.unimelb.edu.au/amoffat/ppsaa/ for further
19  information.
20
21  Prepared December 2012 for the Revised Edition.
22  */
```

Submissions ↔ Output

Autosaved a few seconds ago Load

8. Compile.

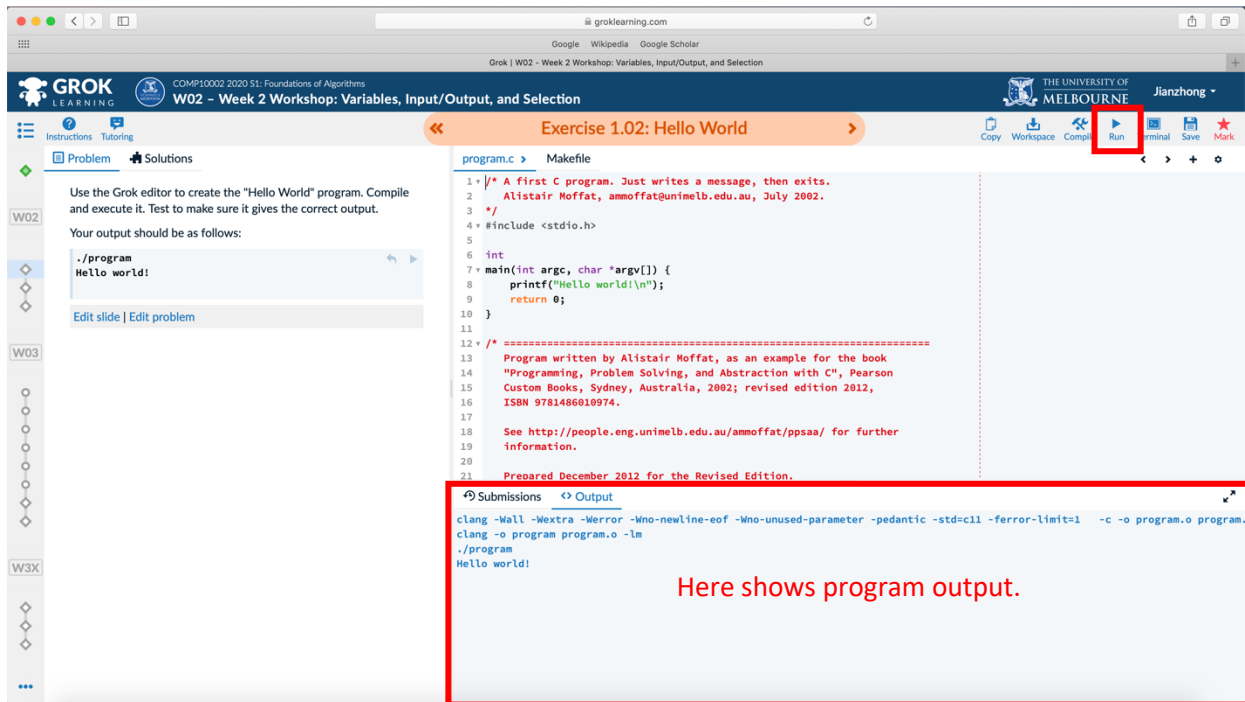
Here links to the Makefile to automate the compilation process. No need to worry about it.

Compile

```
clang -Wall -Wextra -Werror -Wno-newline-eof -Wno-unused-parameter -pedantic -std=c11 -ferror-limit=1 -c -o program.o program.c
clang -o program program.o -lm
```

Here shows the compilation command and output – we use a "Makefile" and a different C compiler called "clang" to automate the compilation process. You don't need to know.

9. Run.



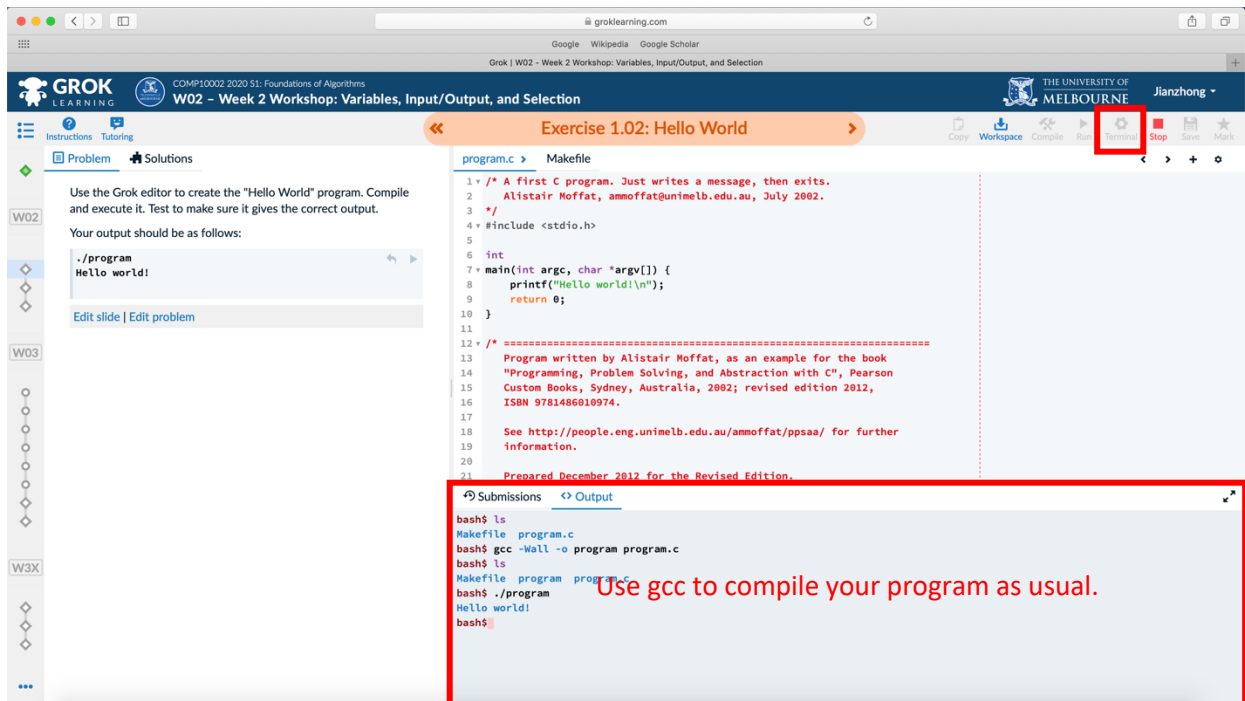
The screenshot shows the Grok Learning interface for Exercise 1.02: Hello World. The 'Run' button is highlighted in the top right. The output window shows the program's output: "Hello world!".

```
1 /* A first C program. Just writes a message, then exits.
2  Alistair Moffat, ammoffat@unimelb.edu.au, July 2002.
3  */
4 #include <stdio.h>
5
6 int
7 main(int argc, char *argv[]) {
8     printf("Hello world!\n");
9     return 0;
10 }
11
12 /* =====
13 Program written by Alistair Moffat, as an example for the book
14 "Programming, Problem Solving, and Abstraction with C", Pearson
15 Custom Books, Sydney, Australia, 2002; revised edition 2012,
16 ISBN 9781486010974.
17
18 See http://people.eng.unimelb.edu.au/ammoffat/ppsaa/ for further
19 information.
20
21 Prepared December 2012 for the Revised Edition.
```

clang -Wall -Wextra -Werror -Wno-newline-eof -Wno-unused-parameter -pedantic -std=c11 -ferror-limit=1 -c -o program.o program.c
clang -o program program.o -lm
./program
Hello world!

Here shows program output.

10. You can also compile and run your program using the “terminal”.

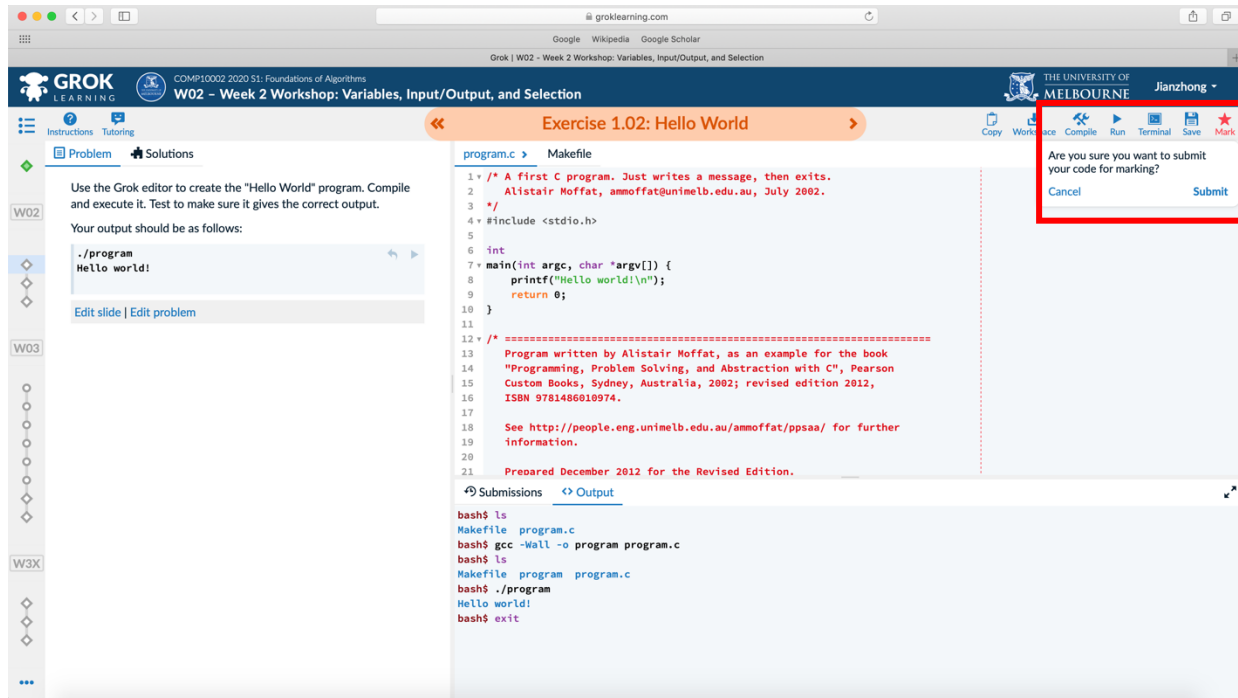


The screenshot shows the Grok Learning interface for Exercise 1.02: Hello World. The 'Terminal' button is highlighted in the top right. The terminal window shows the commands used to compile and run the program.

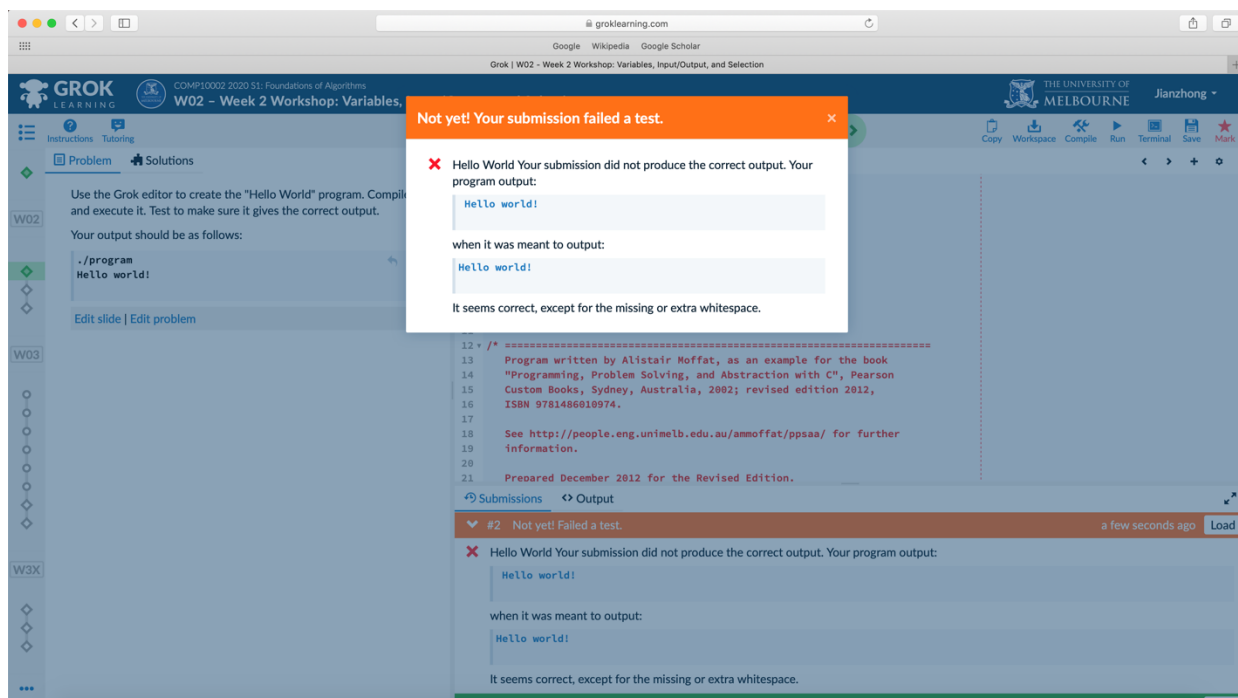
```
bash$ ls
Makefile program.c
bash$ gcc -Wall -o program program.c
bash$ ls
Makefile program program.o
bash$ ./program
Hello world!
bash$
```

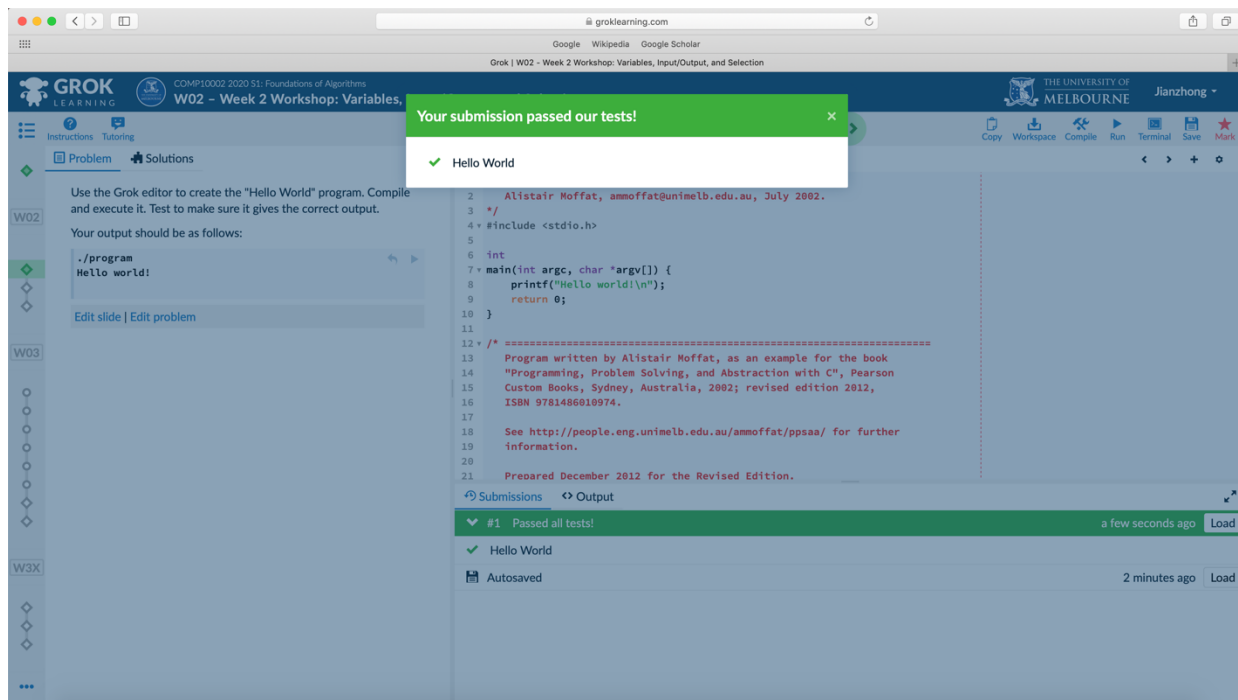
Use gcc to compile your program as usual.

11. When you are confident with your solution, click on “Mark” to submit for testing. You may submit for as many times as you want. **Your workshop submission will be tested automatically, but it will *not* be marked. Only assignment submissions will be marked.**



12. If your submission failed the tests, an error message will pop and explain the error. Otherwise, it will show that your submission passes the tests.





Note: There is no upload or download button to upload or download a C program in the Grok system. You will have to use “copy” and “paste” if you want to put a program into or get a program out from the Grok browser.