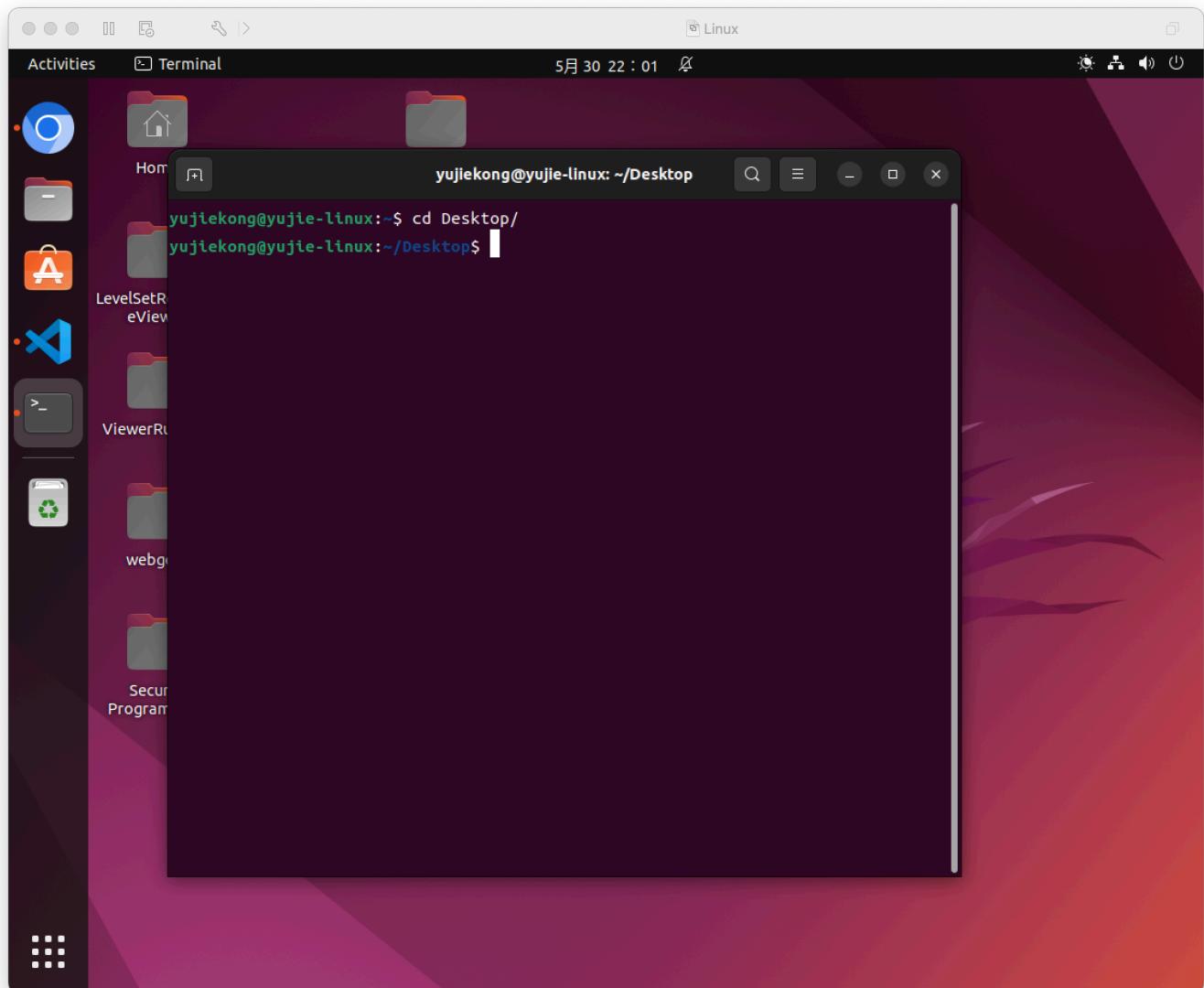
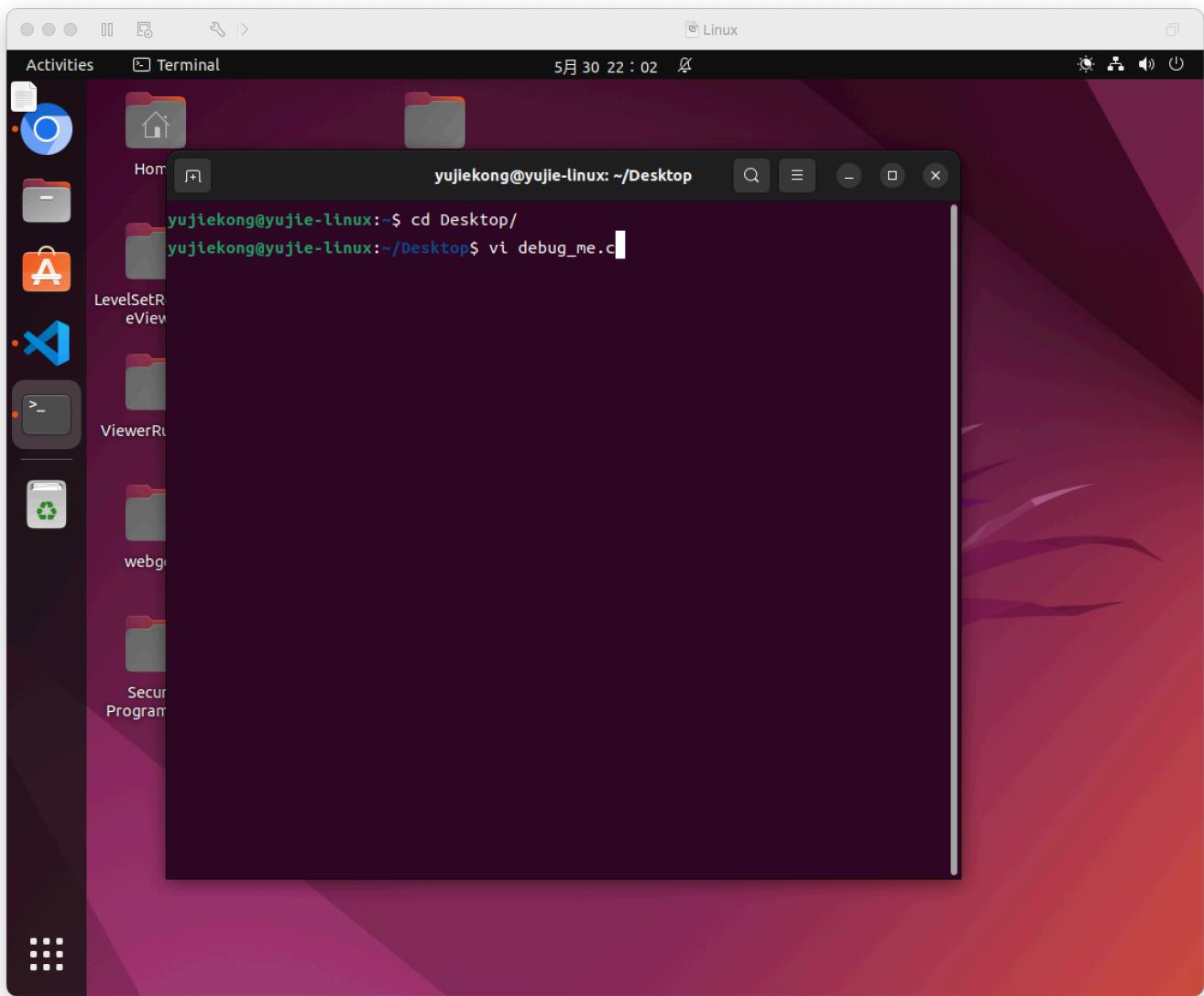


LAB2.2 Running a Hello World Program in C using GCC

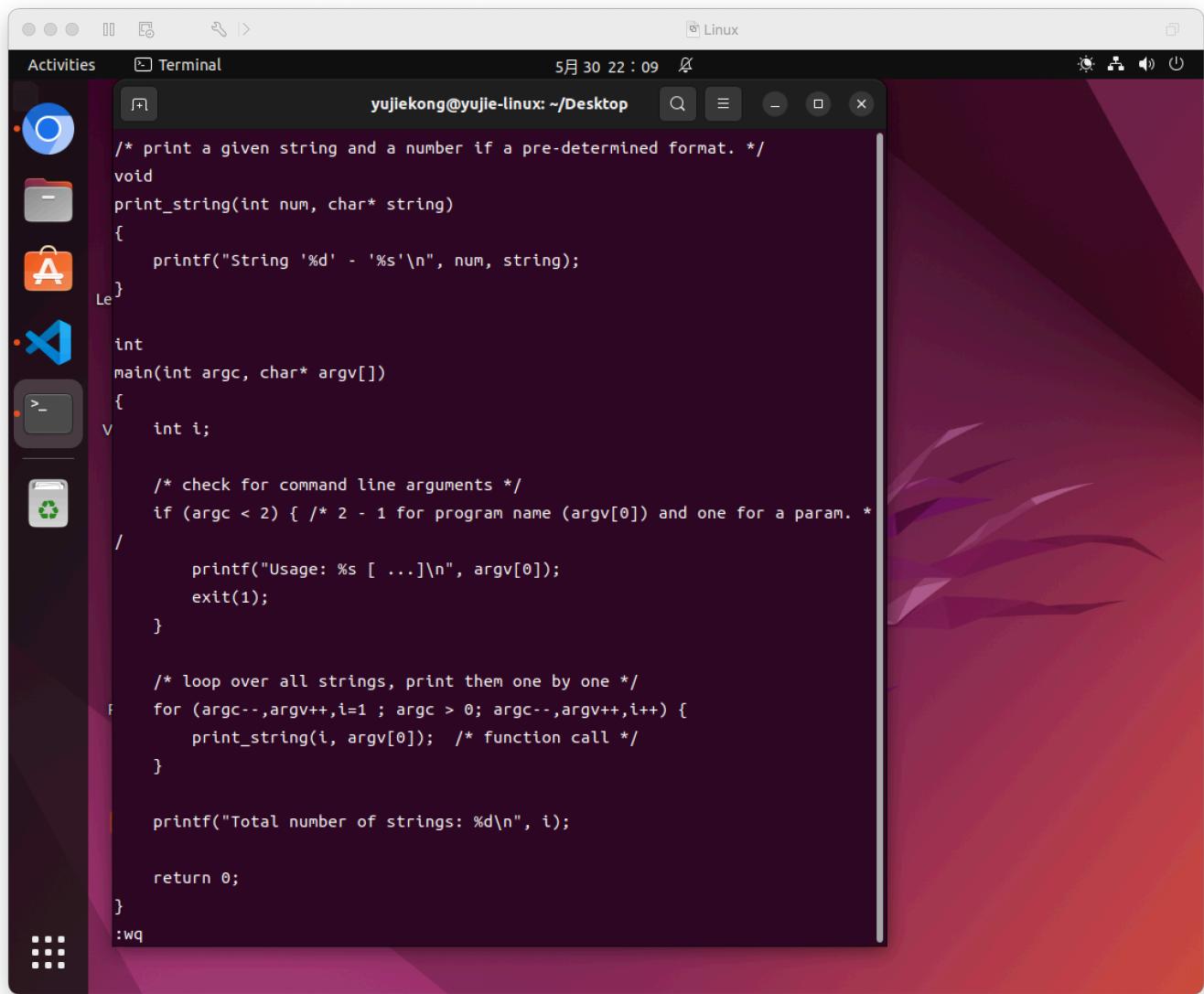
Open the Terminal in Ubuntu.



Create 'debug_me.c':



按i进入插入模式后，粘贴后保存退出：



A screenshot of a Linux desktop environment, specifically Ubuntu, showing a terminal window. The terminal window is titled "Terminal" and has the command "yujiekong@yujie-linux: ~/Desktop" and the date "5月 30 22:09". The terminal content is a C program named "debug_me.c". The code prints strings from command-line arguments. It includes a usage check for argc < 2, a loop to print all strings, and a final printf statement. The code ends with a ":wq" command to save and quit.

```
/* print a given string and a number if a pre-determined format. */
void
print_string(int num, char* string)
{
    printf("String '%d' - '%s'\n", num, string);
}

int
main(int argc, char* argv[])
{
    int i;

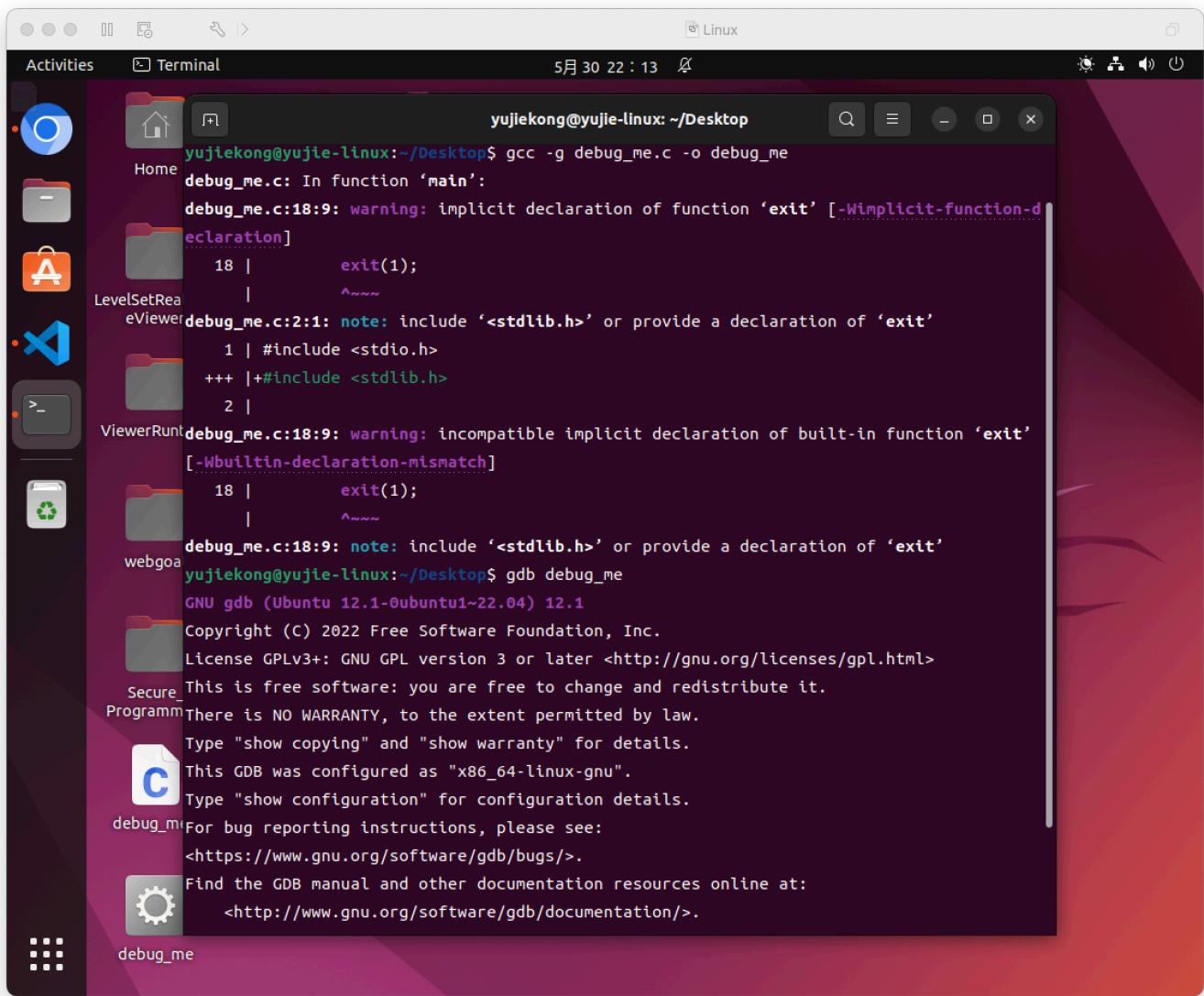
    /* check for command line arguments */
    if (argc < 2) { /* 2 - 1 for program name (argv[0]) and one for a param. */
        printf("Usage: %s [ ...]\n", argv[0]);
        exit(1);
    }

    /* loop over all strings, print them one by one */
    for (argc--, argv++, i=1 ; argc > 0; argc--, argv++, i++) {
        print_string(i, argv[0]); /* function call */
    }

    printf("Total number of strings: %d\n", i);

    return 0;
}
:wq
```

Invoke gdb to debug 'debug_me.c':

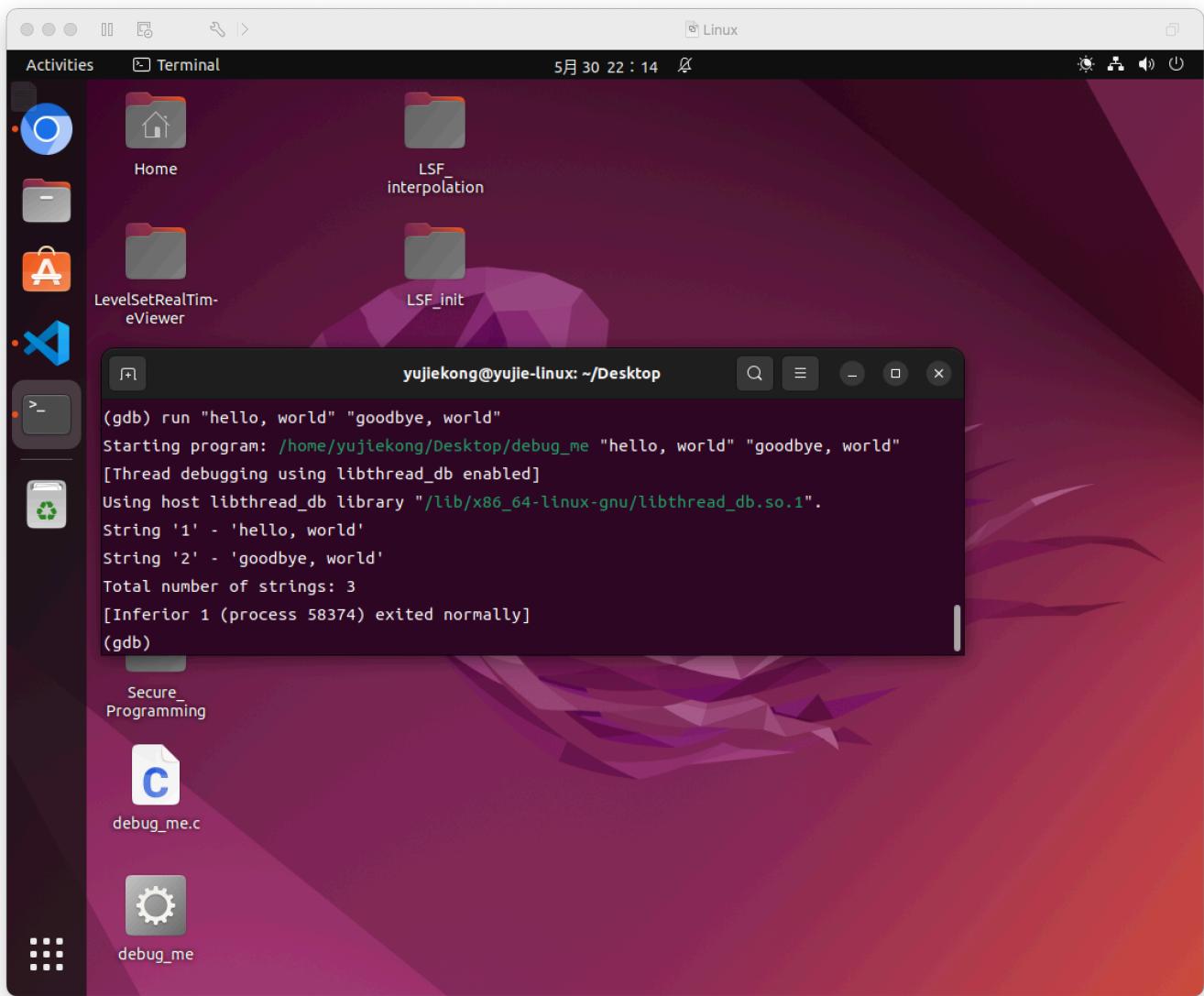


A screenshot of a Linux desktop environment, specifically Ubuntu, showing a terminal window. The terminal window is titled "Terminal" and has the command "yujiekong@yujie-linux: ~/Desktop". The terminal output shows the compilation of a C program named "debug_me.c" using the gcc compiler. The code contains several warnings about implicit declarations and incompatible implicit declarations of the "exit" function. After compilation, the command "gdb debug_me" is run, which starts the GNU Debugger (GDB) version 12.1. The GDB prompt displays standard usage information, including configuration details and documentation resources.

```
yujiekong@yujie-linux:~/Desktop$ gcc -g debug_me.c -o debug_me
debug_me.c: In function ‘main’:
debug_me.c:18:9: warning: implicit declaration of function ‘exit’ [-Wimplicit-function-declaration]
    18 |         exit(1);
          |         ^
          |
eViewerdebug_me.c:2:1: note: include ‘<stdlib.h>’ or provide a declaration of ‘exit’
   1 | #include <stdio.h>
+++ |+#include <stdlib.h>
   2 |
ViewerRuntdebug_me.c:18:9: warning: incompatible implicit declaration of built-in function ‘exit’
[-Wbuiltin-declaration-mismatch]
    18 |         exit(1);
          |         ^
          |
webgoa debug_me.c:18:9: note: include ‘<stdlib.h>’ or provide a declaration of ‘exit’
yujiekong@yujie-linux:~/Desktop$ gdb debug_me
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
Secure This is free software: you are free to change and redistribute it.
Programm There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
debug_m For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

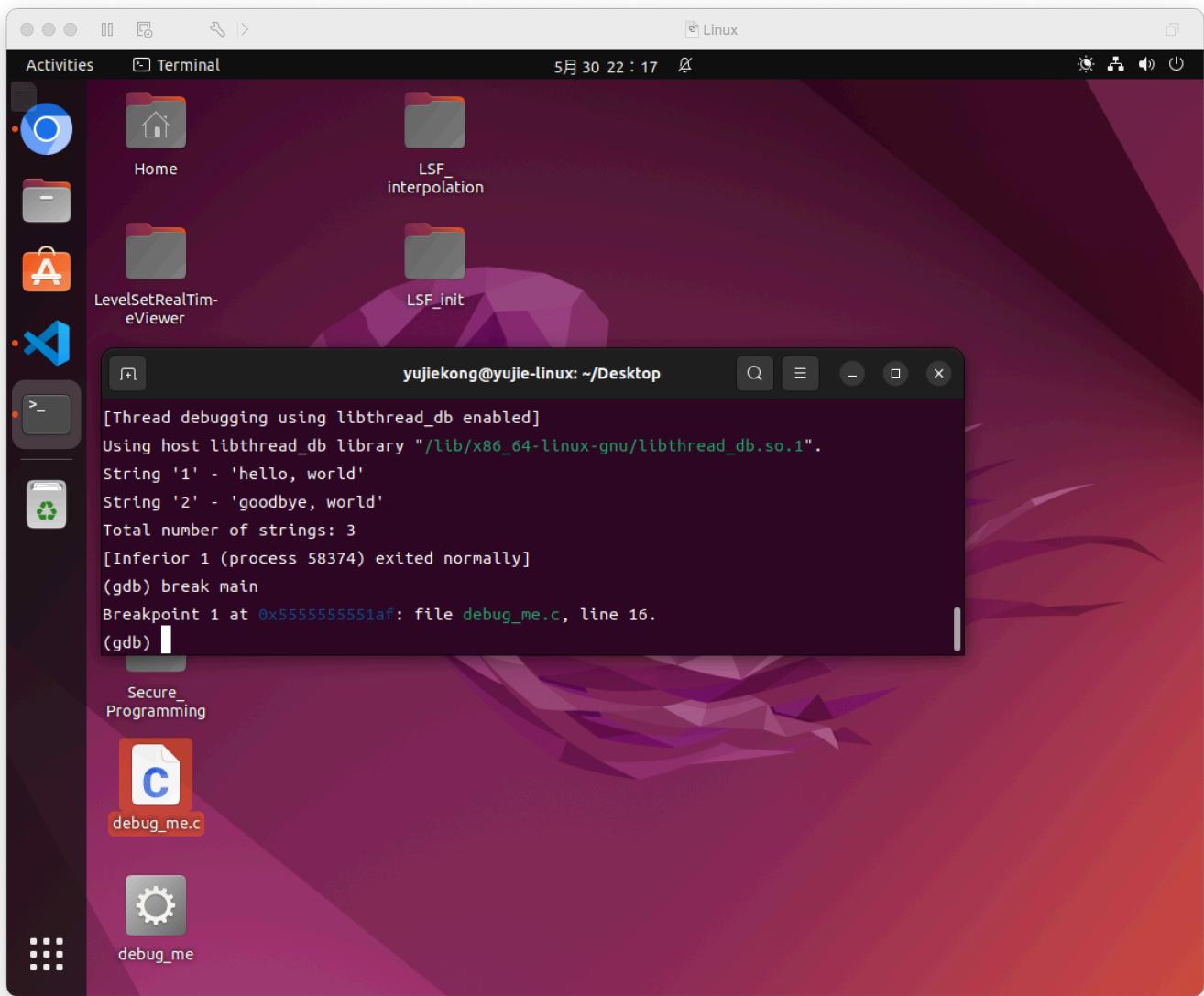
debug_me
```

Run the program inside gdb:



Set breakpoints.

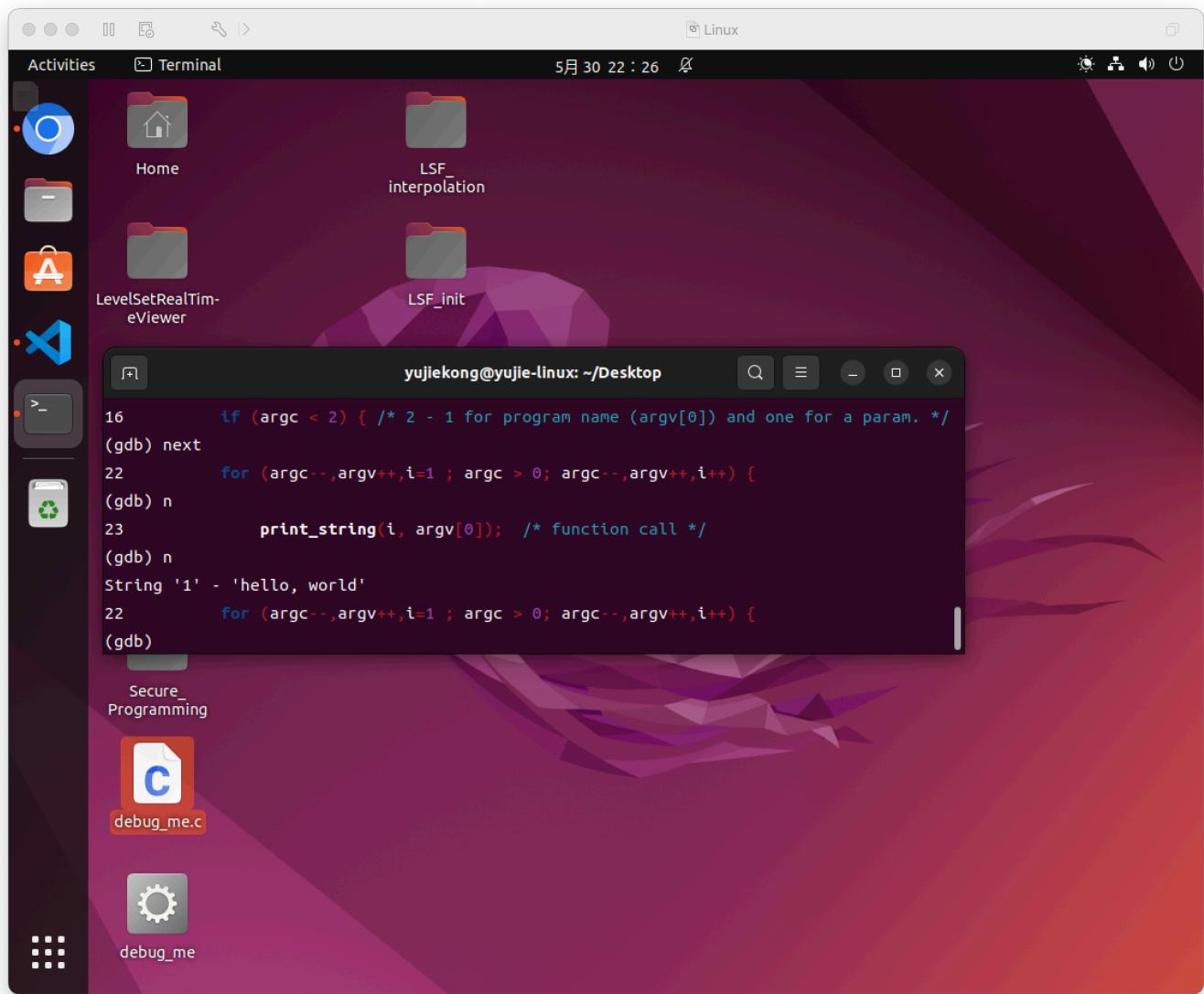
在main函数开头打断点。



Step a command at a time.

```
# 执行下一条指令，但如果该指令是一个函数调用，next 会将整个函数作为一个单元执行，而不进入函数内部。
(gdb) next
```

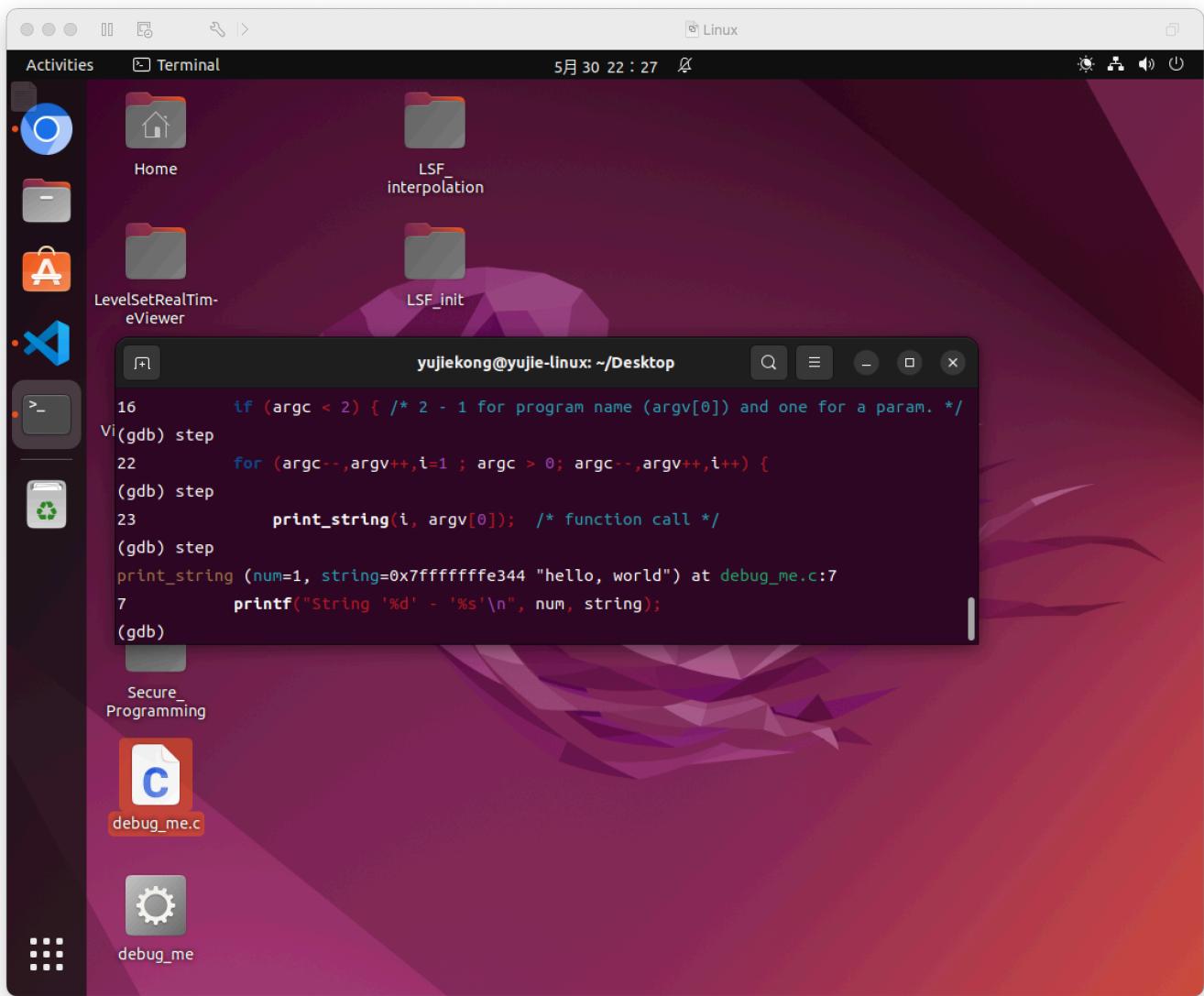
next只运行了print_string函数，没有进入函数内部。



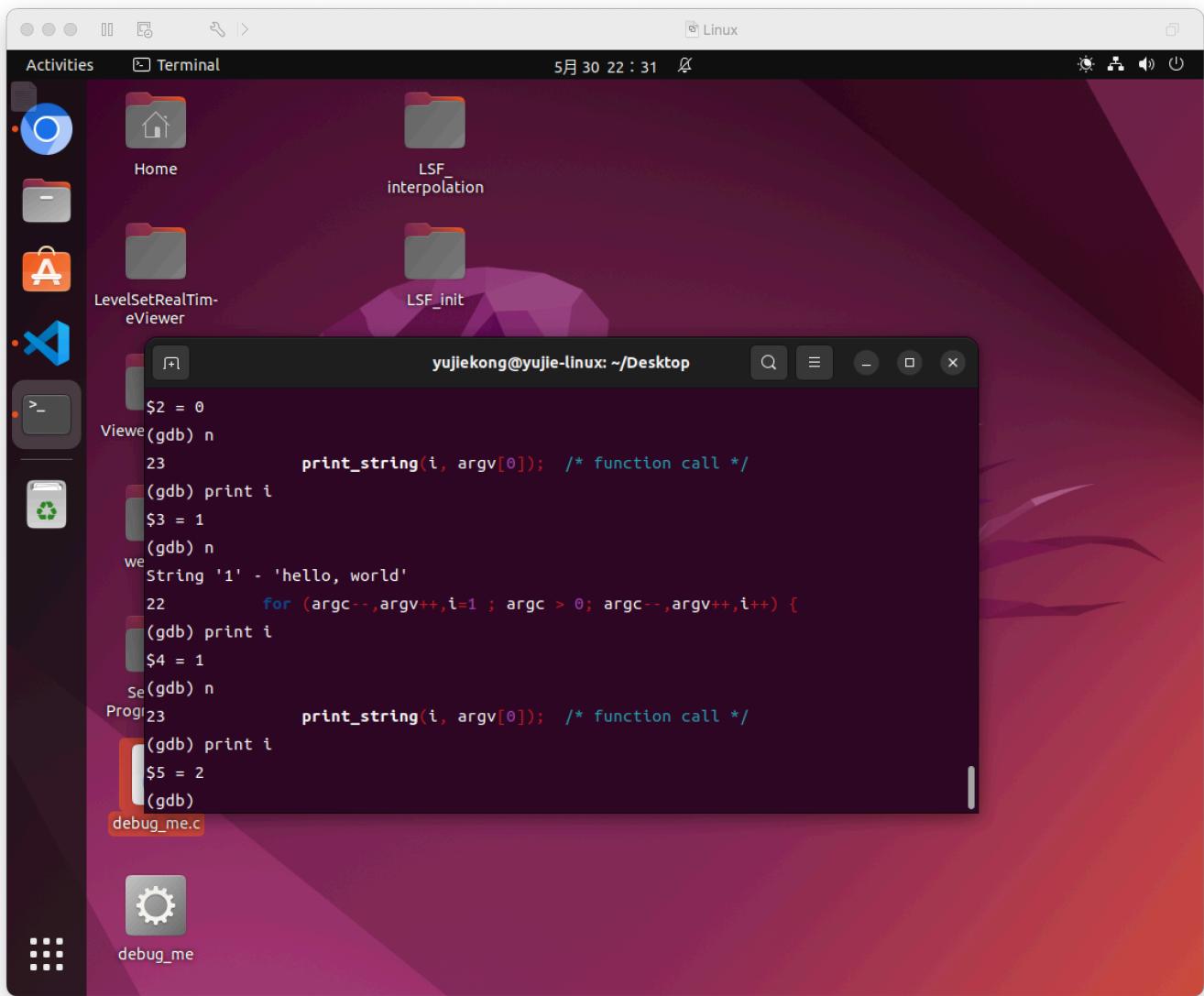
执行下一条指令，如果该指令是一个函数调用，step 会进入该函数并停在函数的第一条指令。

```
(gdb) step
```

step进入了print_string函数内部。



Print variables.



Examine the function call stack.

(1) Run the program step by step to line 7. To examine the function call stack, type:

```
(gdb) where
```

A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window is active and displays a GDB session. The session shows the execution of a C program named "debug_me.c". The program prints "hello, world" and "goodbye, world". The GDB command "Breakpoint 2" is set at line 7 of main(). The command "print_string" is used to inspect the strings. The command "frame 0" is used to switch to the main() frame, and "frame 1" is used to switch to the print_string() frame. The terminal window has a dark theme and is located on a desktop with a purple and orange gradient background. The desktop environment includes a dock with various icons like Home, LSF_interpolation, and a file manager.

```
$5 = 2
Level:(gdb) c
Continuing.
String '2' - 'goodbye, world'
Total number of strings: 3
[Inferior 1 (process 58494) exited normally]
View(gdb) break debug_me.c:7
Breakpoint 2 at 0x55555555517c: file debug_me.c, line 7.
(gdb) run "hello, world" "goodbye, world"
Starting program: /home/yujiekong/Desktop/debug_me "hello, world" "goodbye, world"
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Breakpoint 1, main (argc=3, argv=0x7fffffffdfb8) at debug_me.c:16
16          if (argc < 2) { /* 2 - 1 for program name (argv[0]) and one for a param. */
s(gdb) c
Continuing.

Breakpoint 2, print_string (num=1, string=0x7fffffffde344 "hello, world") at debug_me.c:7
7          printf("String '%d' - '%s'\n", num, string);
del(gdb) where
#0  print_string (num=1, string=0x7fffffffde344 "hello, world") at debug_me.c:7
#1  0x0000555555555203 in main (argc=2, argv=0x7fffffffdfc0) at debug_me.c:23
(gdb)
```

(2) You will see currently executing function "print_string" and the function "main" which called it. Then type "frame 0" and "frame 1" to see the difference:

```
(gdb) frame 0
(gdb) print i
...
(gdb) frame 1
(gdb) print i
```

The screenshot shows a Linux desktop environment with a dark theme. A terminal window is open, displaying a GDB session for a program named 'debug_me'. The terminal title is 'yujiekong@yujie-linux: ~/Desktop'. The session output is as follows:

```
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
e

Breakpoint 1, main (argc=3, argv=0x7fffffffdfb8) at debug_me.c:16
16          if (argc < 2) { /* 2 - 1 for program name (argv[0]) and one for a param. */
(gdb) c
ViewContinuing.

Breakpoint 2, print_string (num=1, string=0x7fffffff344 "hello, world") at debug_me.c:7
7              printf("String '%d' - '%s'\n", num, string);
(gdb) where
#0  print_string (num=1, string=0x7fffffff344 "hello, world") at debug_me.c:7
#1  0x000055555555203 in main (argc=2, argv=0x7fffffffdfc0) at debug_me.c:23
(gdb) frame 0
#0  print_string (num=1, string=0x7fffffff344 "hello, world") at debug_me.c:7
s7          printf("String '%d' - '%s'\n", num, string);
Pro(gdb) print i
No symbol "i" in current context.
(gdb) frame 1
#1  0x000055555555203 in main (argc=2, argv=0x7fffffffdfc0) at debug_me.c:23
del23          print_string(i, argv[0]); /* function call */
(gdb) print i
$6 = 1
(gdb)
```

The terminal window has a dark background with light-colored text. It includes standard Linux desktop icons in the dock on the left and a system tray at the top.

在frame 0中，实参i不存在，只有函数内部的形参num为1；在frame1中，实参i存在为1。