SP Lab 2.2

Lab: Running a Hello World Program in C using GCC

Name: 吴欣倍

StuID: 3190103044

Objectives

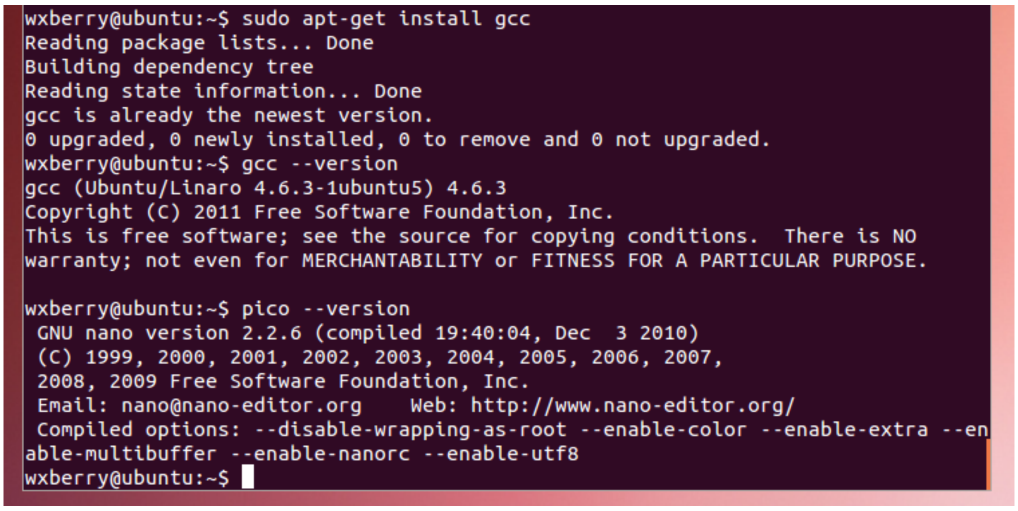
1. Learn to run a program in gcc
2. Learn to debug a program in gdb

Procedure

Part 1 : Run a "Hello World"

This part is introduced in the title of lab rather than the main content. But still do this as a first programming try in gcc in Linux.

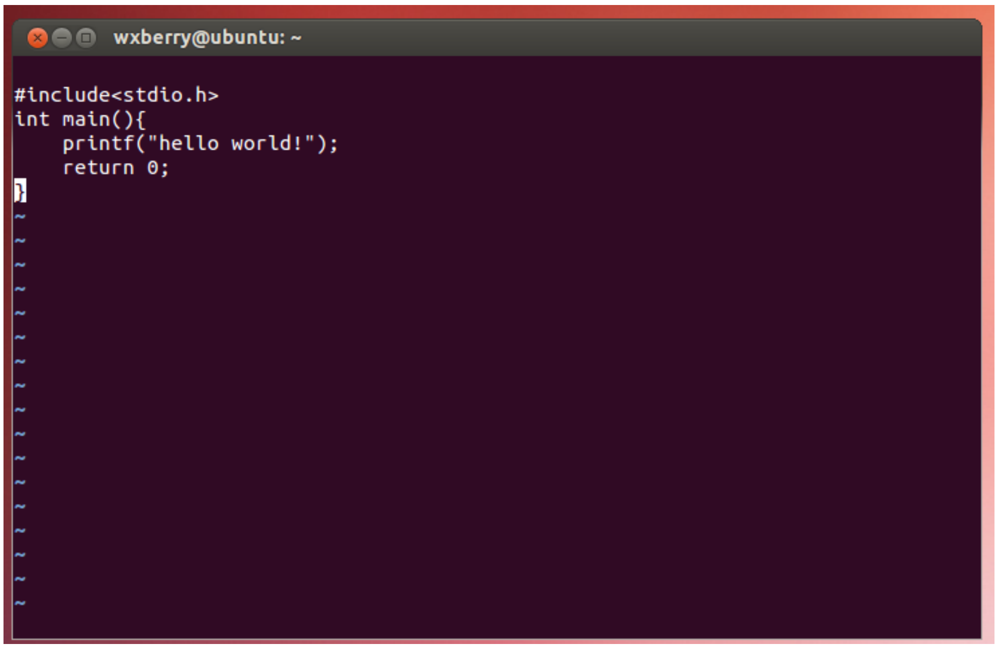
1. View the version of GCC and Pico to ensure that they are workable.



```
wxberry@ubuntu:~$ sudo apt-get install gcc
Reading package lists... Done
Building dependency tree
Reading state information... Done
gcc is already the newest version.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
wxberry@ubuntu:~$ gcc --version
gcc (Ubuntu/Linaro 4.6.3-1ubuntu5) 4.6.3
Copyright (C) 2011 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

wxberry@ubuntu:~$ pico --version
 GNU nano version 2.2.6 (compiled 19:40:04, Dec  3 2010)
 (C) 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007,
 2008, 2009 Free Software Foundation, Inc.
 Email: nano@nano-editor.org    Web: http://www.nano-editor.org/
 Compiled options: --disable-wrapping-as-root --enable-color --enable-extra --en
able-multibuffer --enable-nanorc --enable-utf8
wxberry@ubuntu:~$
```

2. Start vi editor and write a "hello word" program in C.

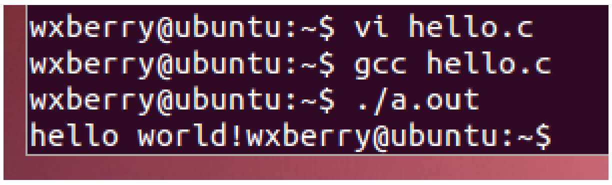
```
$vi hello.c
```



```
wxberry@ubuntu: ~
#include<stdio.h>
int main(){
    printf("hello world!");
    return 0;
}
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

3. Save and Exit. Compile and run the program.

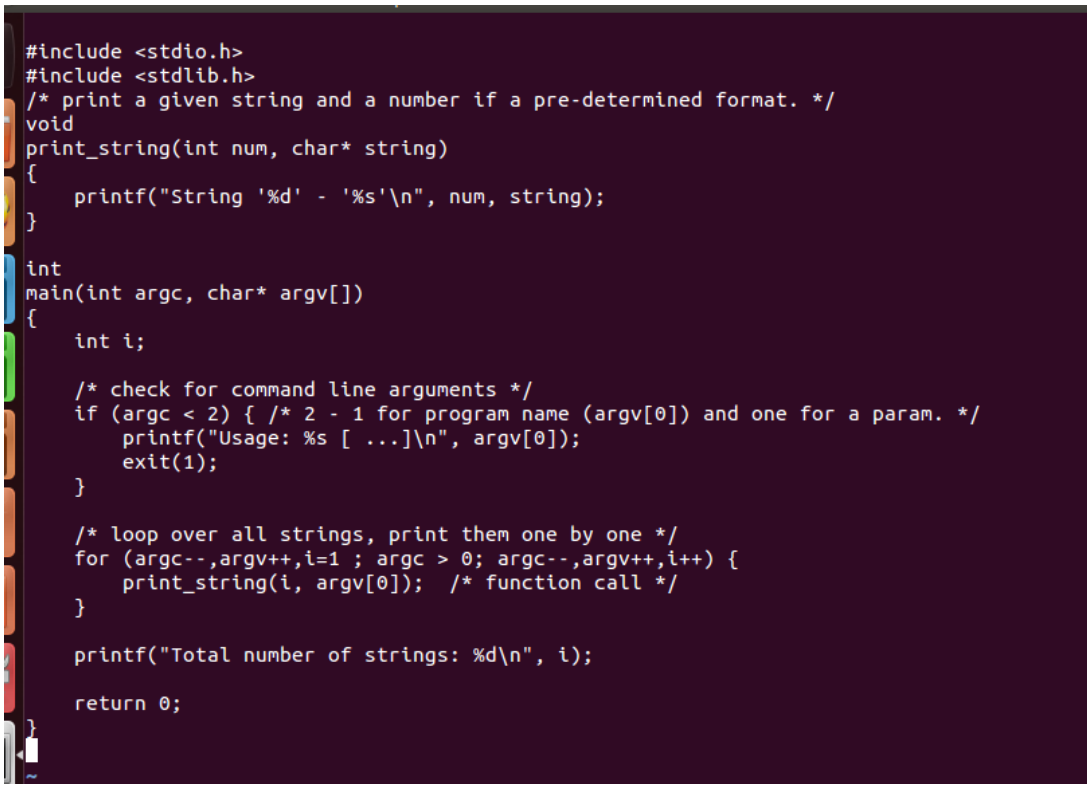
```
$gcc hello.c
$./a.out
```



```
wxberry@ubuntu:~$ vi hello.c
wxberry@ubuntu:~$ gcc hello.c
wxberry@ubuntu:~$ ./a.out
hello world!wxberry@ubuntu:~$
```

Part 2: Debug in gdb

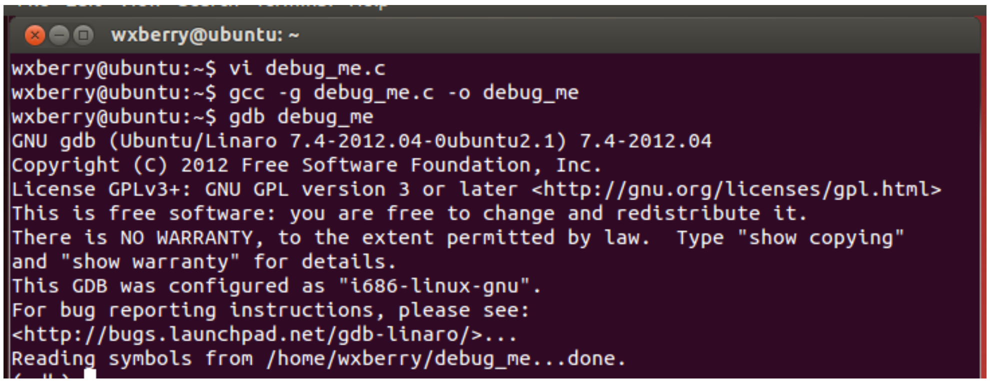
1. Open the Terminal in Ubuntu and Create 'debug_me.c'.



```
#include <stdio.h>
#include <stdlib.h>
/* 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;
}
~
```

2. Invoke gdb to debug 'debug_me.c'.

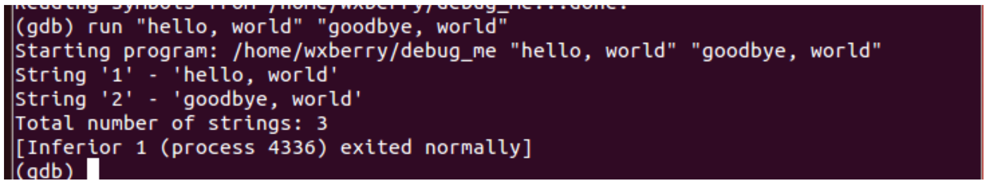
```
$gcc -g debug_me.c -o debug_me
$gdb debug_me
```



```
wxberry@ubuntu:~$ vi debug_me.c
wxberry@ubuntu:~$ gcc -g debug_me.c -o debug_me
wxberry@ubuntu:~$ gdb debug_me
GNU gdb (Ubuntu/Linaro 7.4-2012.04-0ubuntu2.1) 7.4-2012.04
Copyright (C) 2012 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "i686-linux-gnu".
For bug reporting instructions, please see:
<http://bugs.launchpad.net/gdb-linaro/>...
Reading symbols from /home/wxberry/debug_me...done.
```

3. Run the program inside gdb

```
(gdb) run "hello, world" "goodbye, world"
```



```
(gdb) run "hello, world" "goodbye, world"
Starting program: /home/wxberry/debug_me "hello, world" "goodbye, world"
String '1' - 'hello, world'
String '2' - 'goodbye, world'
Total number of strings: 3
[Inferior 1 (process 4336) exited normally]
(gdb)
```

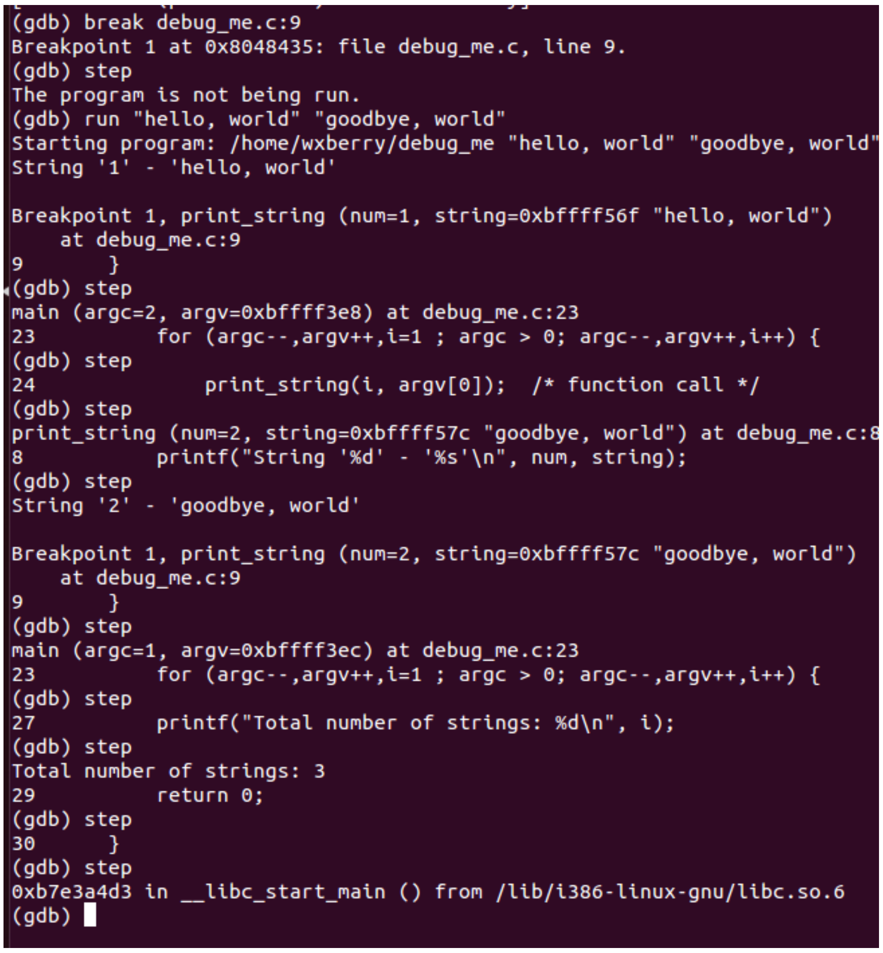
4. Set breakpoints.
   - Specifying a specific line of code to stop in:

     ```
     (gdb) break debug_me.c:9
     ```

   - Specifying a function name to break every time it is being called:

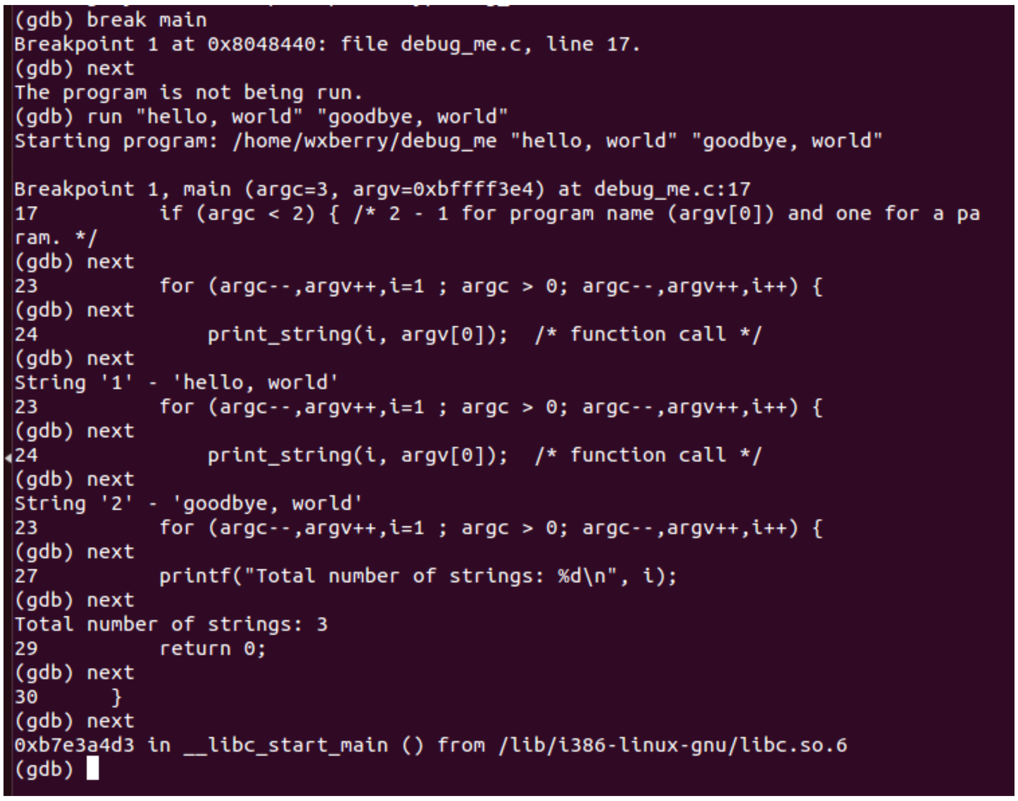
     ```
     (gdb) break main
     ```

5. Step a command at a time. Run the program step by step.
   - debug step by step:



```
(gdb) break debug_me.c:9
Breakpoint 1 at 0x8048435: file debug_me.c, line 9.
(gdb) step
The program is not being run.
(gdb) run "hello, world" "goodbye, world"
Starting program: /home/wxberry/debug_me "hello, world" "goodbye, world"
String '1' - 'hello, world'

Breakpoint 1, print_string (num=1, string=0xbffff56f "hello, world")
    at debug_me.c:9
9       }
(gdb) step
main (argc=2, argv=0xbffff3e8) at debug_me.c:23
23          for (argc--,argv++,i=1 ; argc > 0; argc--,argv++,i++) {
(gdb) step
24              print_string(i, argv[0]);  /* function call */
(gdb) step
print_string (num=2, string=0xbffff57c "goodbye, world") at debug_me.c:8
8           printf("String '%d' - '%s'\n", num, string);
(gdb) step
String '2' - 'goodbye, world'

Breakpoint 1, print_string (num=2, string=0xbffff57c "goodbye, world")
    at debug_me.c:9
9       }
(gdb) step
main (argc=1, argv=0xbffff3ec) at debug_me.c:23
23          for (argc--,argv++,i=1 ; argc > 0; argc--,argv++,i++) {
(gdb) step
27          printf("Total number of strings: %d\n", i);
(gdb) step
Total number of strings: 3
29          return 0;
(gdb) step
30      }
(gdb) step
0xb7e3a4d3 in __libc_start_main () from /lib/i386-linux-gnu/libc.so.6
(gdb) 
```

- debug by "next"



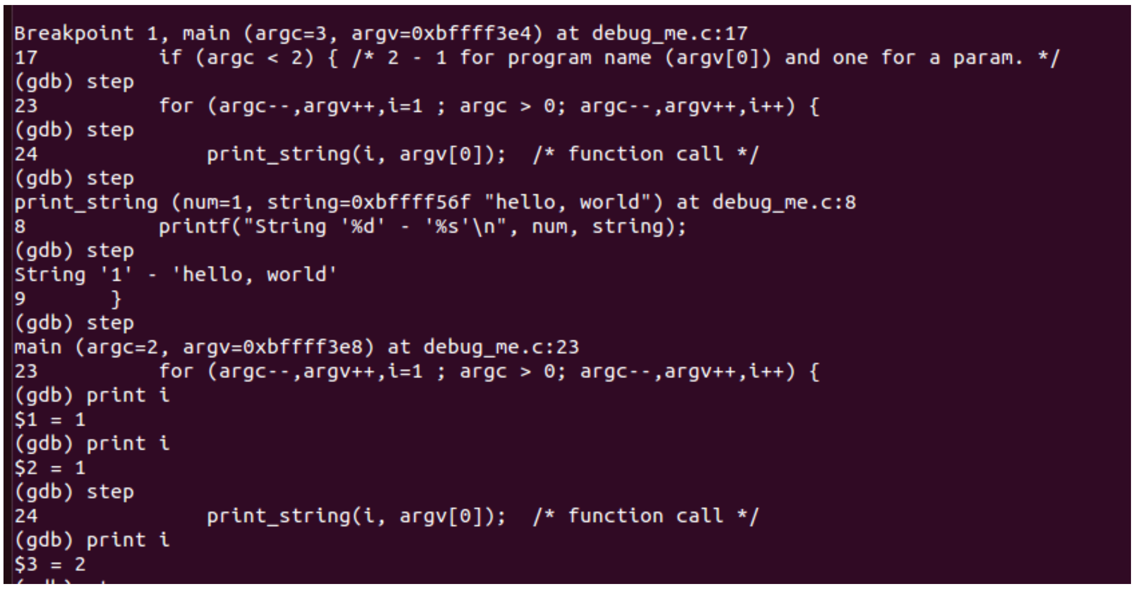
```
(gdb) break main
Breakpoint 1 at 0x8048440: file debug_me.c, line 17.
(gdb) next
The program is not being run.
(gdb) run "hello, world" "goodbye, world"
Starting program: /home/wxberry/debug_me "hello, world" "goodbye, world"

Breakpoint 1, main (argc=3, argv=0xbffff3e4) at debug_me.c:17
17          if (argc < 2) { /* 2 - 1 for program name (argv[0]) and one for a pa
ram. */
(gdb) next
23          for (argc--,argv++,i=1 ; argc > 0; argc--,argv++,i++) {
(gdb) next
24              print_string(i, argv[0]);  /* function call */
(gdb) next
String '1' - 'hello, world'
23          for (argc--,argv++,i=1 ; argc > 0; argc--,argv++,i++) {
(gdb) next
24              print_string(i, argv[0]);  /* function call */
(gdb) next
String '2' - 'goodbye, world'
23          for (argc--,argv++,i=1 ; argc > 0; argc--,argv++,i++) {
(gdb) next
27          printf("Total number of strings: %d\n", i);
(gdb) next
Total number of strings: 3
29          return 0;
(gdb) next
30      }
(gdb) next
0xb7e3a4d3 in __libc_start_main () from /lib/i386-linux-gnu/libc.so.6
(gdb) 
```

The difference between "next" and "step".

- next executes a function as a line
- step executes a function line by line

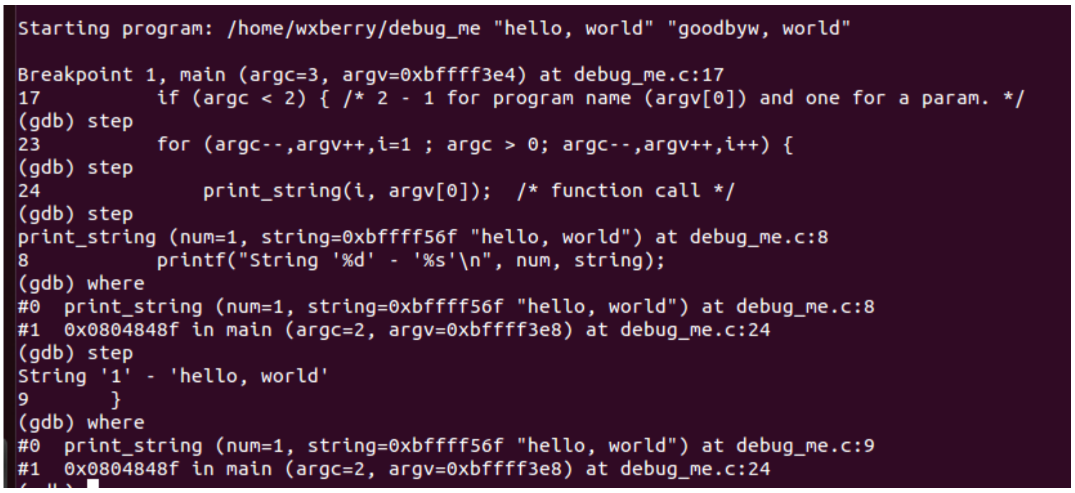
6. Print variables. When running the program step by step, print the contents of a virable with a command.

```
(gdb) print i
```



```
Breakpoint 1, main (argc=3, argv=0xbffff3e4) at debug_me.c:17
17          if (argc < 2) { /* 2 - 1 for program name (argv[0]) and one for a param. */
(gdb) step
23          for (argc--,argv++,i=1 ; argc > 0; argc--,argv++,i++) {
(gdb) step
24              print_string(i, argv[0]);  /* function call */
(gdb) step
print_string (num=1, string=0xbffff56f "hello, world") at debug_me.c:8
8           printf("String '%d' - '%s'\n", num, string);
(gdb) step
String '1' - 'hello, world'
9       }
(gdb) step
main (argc=2, argv=0xbffff3e8) at debug_me.c:23
23          for (argc--,argv++,i=1 ; argc > 0; argc--,argv++,i++) {
(gdb) print i
$1 = 1
(gdb) print i
$2 = 1
(gdb) step
24              print_string(i, argv[0]);  /* function call */
(gdb) print i
$3 = 2
```

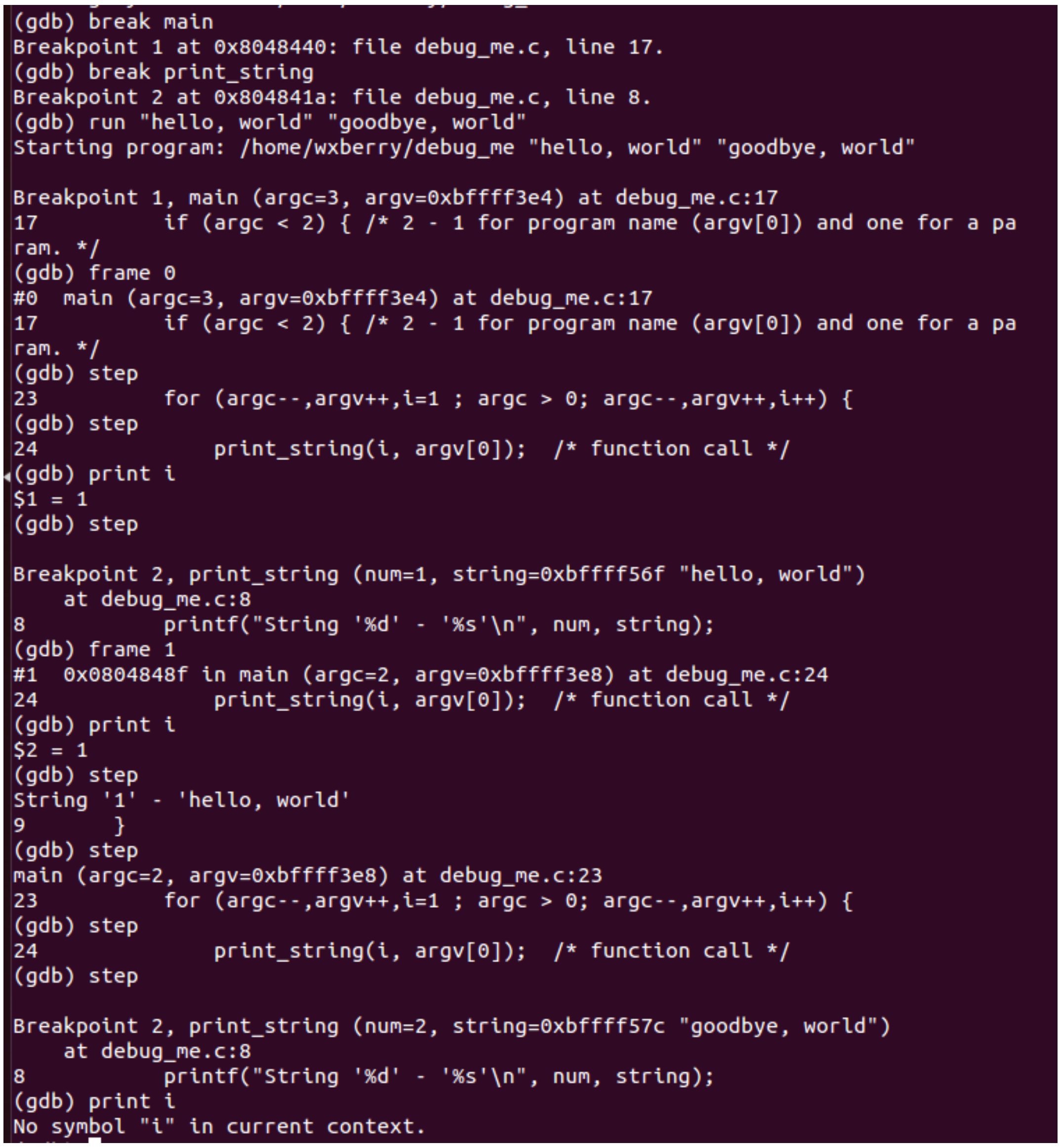
7. Examine the function call stack.
   - "where" command



```
Starting program: /home/wxberry/debug_me "hello, world" "goodbyw, world"

Breakpoint 1, main (argc=3, argv=0xbffff3e4) at debug_me.c:17
17          if (argc < 2) { /* 2 - 1 for program name (argv[0]) and one for a param. */
(gdb) step
23          for (argc--,argv++,i=1 ; argc > 0; argc--,argv++,i++) {
(gdb) step
24              print_string(i, argv[0]);  /* function call */
(gdb) step
print_string (num=1, string=0xbffff56f "hello, world") at debug_me.c:8
8           printf("String '%d' - '%s'\n", num, string);
(gdb) where
#0  print_string (num=1, string=0xbffff56f "hello, world") at debug_me.c:8
#1  0x0804848f in main (argc=2, argv=0xbffff3e8) at debug_me.c:24
(gdb) step
String '1' - 'hello, world'
9       }
(gdb) where
#0  print_string (num=1, string=0xbffff56f "hello, world") at debug_me.c:9
#1  0x0804848f in main (argc=2, argv=0xbffff3e8) at debug_me.c:24
```

   - "frame" command

     Different frames print different i(value). Because *i* isn' defined in sub-function,but in the main function , *i* is 1.



```
(gdb) break main
Breakpoint 1 at 0x8048440: file debug_me.c, line 17.
(gdb) break print_string
Breakpoint 2 at 0x804841a: file debug_me.c, line 8.
(gdb) run "hello, world" "goodbye, world"
Starting program: /home/wxberry/debug_me "hello, world" "goodbye, world"

Breakpoint 1, main (argc=3, argv=0xbffff3e4) at debug_me.c:17
17          if (argc < 2) { /* 2 - 1 for program name (argv[0]) and one for a pa
ram. */
(gdb) frame 0
#0  main (argc=3, argv=0xbffff3e4) at debug_me.c:17
17          if (argc < 2) { /* 2 - 1 for program name (argv[0]) and one for a pa
ram. */
(gdb) step
23          for (argc--,argv++,i=1 ; argc > 0; argc--,argv++,i++) {
(gdb) step
24              print_string(i, argv[0]);  /* function call */
(gdb) print i
$1 = 1
(gdb) step

Breakpoint 2, print_string (num=1, string=0xbffff56f "hello, world")
    at debug_me.c:8
8           printf("String '%d' - '%s'\n", num, string);
(gdb) frame 1
#1  0x0804848f in main (argc=2, argv=0xbffff3e8) at debug_me.c:24
24              print_string(i, argv[0]);  /* function call */
(gdb) print i
$2 = 1
(gdb) step
String '1' - 'hello, world'
9       }
(gdb) step
main (argc=2, argv=0xbffff3e8) at debug_me.c:23
23          for (argc--,argv++,i=1 ; argc > 0; argc--,argv++,i++) {
(gdb) step
24              print_string(i, argv[0]);  /* function call */
(gdb) step

Breakpoint 2, print_string (num=2, string=0xbffff57c "goodbye, world")
    at debug_me.c:8
8           printf("String '%d' - '%s'\n", num, string);
(gdb) print i
No symbol "i" in current context.
```

End