

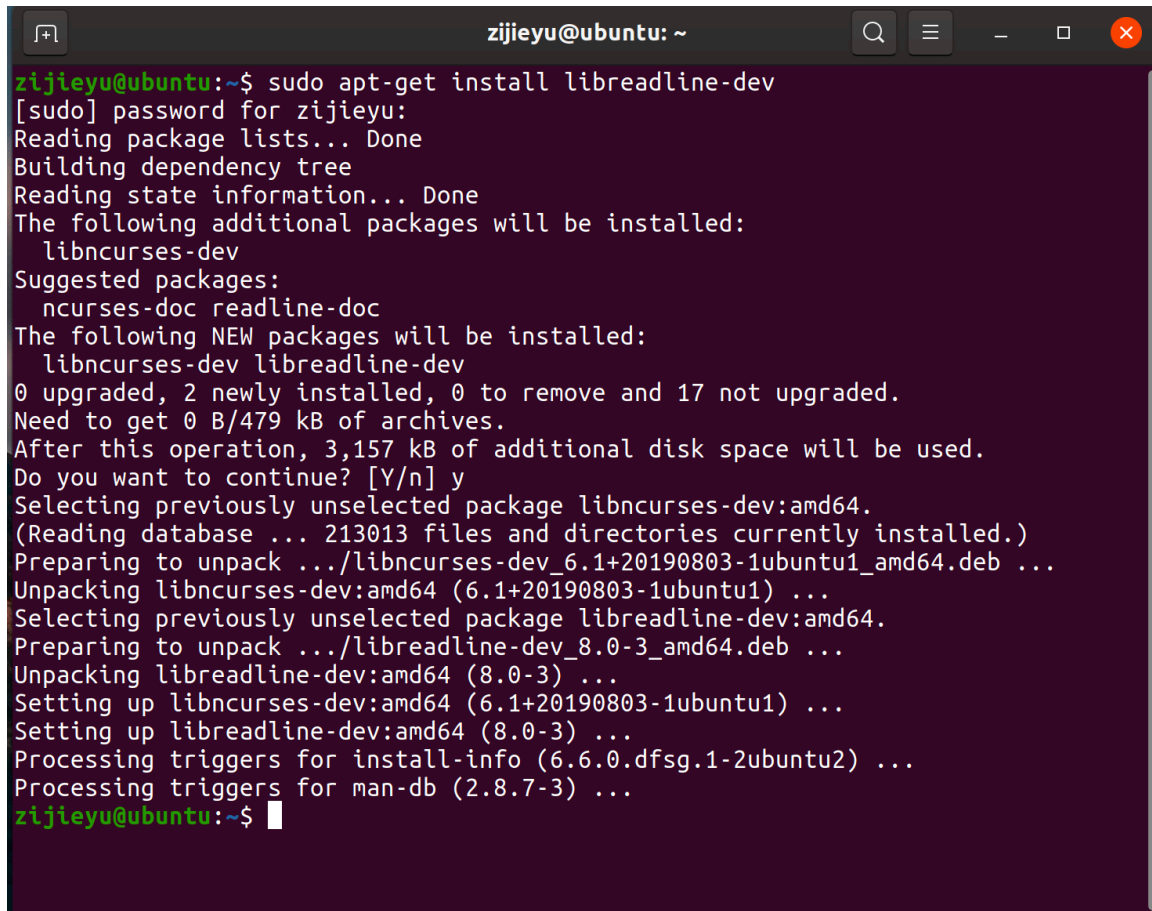
Program Author: Zijie Yu

Before running my code:

Since I use `#include <readline/readline.h>`, so if you want to run my program, you need to install **libreadline-dev** first.

Just paste below code to the shell, and then it will install automatically:

```
sudo apt-get install libreadline-dev
```

A terminal window titled 'zijieyu@ubuntu: ~' with standard window controls. The terminal shows the command 'sudo apt-get install libreadline-dev' being executed. It prompts for a password, then shows the package lists, dependency tree, and state information. It lists additional packages to be installed (libncurses-dev) and suggested packages (ncurses-doc, readline-doc). It shows the disk space requirements and asks for confirmation to continue. The installation proceeds, showing the unpacking and setting up of libncurses-dev and libreadline-dev. The terminal ends with the prompt 'zijieyu@ubuntu:~\$' and a cursor.

```
zijieyu@ubuntu:~$ sudo apt-get install libreadline-dev
[sudo] password for zijieyu:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libncurses-dev
Suggested packages:
  ncurses-doc readline-doc
The following NEW packages will be installed:
  libncurses-dev libreadline-dev
0 upgraded, 2 newly installed, 0 to remove and 17 not upgraded.
Need to get 0 B/479 kB of archives.
After this operation, 3,157 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Selecting previously unselected package libncurses-dev:amd64.
(Reading database ... 213013 files and directories currently installed.)
Preparing to unpack .../libncurses-dev_6.1+20190803-1ubuntu1_amd64.deb ...
Unpacking libncurses-dev:amd64 (6.1+20190803-1ubuntu1) ...
Selecting previously unselected package libreadline-dev:amd64.
Preparing to unpack .../libreadline-dev_8.0-3_amd64.deb ...
Unpacking libreadline-dev:amd64 (8.0-3) ...
Setting up libncurses-dev:amd64 (6.1+20190803-1ubuntu1) ...
Setting up libreadline-dev:amd64 (8.0-3) ...
Processing triggers for install-info (6.6.0.dfsg.1-2ubuntu2) ...
Processing triggers for man-db (2.8.7-3) ...
zijieyu@ubuntu:~$
```

(If you want to uninstall or remove libreadline-dev after running my program:)

```
sudo apt-get remove --auto-remove libreadline-dev
```

How to run part 2:

To meet assignment requirements, use the exec function, my majority part of part 2 code is actually in part 1.c file, and part2.c is a program that would just create two process that one use exec function to call part1.c program, the other one is add some welcome sentence to the shell.

In this case, please run part 1 code first. Using command line below:

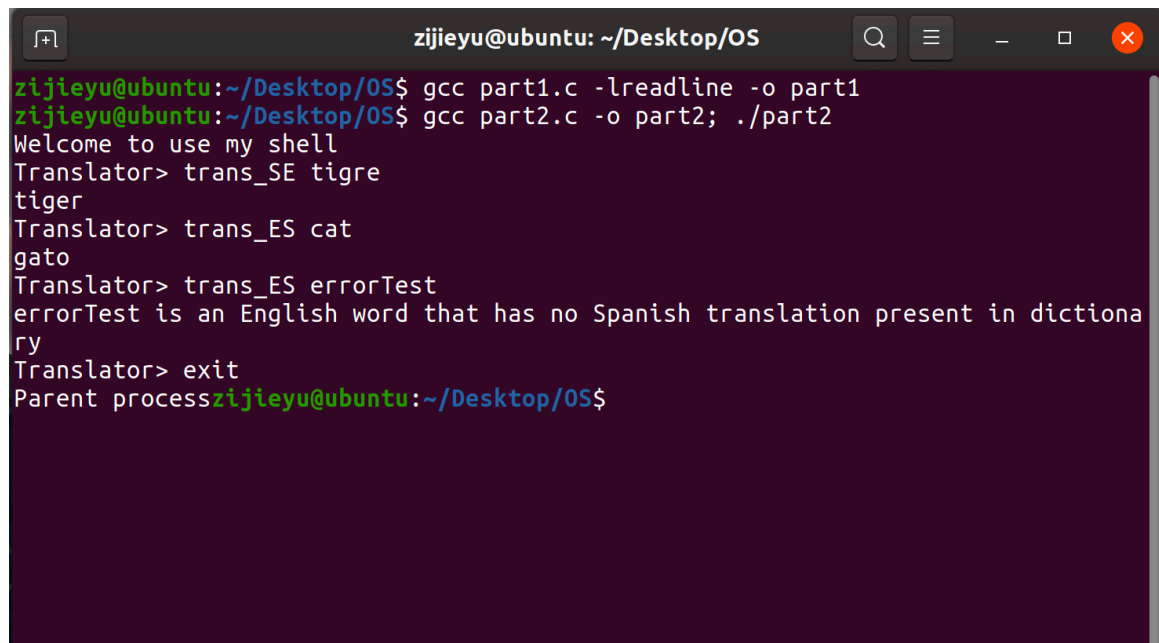
```
gcc part1.c -lreadline -o part1
```

Then it will create part1 file. Then run part 2 code:

```
gcc part2.c -o part2; ./part2
```

It will show my shell.

Output screenshot:



```
zjy@ubuntu: ~/Desktop/OS
zjy@ubuntu:~/Desktop/OS$ gcc part1.c -lreadline -o part1
zjy@ubuntu:~/Desktop/OS$ gcc part2.c -o part2; ./part2
Welcome to use my shell
Translator> trans_SE tigre
tiger
Translator> trans_ES cat
gato
Translator> trans_ES errorTest
errorTest is an English word that has no Spanish translation present in dictionary
Translator> exit
Parent process zjy@ubuntu:~/Desktop/OS$
```

How to run part 3:

Since only Part 2 asks us to use exec function, so my part 3 program is independent, just simply run one file should be fine.

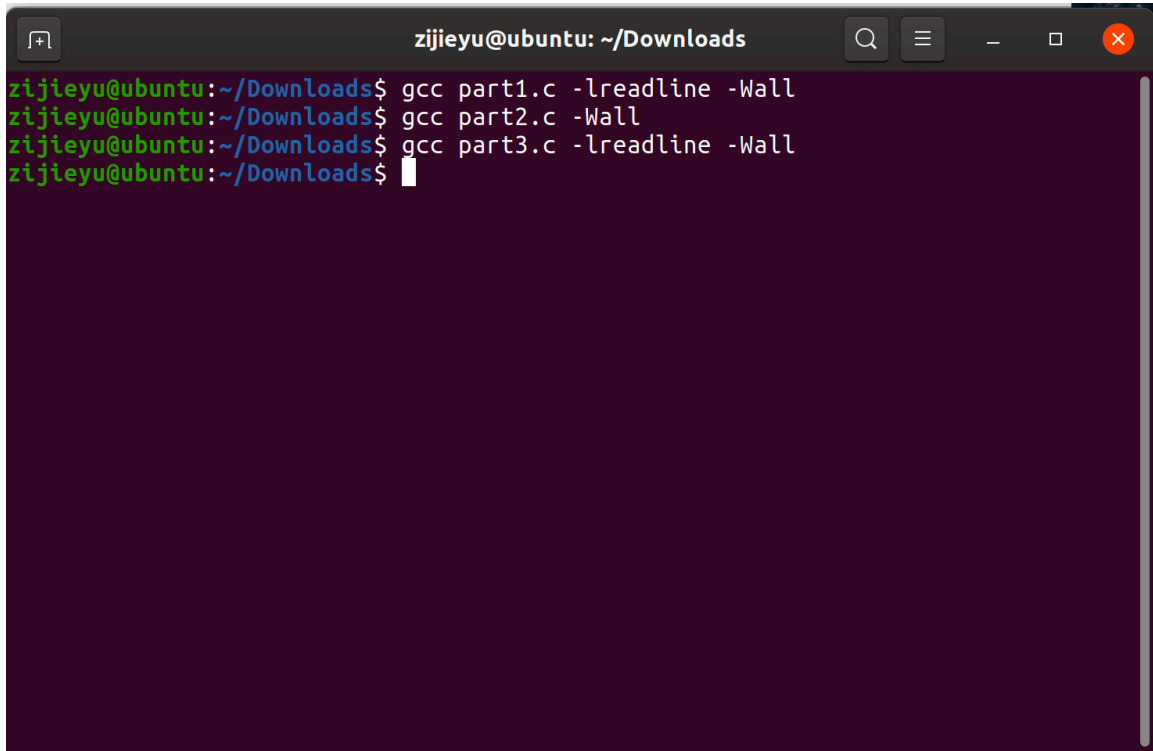
```
gcc part3.c -lreadline -o part3; ./part3
```

Output screenshot:

A terminal window titled 'zijieyu@ubuntu: ~/Desktop/1' showing the execution of the 'part3' program. The user runs 'gcc part3.c -lreadline -o part3; ./part3'. The program starts with a 'Translator>' prompt. The user enters 'trans_FS vache', and the program outputs 'vaca'. The user enters 'trans_SF mono', and the program outputs 'singe'. The user enters 'trans_SE tigre', and the program outputs 'tiger'. The user enters 'trans_ES cat', and the program outputs 'gato'. Finally, the user enters 'exit', and the program returns to the shell prompt 'zijieyu@ubuntu:~/Desktop/1\$'.

```
zijieyu@ubuntu:~/Desktop/1$ gcc part3.c -lreadline -o part3; ./part3
Translator> trans_FS vache
vaca
Translator> trans_SF mono
singe
Translator> trans_SE tigre
tiger
Translator> trans_ES cat
gato
Translator> exit
zijieyu@ubuntu:~/Desktop/1$
```

gcc -Wall test

A terminal window titled 'zijieyu@ubuntu: ~/Downloads' with standard window controls. It shows four lines of terminal output where the user compiles three C files: part1.c, part2.c, and part3.c. Each compilation command includes the flags '-lreadline' and '-Wall'. The prompt is 'zijieyu@ubuntu:~/Downloads\$' and the cursor is at the end of the fourth line.

```
zijieyu@ubuntu:~/Downloads$ gcc part1.c -lreadline -Wall
zijieyu@ubuntu:~/Downloads$ gcc part2.c -Wall
zijieyu@ubuntu:~/Downloads$ gcc part3.c -lreadline -Wall
zijieyu@ubuntu:~/Downloads$
```