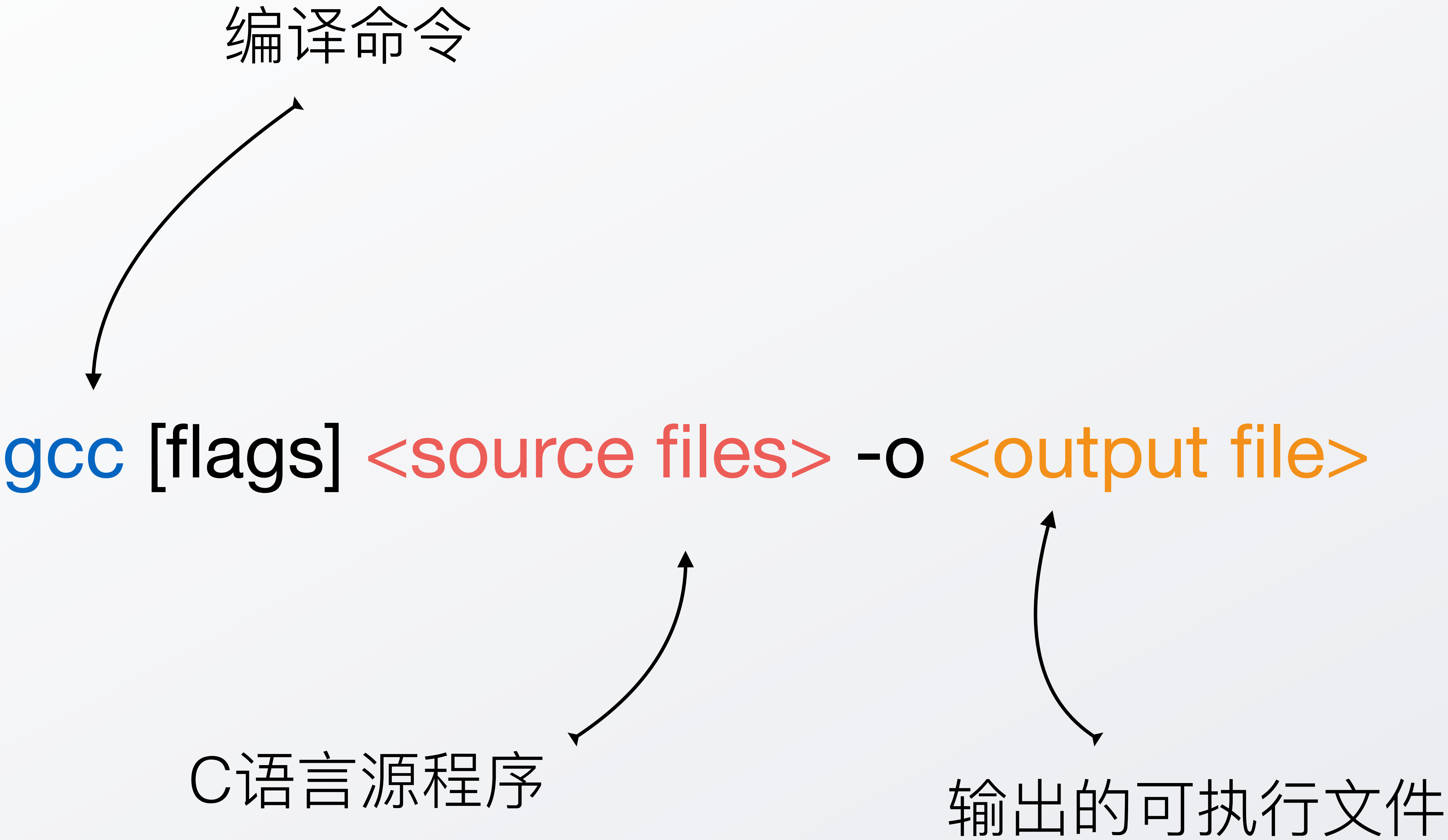
C语言程序的

编译、调试与执行

huiw@suda.edu.cn

```
gcc [flags] <source files> -o <output file>
```



编译命令

```
gcc [flags] <source files> -o <output file>
```

C语言源程序

输出的可执行文件

编译命令举例

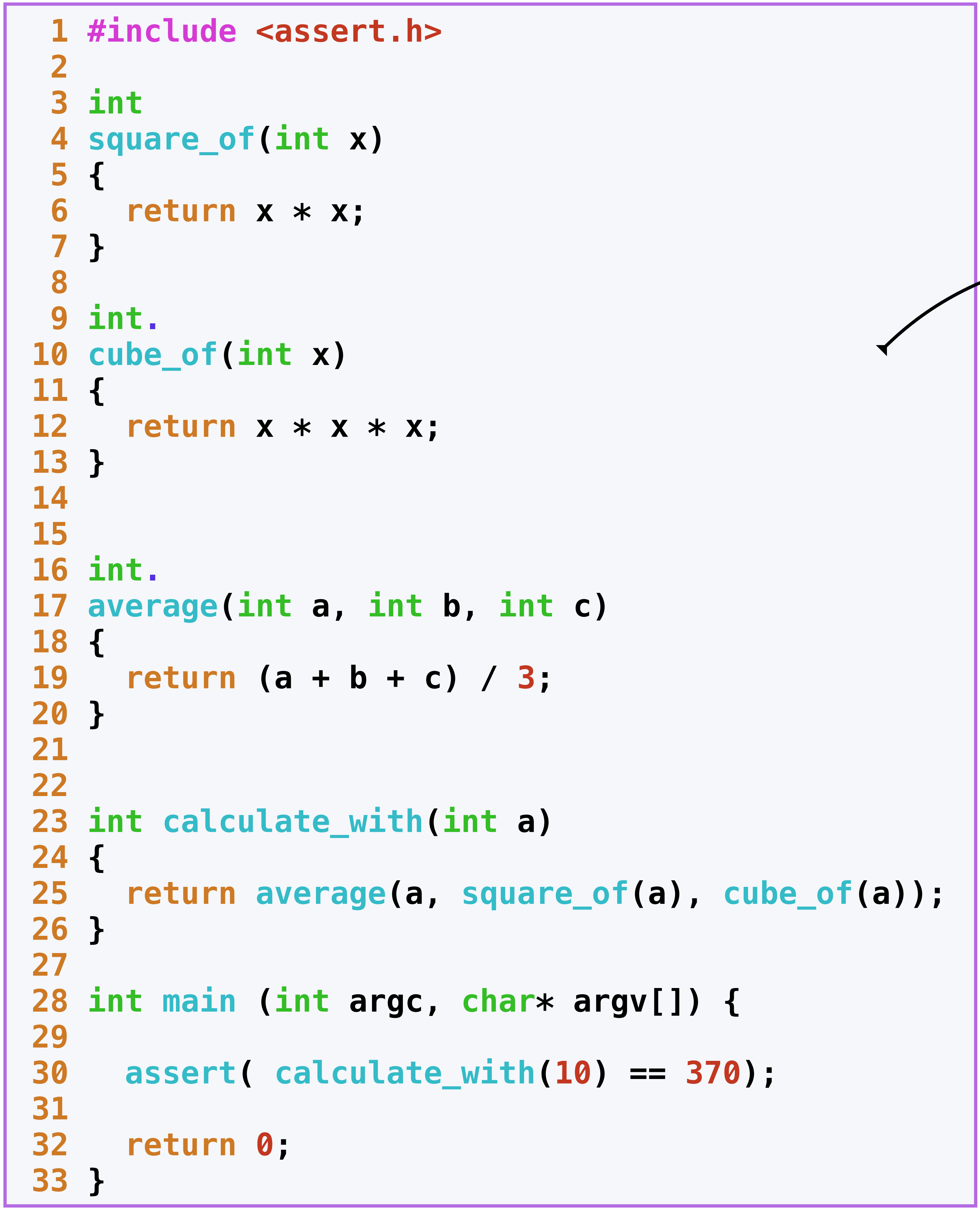
```
gcc -Wall -Werror -ansi -pedantic-errors prog1.c -o prog1.x
```

编译选项

编译命令举例

```
gcc -Wall -Werror -ansi -pedantic-errors  -g prog1.c -o prog1.x
```

调试选项，输出可执行文件中增加内嵌调试信息

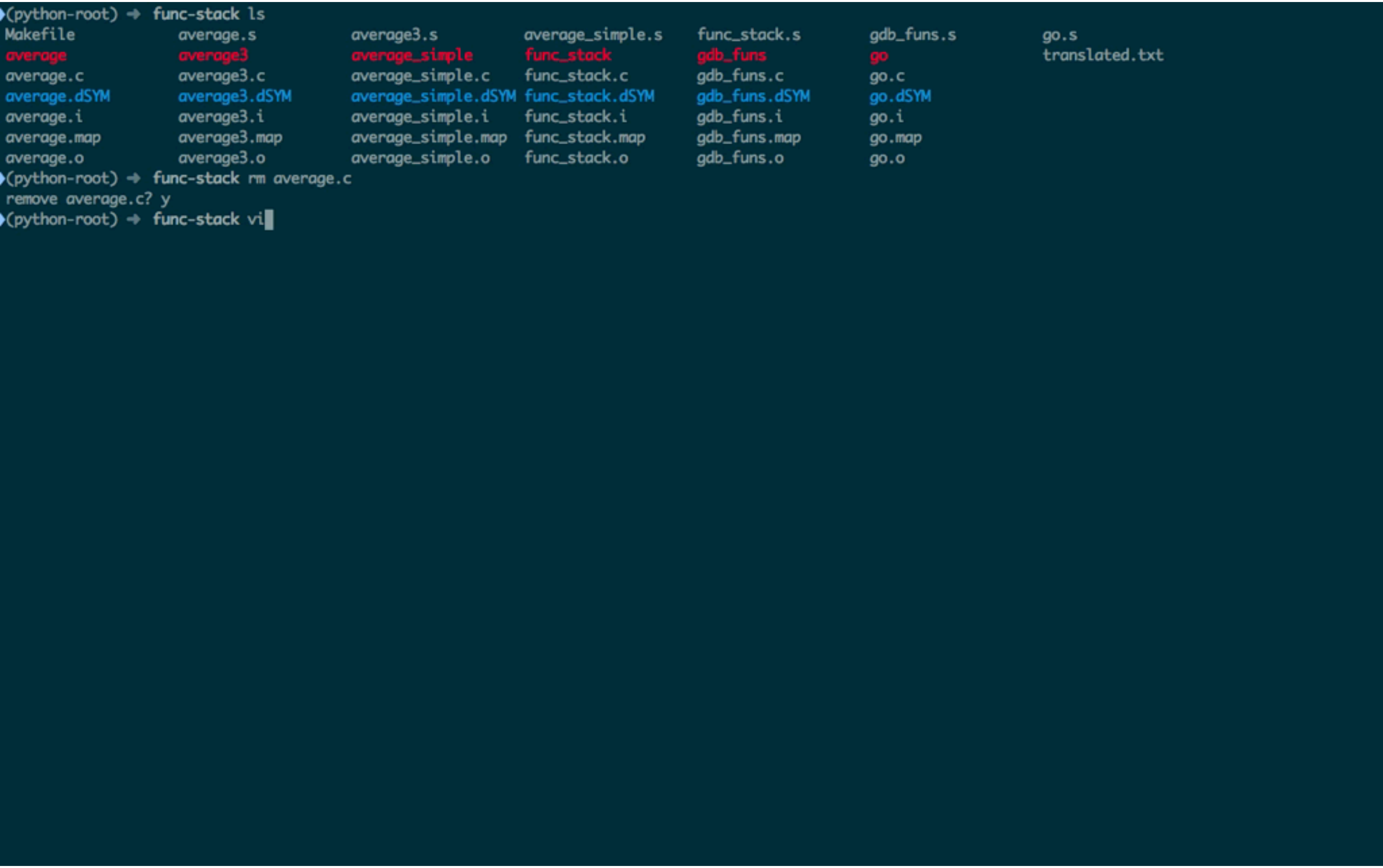


```c
#include <assert.h>

int
square_of(int x)
{
  return x * x;
}

int.
cube_of(int x)
{
  return x * x * x;
}


int.
average(int a, int b, int c)
{
  return (a + b + c) / 3;
}


int calculate_with(int a)
{
  return average(a, square_of(a), cube_of(a));
}

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

  assert( calculate_with(10) == 370);

  return 0;
}
```

average.c

请同学们把代码抄写在笔记本上…

代码录入演示



```
(python-root) ➜  func-stack ls
Makefile            average.s           average3.s          average_simple.s    func_stack.s        gdb_funs.s          go.s
average             average3            average_simple      func_stack          gdb_funs            go                  translated.txt
average.c           average3.c          average_simple.c    func_stack.c        gdb_funs.c          go.c
average.dSYM        average3.dSYM       average_simple.dSYM func_stack.dSYM     gdb_funs.dSYM       go.dSYM
average.i           average3.i          average_simple.i    func_stack.i        gdb_funs.i          go.i
average.map         average3.map        average_simple.map  func_stack.map      gdb_funs.map        go.map
average.o           average3.o          average_simple.o    func_stack.o        gdb_funs.o          go.o
(python-root) ➜  func-stack rm average.c
remove average.c? y
(python-root) ➜  func-stack vi
```

```
gcc -g average.c -o average
```

调试程序

```
gdb ./average
```

GDB快捷键

ctrl-x-a	TUI模式切换
ctrl-l	刷新屏幕
ctrl-p / ctrl-n	上一条/下一条 命令
ctrl-x-2	切分循环显示第二窗口
ctrl-x-1	代码窗口

(gdb) 命令

start
run

next
step

continue

break
where

list

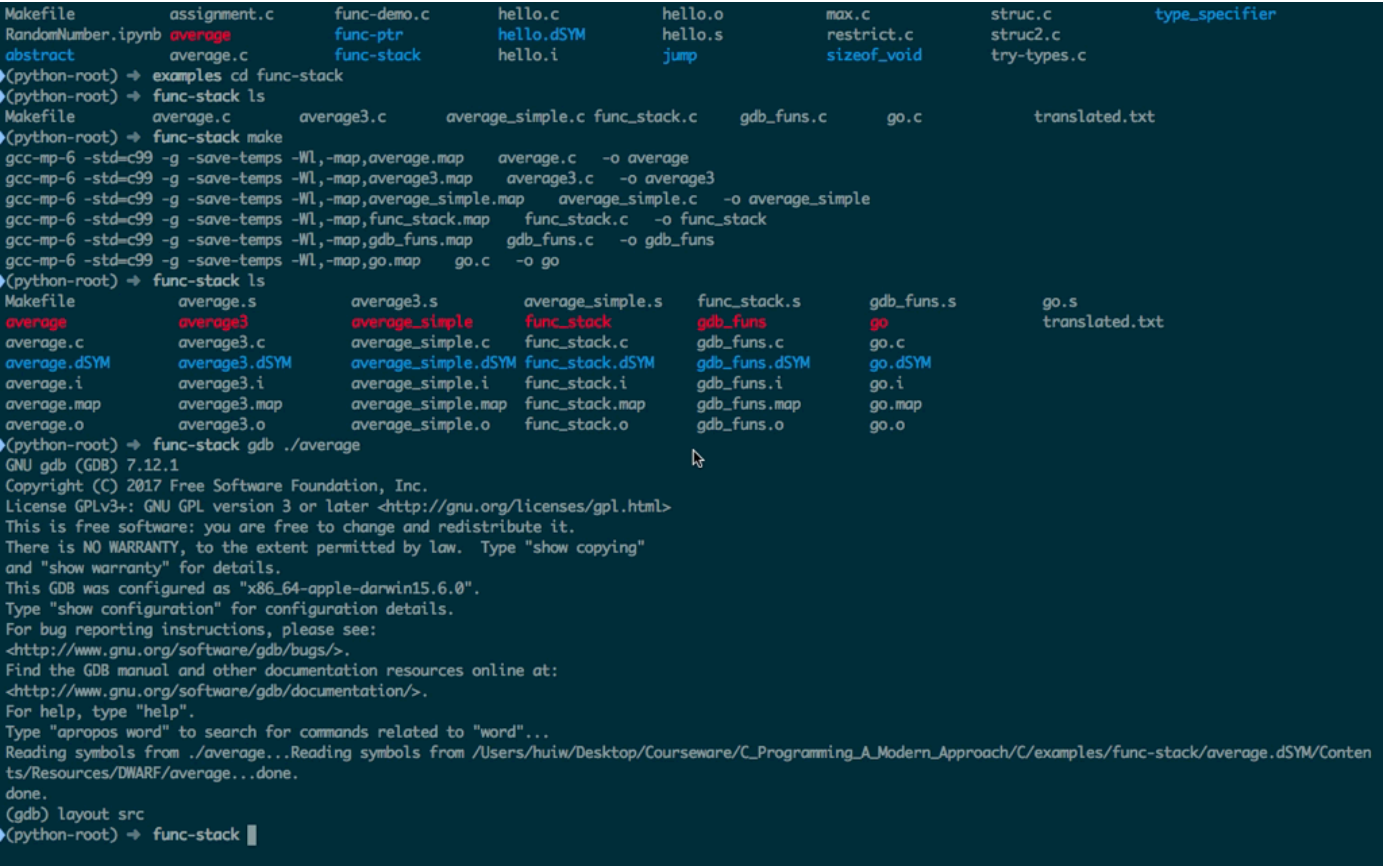
print

quit

help

Command	说明
run [args]	开始执行程序
start [args]	开始执行(并自动在main函数处 break)
break [line/function]	设定断点breakpoint
break [condition]	在条件成立时中断程序
continue	执行到被中断为止
next	单步执行(不会进入function内部)
step	单步执行(会进入function内部)
list [line/function]	列出程序代码
print [exp]	显示expression的值
print [var=val]	即时修改变量的值
where/backtrace	显示目前堆栈(stack)状态
help [subcommand]	说明

视频演示



```
Makefile          assignment.c      func-demo.c       hello.c           hello.o           max.c             struc.c           type_specifier
RandomNumber.ipynb average          func-ptr          hello.dSYM        hello.s           restrict.c        struc2.c
abstract          average.c         func-stack        hello.i           jump              sizeof_void       try-types.c
(python-root) ➜  examples cd func-stack
(python-root) ➜  func-stack ls
Makefile         average.c        average3.c        average_simple.c func_stack.c     gdb_funs.c       go.c             translated.txt
(python-root) ➜  func-stack make
gcc-mp-6 -std=c99 -g -save-temps -Wl,-map,average.map    average.c   -o average
gcc-mp-6 -std=c99 -g -save-temps -Wl,-map,average3.map    average3.c   -o average3
gcc-mp-6 -std=c99 -g -save-temps -Wl,-map,average_simple.map    average_simple.c   -o average_simple
gcc-mp-6 -std=c99 -g -save-temps -Wl,-map,func_stack.map    func_stack.c   -o func_stack
gcc-mp-6 -std=c99 -g -save-temps -Wl,-map,gdb_funs.map    gdb_funs.c   -o gdb_funs
gcc-mp-6 -std=c99 -g -save-temps -Wl,-map,go.map    go.c   -o go
(python-root) ➜  func-stack ls
Makefile            average.s           average3.s          average_simple.s    func_stack.s        gdb_funs.s          go.s
average             average3            average_simple      func_stack          gdb_funs            go                  translated.txt
average.c           average3.c          average_simple.c    func_stack.c        gdb_funs.c          go.c
average.dSYM        average3.dSYM       average_simple.dSYM func_stack.dSYM     gdb_funs.dSYM       go.dSYM
average.i           average3.i          average_simple.i    func_stack.i        gdb_funs.i          go.i
average.map         average3.map        average_simple.map  func_stack.map      gdb_funs.map        go.map
average.o           average3.o          average_simple.o    func_stack.o        gdb_funs.o          go.o
(python-root) ➜  func-stack gdb ./average
GNU gdb (GDB) 7.12.1
Copyright (C) 2017 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 "x86_64-apple-darwin15.6.0".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./average...Reading symbols from /Users/huiw/Desktop/Courseware/C_Programming_A_Modern_Approach/C/examples/func-stack/average.dSYM/Conten
ts/Resources/DWARF/average...done.
done.
(gdb) layout src
(python-root) ➜  func-stack 
```

WENZHENG COLLEGE OF SOOCHOW UNIVERSITY

2017.3.29

Soochow University