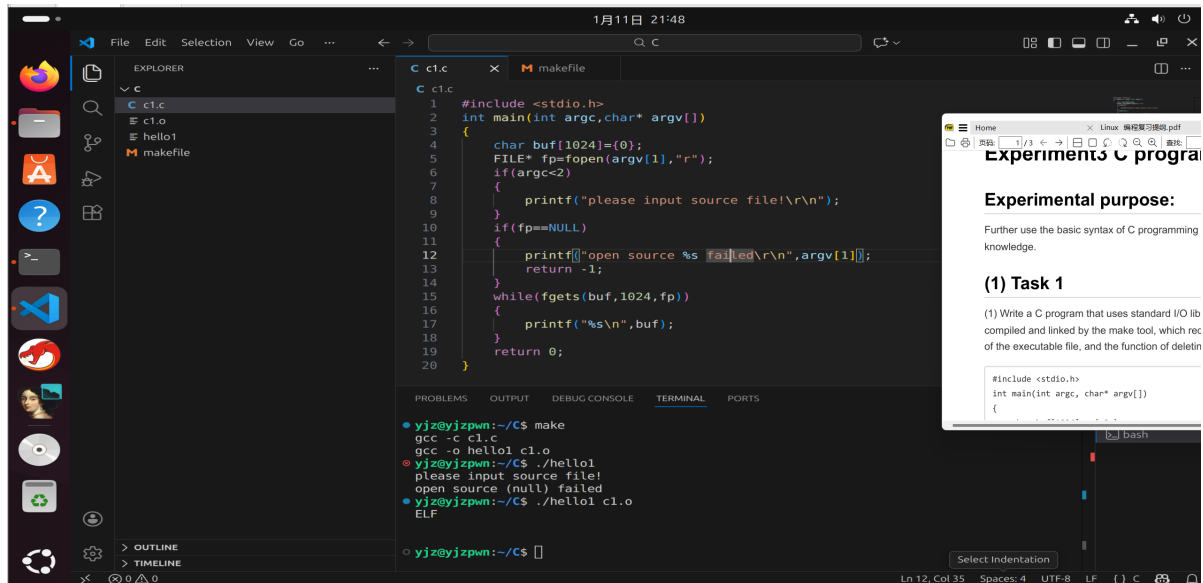


# Experiment\_3\_B22170220余佳卓

## (1)Task 1

源码：



The screenshot shows a code editor with two files: `c1.c` and `makefile`. The `c1.c` file contains a C program that prompts the user for a source file, attempts to open it, and prints its contents. The `makefile` file defines the compilation rules for `c1.c`. The terminal output shows the execution of the program, which fails to open the source file and prints an error message.

```
#include <stdio.h>
int main(int argc, char* argv[])
{
    char buf[1024]={0};
    FILE* fp=fopen(argv[1], "r");
    if(argc<2)
    {
        printf("please input source file!\n");
    }
    if(fp==NULL)
    {
        printf("open source %s failed\n", argv[1]);
        return -1;
    }
    while(fgets(buf, 1024, fp))
    {
        printf("%s\n", buf);
    }
    return 0;
}
```

```
hello: c1.o
gcc -o hello1 c1.o

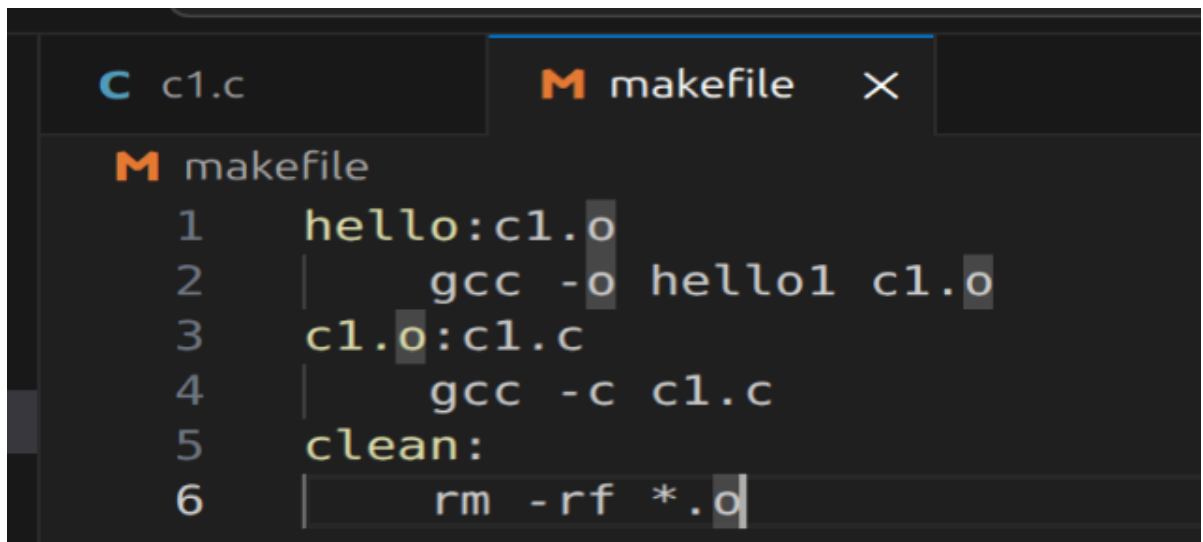
c1.o: c1.c
gcc -c c1.c

clean:
rm -rf *.o
```

Terminal output:

```
yjz@yjzpw: ~/C$ make
gcc -c c1.c
gcc -o hello1 c1.o
yjz@yjzpw: ~/C$ ./hello1
please input source file!
open source (null) failed
yjz@yjzpw: ~/C$ ./hello1 c1.o
ELF
```

makefile文件



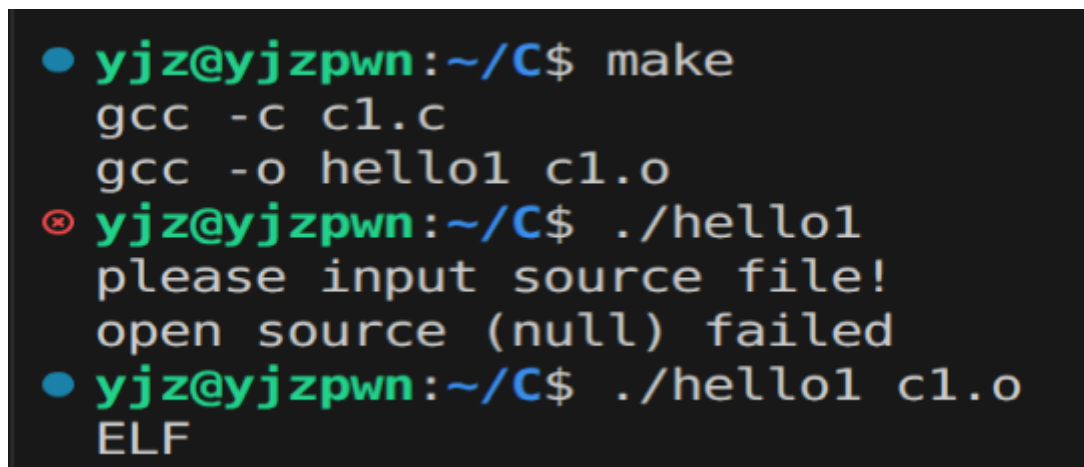
The screenshot shows the content of the `makefile` file, which defines the compilation rules for the C program.

```
hello: c1.o
gcc -o hello1 c1.o

c1.o: c1.c
gcc -c c1.c

clean:
rm -rf *.o
```

运行结果：



The screenshot shows the terminal output of the program execution, which matches the output shown in the first screenshot.

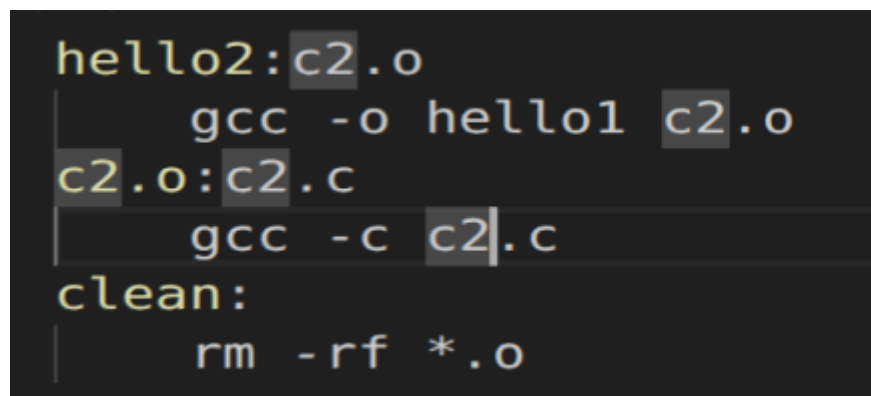
```
yjz@yjzpw: ~/C$ make
gcc -c c1.c
gcc -o hello1 c1.o
yjz@yjzpw: ~/C$ ./hello1
please input source file!
open source (null) failed
yjz@yjzpw: ~/C$ ./hello1 c1.o
ELF
```

## (2)Task 2



```
C c1.c  M makefile  C c2.c
C c2.c
1  #include <stdio.h>
2  #include <dirent.h>
3  #include <sys/types.h>
4
5  int main(int argc, char* argv[])
6  {
7      DIR* dirp;
8      struct dirent* direntp;
9      if((dirp=opendir(argv[1]))==NULL){
10         printf("error\r\n");
11     }
12     while((direntp=readdir(dirp))!=NULL)
13         printf("%s\r\n",direntp->d_name);
14     closedir(dirp);
15     exit(0);
16 }
```

makefile文件:



```
hello2:c2.o
|   gcc -o hello1 c2.o
c2.o:c2.c
|   gcc -c c2.c
clean:
|   rm -rf *.o
```

运行:

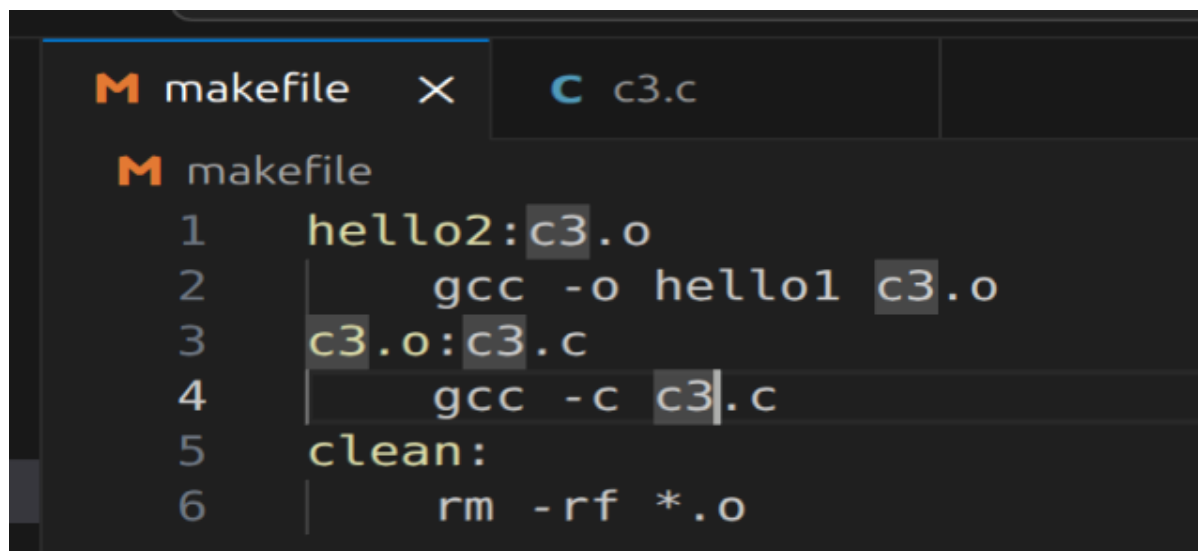
```
File Edit Selection View Go ...  
EXPLORER  
c  
  c1.c  
  c2.c  
  c2.o  
  hello1  
  makefile  
c2.c  
1 #include <stdio.h>  
2 #include <dirent.h>  
3 #include <sys/types.h>  
4  
5 int main(int argc, char* argv[])  
6 {  
7     DIR* dirp;  
8     struct dirent* direntp;  
9     if((dirp=opendir(argv[1]))==NULL){  
10         printf("error\n");  
11     }  
12 }  
yjs@yjszpw: ~/C  
yjs@yjszpw:~/C$ make  
13 gcc -c c2.c  
14 gcc -o hello1 c2.o  
yjs@yjszpw:~/C$ ./hello1  
Segmentation fault (core dumped)  
yjs@yjszpw:~/C$ ./hello1 .  
hello1  
c1.c  
c2.c  
makefile  
c2.o  
..  
yjs@yjszpw:~/C$
```

### (3)Task 3

源码

```
makefile c3.c  
c3.c  
1 #include <stdio.h>  
2 #include <stdlib.h>  
3 #include <unistd.h>  
4  
5 int main(){  
6     char buf[1024]={0};  
7     char buf2[1024]={0};  
8     getcwd(buf,1024);  
9     printf("%s\n",buf);  
10    if(chdir("/home")<0){  
11        printf("error\n");  
12    }  
13    else  
14    {  
15        printf("success\n");  
16    }  
17    getcwd(buf2,1024);  
18    printf("%s\n",buf2);  
19    return 0;  
20 }
```

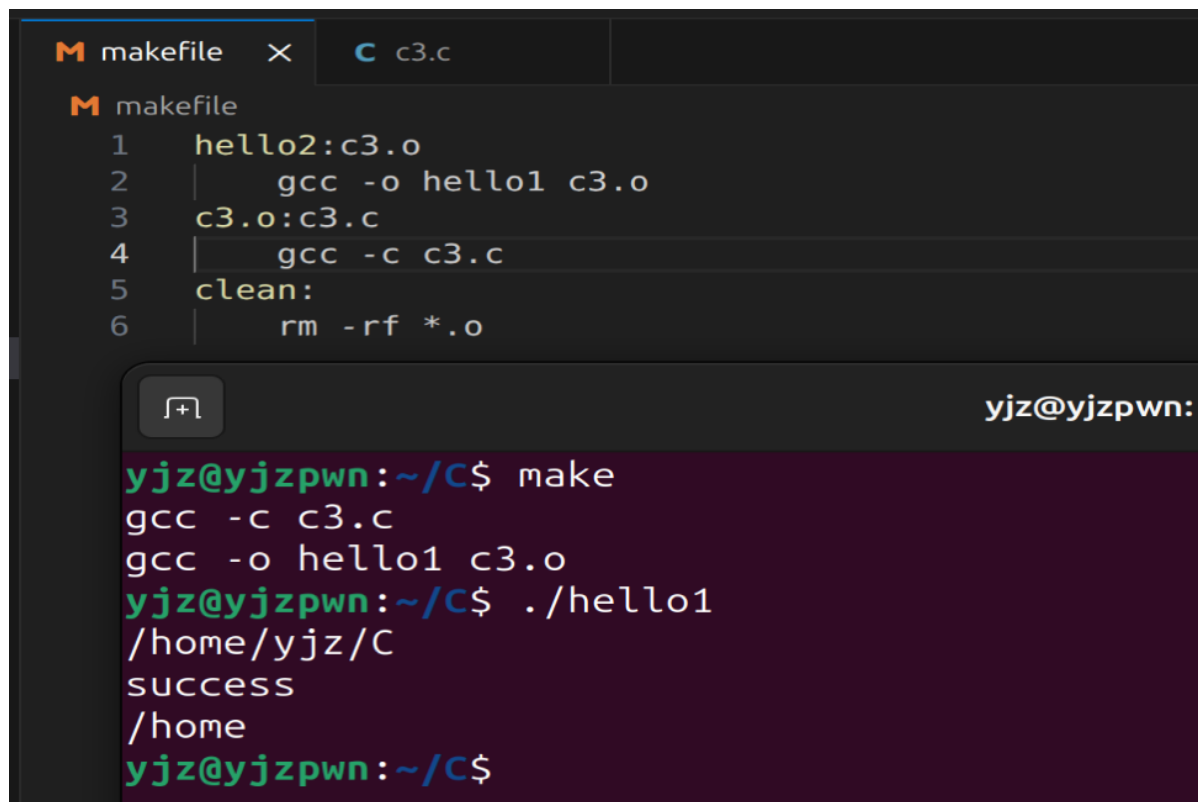
makefile文件



The screenshot shows a code editor with two tabs: 'makefile' and 'c3.c'. The 'makefile' tab is active, displaying the following content:

```
M makefile
1  hello2:c3.o
2      gcc -o hello1 c3.o
3  c3.o:c3.c
4      gcc -c c3.c
5  clean:
6      rm -rf *.o
```

运行



The screenshot shows a terminal window with the prompt 'yz@yzpwn:'. The terminal output is as follows:

```
yz@yzpwn:~/C$ make
gcc -c c3.c
gcc -o hello1 c3.o
yz@yzpwn:~/C$ ./hello1
/home/yz/C
success
/home
yz@yzpwn:~/C$
```