

Describe the difference between `strerror` and `perror`.

`Perror(const char *s)`

用來將上一個函數發生錯誤的原因輸出到標準錯誤 `stderr` 裡面。

假如我們 `include <errno.h>` 出錯的函式也會把它出錯的原因代號放進 `errno` 裡面。

參數 `s` 所指的字符串會先被打印出，冒號後面再印出錯誤原因字符串。

`Strerror(int errno)`

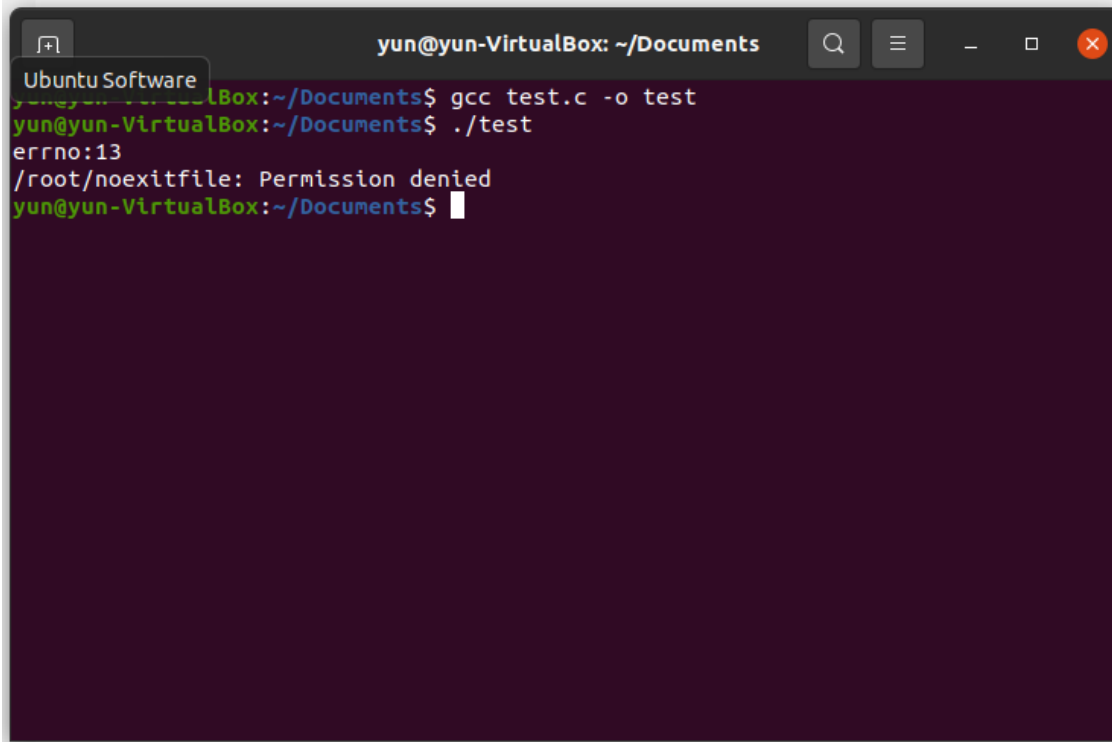
使用他的時候要 `include <errno.h>`，因為裡面有設定一個變數叫做 `errno`，假如程式有錯誤，系統就會把這個錯誤代號存到 `errno` 裡面，當我們呼叫 `strerror(errno)` 的時候，就可以利用印出字串的方式把這個 `errno` 代表的意義給印出來，我們就能知道這個程式出了什麼錯。

差異：

`Perror` 函式會直接打印出問題的字符串和錯誤原因。

`strerror` 函式要自己用 `%s` 印出錯誤的原因。

```
1 #include <stdio.h>
2 #include <errno.h>
3
4 int main()
5 {
6     FILE *fp;
7     fp= fopen("/root/noexitfile","r");
8     printf("errno:%d\n",errno);
9     if(NULL==fp)
10    {
11        perror("/root/noexitfile");
12    }
13
14    return 0;
15 }
```



The image shows a terminal window titled "yun@yun-VirtualBox: ~/Documents". The terminal output is as follows:

```
yun@yun-VirtualBox:~/Documents$ gcc test.c -o test
yun@yun-VirtualBox:~/Documents$ ./test
errno:13
/root/noexitfile: Permission denied
yun@yun-VirtualBox:~/Documents$
```

The terminal window has a dark background and a light-colored text. The prompt "yun@yun-VirtualBox:~/Documents\$" is shown in green. The output "errno:13" and "/root/noexitfile: Permission denied" are shown in white. The prompt "yun@yun-VirtualBox:~/Documents\$" is shown in green.