作業五：setuid

學習目標：

知道 Linux 中最基本的權限管理機制 - setuid

題目：

計算 GNU/Linux 中所有「正規檔案」的大小

寫一支程式，程式名稱為 preBirthday，無論誰執行這個檔案，都可以知道任何目錄下，所有「正規檔案」的「邏輯大小」的總和。例如：執行『preBirthday/』，會統計『根目錄』下所有的檔案。此外 preBirthday 會列出該目錄下有哪些檔案種類。

底下是關於：『正規檔案』的解釋

```
0 -r--------   1 root root 0 Apr  9 13:23 syscall
0 dr-xr-xr-x   3 root root 0 Apr  9 13:23 task
0 -r--r--r--   1 root root 0 Apr  9 13:23 timers
0 -rw-rw-rw-   1 root root 0 Apr  9 13:23 timerslack_ns
0 -rw-r--r--   1 root root 0 Apr  9 13:23 uid_map
0 -r--r--r--   1 root root 0 Apr  9 13:23 wchan
shiwulo@vm:/proc/43$
12 -rw------- 1 shiwulo shiwulo 12288 Mar 31 20:48 .swp
 0 drwxr-xr-x 1 shiwulo shiwulo   242 Mar 19 14:59 system-
programming
 0 drwxr-xr-x 1 shiwulo shiwulo     0 Feb 17 15:42 Templates
 0 drwxr-xr-x 1 shiwulo shiwulo     0 Feb 17 15:42 Videos
12 -rw------- 1 shiwulo shiwulo 11527 Apr  9 13:16 .viminfo
 4 -r-------- 1 shiwulo shiwulo  1679 Mar 24 14:42 vmkey.pem
```

```
 0 drwxr-xr-x 1 shiwulo shiwulo    80 Mar 19 15:01 .vscode
shiwulo@vm:~$
0 brw-rw----   1 root    disk      7,   9 Apr  9 13:19 loop9
0 crw-rw----   1 root    disk     10, 237 Apr  9 13:19 loop-control
shiwulo@vm:/dev$
```

在上面的例子中只有黃色標註起來的要統計大小，也就是檔案類型是「-」的要統計，其他的都不需要統計。另外再特別注意一下，preBirthday 列出的不只是 shiwulo 的檔案，也會進入「/」的所有目錄及子目錄，並計算大小及統計種類。

假設根目錄的檔案如上所示，那麼 preBirthday 的輸出是：

授課老師（羅習五）的生日是：1990/04/10

總共大小：25736 byte

檔案種類：-dbc

（別懷疑請輸出中文，不要忘記第一行字串，即『授課老師（羅習五）的生日是：1990/04/10』，「：」為全形或半形都可，大小與種類間換行）

提示：

1. 一定要用「setuid」

報告：

1. 報告上面寫上姓名（可隱匿一個字）和學號
2. 請在報告上說明你的檔案系統中『preBirthday /』時所列出的所有檔案類型的意義。例如：

   甲、「-」一般的檔案

   乙、「d」目錄檔案（在 UNIX 中目錄是一個特別的檔案，裡面記載這個目錄下包含了哪些檔案）（俺知道我上面這段話讀起來怪怪的，但就是這樣😃 😄 😅）

程式碼評分重點：

1. 確實能夠使用 setuid， 將 task 的權限變成「super user」
2. 能辨別「normal file」和其他的檔案
3. 能讀取檔案的「邏輯大小」
4. 列出該目錄共有多少檔案

能做到上面四點就給全部的分數

繳交：

1. 程式碼和 makefile，助教執行『sudo make』指令後，必須自動產生 preBirthday。
2. 撰寫報告，格式並須為 pdf。測試報告前請附上姓名（可隱匿一個字）及學號
3. 請將所有檔案壓縮成.tar.bz2。繳交到 ecourse2 上
4. 不能遲交
5. 再次提醒，助教會將所有人的作業於 dropbox 上公開
6. 繳交期限：2021/4/13 早上八點
7. 如果真的不會寫，記得去請教朋友。在你的報告上寫你請教了誰即可。

程式碼框架：

```
/**
DT_BLK      This is a block device. 「b」
DT_CHR      This is a character device. 「c」
DT_DIR      This is a directory. 「d」
DT_FIFO     This is a named pipe (FIFO). 「f」
DT_LNK      This is a symbolic link. 「l」
DT_REG      This is a regular file. 「-」
DT_SOCK     This is a UNIX domain socket. 「s」
DT_UNKNOWN  The file type could not be determined. 「U」
*/


#define _DEFAULT_SOURCE
#include <dirent.h>
#include <stdlib.h>
```

```c
#include <stdio.h>
#include <sys/types.h>
#include <stddef.h>
#include <sys/stat.h>
#include <unistd.h>
#include <dirent.h>
#include <string.h>
#include <assert.h>
//底下這個.h 定義了最長的 path name 長度
#include <linux/limits.h>

//檔案型別應該不會超過 100 個
int filetype[100];
char fileSymbol[100];

//準備統計目錄中到底有多少種檔案型別，如果有該型別在 filetype[X]上設定為 1，該型別的代表字放在 fileSymbol
void initFileTypoe() {
	for (int i=0; i< 99; i++) {
		filetype[i] = -1;
	}
	/*同學自己補下去*/
	fileSymbol[DT_BLK]='b';
	fileSymbol[DT_CHR]='c';
	fileSymbol[DT_DIR]='c';
}

//回傳某個檔案的大小
int readSize(char* pathname) {
	struct stat buf;
	//On success, zero is returned.  On error, -1 is returned, and errno is set appropriately.
	//https://blog.xuite.net/chowler/mainblog/5194764-assert%28%29+%E7%94%A8%E6%B3%95
	//assert()裡面填寫『我認為應該如此』，如果不是這樣的話，C 函數庫會吐出錯誤訊息給 programmer
	//在這裡用 lstat 和 stat 都可以，因為 pathname 傳進來的只會是 normal file，不會是「捷徑」（softlink）
	printf("readSize:%s\n", pathname);
	assert(stat(pathname, &buf)==0);
```

```
	return buf.st_size;

}

//使用遞迴計算某個目錄中的所有正規檔案的大小，並統計到底有多少種檔案型別
//如果沒有權限打開該打檔案怎麼辦？
// man access
/*          access() checks whether the calling process can access the file path-
           name.    If pathname is a symbolic link, it is dereferenced.

           The mode specifies the accessibility check(s) to be performed, and is
           either    the value F_OK, or a mask consisting of the bitwise OR of one
           or more of R_OK, W_OK, and X_OK.    F_OK tests for the existence of the
           file.     R_OK,    W_OK, and X_OK test whether the file exists and grants
           read, write, and execute permissions, respectively.
*/
long myCountDir(char* path) {
	long size = 0;
	//打開該目錄
	DIR* dirp = opendir(path);
	//讀取該目錄的第一個「物件」
	struct dirent* ent = readdir(dirp);
	while (ent != NULL) {
		//『這個目錄』及『上一層目錄』跳過不處理
		if (strcmp(ent->d_name, "." )==0 || strcmp(ent->d_name, ".." )==0) {
			ent = readdir(dirp);
			continue;
		}
		//設定有這種檔案型別
		filetype[ent->d_type] = 1;
		//製造『路徑/名』
		//如果使用者的輸入是「/」怎麼辦？，例如：「//home」會發生錯誤嗎？
		char pathname[PATH_MAX]="";
		strcat(pathname, path);
		strcat(pathname, "/");
		strcat(pathname, ent->d_name);
		printf("%s", pathname);
		//如果是目錄
```

```
            if (ent->d_type == DT_REG) {
                //遞迴呼叫
                size += readSize(pathname);
            } else if (ent->d_type == DT_DIR) {
                printf("myCountDir:%s\n", pathname);
                size += myCountDir(pathname);
            }
            ent = readdir(dirp);
        }
        closedir(dirp);
        return size;
}

int main(int argc, char** argv) {
        initFileTypoe();
        myCountDir(argv[1]);
}
```