‘test_case1.c’

```c
#include "types.h"
#include "stat.h"
#include "user.h"

int
main(void)
{
 countTraps();
 exit();
}
```

This is a trivial test case. It counts the number of times the user process has been trapped in the OS, the types of traps it got into and the number of occurrences. Also, it could the total number of syscalls, the types of syscalls in the user process, and the number of occurrences of those syscalls.

Result: (see instruction on how to run testing)

```
$ test_case1

*------------------------------*
| Total number of traps = 3    |
|------------------------------|
| Trap Name     | Occurrences  |
|------------------------------|
| [T_SYSCALL]   | 3            |
*------------------------------*

*--------------------------------------*
| Total number of system calls = 3     |
|--------------------------------------|
| Syscall Name          | Occurrences  |
|--------------------------------------|
| [SYS_exec]            | 1            |
| [SYS_sbrk]            | 1            |
| [SYS_countTraps]      | 1            |
*--------------------------------------*

*--------------------------------------*
| Total number of all calls = 6        |
*--------------------------------------*
```

‘test_case2.c’

This is a more complex test case. It tests commands such as write(), mkdir(), and read() and counts the number of times the user process has been trapped in the OS, the types of traps it got into and the number of occurrences for each command. Also, it could the total number of

syscalls, the types of syscalls in the user process, and the number of occurrences of those syscalls.

```c
#include "types.h"
#include "stat.h"
#include "user.h"
#include "fcntl.h"

void trapTester(void) {

  /**
   * Print First Counts*
  */

  // pretty print all system call count
  printf(1, "\n\t**********************\n");
  printf(1, "\t* Print First Counts *\n");
  printf(1, "\t**********************\n");
  countTraps();

  /**
   * Test System Calls*
  */

  printf(1, "\n\t**********************\n");
  printf(1, "\t* Sample System Calls *\n");
  printf(1, "\t**********************\n");

  // mkdir test
  printf(1, "\n Calling mkdir()");
  mkdir("Bye-bye");
  printf(1, "\n - Directory 'Bye-bye' has been created\n");

  // write test
  int sz, fd;
  char *c = (char *) malloc(sizeof("Bye-bye"));
  printf(1, "\n Calling write()");

  fd = open("Bye-bye.txt", O_CREATE | O_WRONLY);
  write(fd, "Bye-bye, Miss American Pie", strlen("Bye-bye"));
  printf(1, "\n - File 'Bye-bye.txt' has been created");
  printf(1, "\n - 'Bye-bye, Miss American Pie' has been written to the file");
  printf(1, "\n - %d bytes were written\n", sizeof(strlen("Bye-bye, Miss American Pie")));
  close(fd);

  // pretty print all system call count
  printf(1, "\n\t****************************************\n");
  printf(1, "\t*            Call All Traps After      *\n");
```

```c
  // pretty print all system call count
  printf(1, "\n\t****************************************\n");
  printf(1, "\t*            Call All Traps After      *\n");
  printf(1, "\t* [SYS_mkdir] and [SYS_write] syscalls *\n");
  printf(1, "\t****************************************\n");

  // get traps counts
  countTraps();

  fd = open("Bye-bye.txt", O_RDONLY);
  sz = read(fd, c, 10);
  printf(1, "\n '%s' was read from the file\n", c);
  printf(1, "\n '%d' the number of bytes were read\n", sz);
  close(fd);

  // pretty print all system call count
  printf(1, "\n\t************************\n");
  printf(1, "\t* Call All Traps After *\n");
  printf(1, "\t*  [SYS_READ] syscall  *\n");
  printf(1, "\t************************\n");

  // get traps counts
  countTraps();
  exit();
}

int main(int argc, char *argv[]) {
  trapTester();
}
```

Result: (see instruction on how to run testing) [each side respectively]

```
        ****************************************
        *            Call All Traps After      *
        * [SYS_mkdir] and [SYS_write] syscalls *
        ****************************************

*------------------------------*
| Total number of traps = 527  |
|------------------------------|
| Trap Name     | Occurrences  |
|------------------------------|
| [T_SYSCALL]   | 527          |
*------------------------------*

*--------------------------------------*
| Total number of system calls = 527   |
|--------------------------------------|
| Syscall Name          | Occurrences  |
|--------------------------------------|
| [SYS_exec]            | 1            |
| [SYS_sbrk]            | 2            |
| [SYS_open]            | 1            |
| [SYS_write]           | 519          |
| [SYS_mkdir]           | 1            |
| [SYS_close]           | 1            |
| [SYS_countTraps]      | 2            |
*--------------------------------------*

*--------------------------------------*
| Total number of all calls = 1054     |
*--------------------------------------*
```

```
        ************************
        * Call All Traps After *
        *  [SYS_READ] syscall  *
        ************************

*------------------------------*
| Total number of traps = 702  |
|------------------------------|
| Trap Name     | Occurrences  |
|------------------------------|
| [T_SYSCALL]   | 702          |
*------------------------------*

*--------------------------------------*
| Total number of system calls = 702   |
|--------------------------------------|
| Syscall Name          | Occurrences  |
|--------------------------------------|
| [SYS_read]            | 1            |
| [SYS_exec]            | 1            |
| [SYS_sbrk]            | 2            |
| [SYS_open]            | 2            |
| [SYS_write]           | 690          |
| [SYS_mkdir]           | 1            |
| [SYS_close]           | 2            |
| [SYS_countTraps]      | 3            |
*--------------------------------------*

*--------------------------------------*
| Total number of all calls = 1404     |
*--------------------------------------*
```