计算机系统作业第八章

8.12

`8个`

8.16

`counter = 2`

8.20

```c
#include <unistd.h>

int main(int argc, char **argv, char **envp) {
    execve("/bin/ls", argv, envp);
}
```

8.24

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <ctype.h>
#include <setjmp.h>
#include <signal.h>
#include <sys/time.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <sys/stat.h>
#include <fcntl.h>
```

```c
#include <sys/mman.h>
#include <errno.h>
#include <math.h>
#include <pthread.h>
#include <semaphore.h>
#include <sys/socket.h>
#include <netdb.h>
#include <netinet/in.h>
#include <arpa/inet.h>

#define N 2

void unix_error(char *msg) {
    fprintf(stderr, "%s: %s\n", msg, strerror(errno));
    exit(0);
}

pid_t Fork() {
    pid_t pid;
    if ((pid = fork()) < 0)
        unix_error("Fork error");
    return pid;
}

int main() {
    int status, i;
    pid_t pid;

    for (i = 0; i < N; ++i)
        if ((pid = Fork()) == 0) {
            int x[100];
            for (int j = 10; j <= 1000; ++j) //try to cause an error
                x[j] = j;
            exit(100+i);
        }
    while ((pid = waitpid(-1, &status, 0)) > 0) {
        if (WIFEXITED(status))
            printf("child %d terminated normally with exit status=%d\n", pid, WEXITSTATUS(status));
        else {
            char s[100]; //suppose the string is no longer than 100 letters.
            sprintf(s, "child %d terminated by signal %d", pid, WTERMSIG(status));
            psignal(WTERMSIG(status), s);
        }
    }

    if (errno != ECHILD)
```

```c
        unix_error("waitpid error");

    exit(0);
}

```