Lab4 Report

11911609 葛兆宁

- Q1
  代码中区分父进程与子进程主要靠fork()函数的返回值，父进程的返回值为子进程的pid，子进程的返回值为0，同时也可通过pid来区分（exec*()函数除外）

- Q2
  waitpid()函数原型为

```
    pid_t waitpid(pid_t pid, int *wstatus, int options);
```

- 其中 pid表示需要等待的子进程，其中pid<-1表示等待此pid所指的进程组里的任意子进程死亡，pid=-1表示等待任意子进程死亡，pid=0表示等待任意父进程所在的进程组里的子进程死亡，pid>0表示等待指定pid子进程死亡，wstatus表示子进程的死亡报告的存放地址，options里表示表示返回时的原因（或选项）

- Q3
  在父进程死在子进程前面，那么子进程的父进程会被过继到进他进程下（主要是init或它的其他祖先进程）进行管理。

- Q4
  code:

```
#include <stdio.h>
#include <unistd.h>
int main()
{
    if (fork())
    {
        printf("The parent process pid is %d\n", getpid());
        while (1)
            ;
    }
    else
    {
        printf("The child's pid process is %d\n", getpid());
    }
}
```

result:

```
11911609JohnnyGe@johnny-Ge-WXX9:~/OS/lab4$ g++ zombie.c
11911609JohnnyGe@johnny-Ge-WXX9:~/OS/lab4$ ./a.out
```

```
The parent process pid is 8283
The child's pid process is 8284
```

```
11911609JohnnyGe@johnny-Ge-WXX9:~/OS/lab4$ ps -al
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   997    1544    1542  3  80   0 - 265971 ep_pol tty2    00:03:53 Xorg
0 S   997    1565    1542  0  80   0 - 48474 poll_s tty2     00:00:00
gnome-session-b
0 S   997    4623    4591  0  80   0 -  3279 do_wai pts/0    00:00:00 man
0 S   997    4634    4623  0  80   0 -  2881 wait_w pts/0    00:00:00 pager
0 R   997    8283    8114 96  80   0 -   624 -      pts/1    00:00:11 a.out
1 Z   997    8284    8283  0  80   0 -     0 -      pts/1    00:00:00 a.out
<defunct>
4 R   997    8338    8319  0  80   0 -  3627 -      pts/3    00:00:00 ps
```

- Q5
  code:

```
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>
int main()
{
    printf("waiting for the child\n");
    if (fork())
    {
        int a = wait(NULL);
        printf("success!\n");
    }
    else
    {
        execl("/bin/ps", "ps", "-al", NULL);
        printf("task finished\n");
    }
}
```

result:

```
11911609JohnnyGe@johnny-Ge-WXX9:~/OS/lab4$ g++ Q5.c
11911609JohnnyGe@johnny-Ge-WXX9:~/OS/lab4$ ./a.out
waiting for the child
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   997    1544    1542  3  80   0 - 265534 ep_pol tty2    00:05:18 Xorg
0 S   997    1565    1542  0  80   0 - 48474 poll_s tty2     00:00:00
gnome-session-b
0 S   997    4623    4591  0  80   0 -  3279 do_wai pts/0    00:00:00 man
0 S   997    4634    4623  0  80   0 -  2881 wait_w pts/0    00:00:00 pager
```

```
0 S   997   10317    8114  0  80   0 -   624 do_wai pts/1    00:00:00 a.out
4 R   997   10318   10317  0  80   0 -  3627 -      pts/1    00:00:00 ps
success!
```

- Q7
  父进程从pipe中读信息，读完发信息给子进程（中断），子进程往pipe中写信息，写完后发信息给父进程（中断），如此往复，乐此不疲。