

1008进程控制（复习+一些补充）

测试一下系统最多能创建几个进程

```
3 #include <stdio.h>
4 #include <stdlib.h>
5 #include <unistd.h>
6
7 int main()
8 {
9     int max_cnt = 0;
10    while(1)
11    {
12        pid_t id = fork();
13        if(id < 0)
14        {
15            printf("fork error: %d\n", max_cnt);
16            break;
17        }
18        else if(id == 0)
19        {
20            while(1)
21            {
22                sleep(1); // 不要让子进程退出
23            }
24        }
25        else
26        {
27            // father
28        }
29        max_cnt++;
30    }
31    return 0;
32 }
```

Makefile test.c

- [yufc@VM-12-12-centos 1008]\$ vim Makefile
- [yufc@VM-12-12-centos 1008]\$ make
- gcc -o test test.c
- [yufc@VM-12-12-centos 1008]\$ ./test
- fork error: 7197
- bash: fork: retry: No child processes
- [yufc@VM-12-12-centos 1008]\$

PPID	PID	PGID	SID	TTY	TPGID	STAT	UID	TI
1	13359	8718	8225	pts/3	16448	S	1001	0:

- ^C[yufc@VM-12-12-centos 1008]\$ killall test
- ⊗ [yufc@VM-12-12-centos 1008]\$ while ;; do ps axj | head

全部杀掉 --- 不然系统会很卡

## 给minshell增加导入环境变量的功能

```
PUTENV(3) Linux Programmer's Manual PUTENV(3)

NAME
    putenv - change or add an environment variable

SYNOPSIS
    #include <stdlib.h>

    int putenv(char *string);

Feature Test Macro Requirements for glibc (see feature_test_macros(7)):

    putenv(): _SVID_SOURCE || _XOPEN_SOURCE

DESCRIPTION
    The putenv() function adds or changes the value of environment variables. The argument string is of the form name=value. If name does not already exist in the environment, then string is added to the environment. If name does exist, then the value of name in the environment is changed to value. The string pointed to by string becomes part of the environment, so altering the string changes the environment.
```

```
//4. TODO
if(strcmp(g_argv[0], "cd") == 0)
{
    //让父进程执行 -- 不要创建子进程
    //内置命令（内建命令） -- 本质就是shell中的一个函数调用
    //我们用一个系统调用 -- chdir
    if(g_argv[1] != NULL) chdir(g_argv[1]);
    continue;
}

//导入环境变量
if(strcmp(g_argv[0], "export") == 0 && g_argv[1] != NULL)
{
    putenv(g_argv[1]);
    continue;
}
```

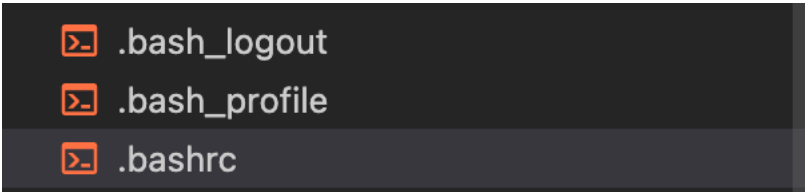
当然直接这样去操作是会有问题的

cmd\_line会被清空，子进程虽然获取到了环境变量，但是存在cmd\_line里面的字符串被清空了这个问题藏得很深 --- 具体要看看蛋哥的视频

不是说好程序替换会替换代码和数据吗？？？  
环境变量相关的数据，会被替换吗？？  
没有！！

这也体现了环境变量具有全局属性

一个问题：  
shell的环境变量从哪儿来呢？  
其实系统的配置文件（shell脚本）中都是有的

A dark-themed terminal window with three lines of text, each preceded by a red prompt icon. The text lists system configuration files for the bash shell.

```
> .bash_logout  
> .bash_profile  
> .bashrc
```

环境变量，都是写在配置文件当中的（shell脚本），shell启动的时候，通过读取配置文件获得的起始环境变量  
（了解即可）