同济大学计算机网络

实验报告

姓名：涂远鹏-1652262

一.正常程序的后台运行及前台切换

1.在 05 子目录下写 test5-1.c，主进程每分裂若干子进程打印一次”已分裂***子进程”，写配套的 makefile，make 后生成 test1-1 和 test1-2 两个可执行文件：

答：

编写的 test5-1.c 如下：

```
#include <stdio.h>
#include <sys/types.h>
#include <signal.h>
#include <string.h>
#include <sys/wait.h>

void func_waitpid(int signo) {
    pid_t pid;
    int stat;
    while( (pid = waitpid(-1, &stat, WNOHANG)) > 0 ) {
        printf( "child %d exit\n", pid );
    }
    return;
}

int sub()
{
char str[1024*10]={20};
for(;;)
    sleep(20);
}
int main(int argc,char *argv[])
{
signal(SIGCHLD, &func_waitpid);
    int i,j,num=0;
for(i=0;argv[1][i];i++)
num=num*10+argv[1][i]-'0';
pid_t pid1;
    pid1=fork();
    if(pid1==-1)
    return 0;
    if(pid1==0)
    {
    for(i=0;i<num;)
    {
    pid_t pid = fork();
    if (pid == 0)
    {
        sub();
        return 0;
    }
    if(pid==-1)
    {
        printf("共分裂%d个子进程\n",i+1 );
        break;
    }
    if(!(i%100))
        printf("已分裂%d个子进程\n",i+1);
        i++;
    }
    while(1)
    {
```

```
    }
    while(1)
    {
      sleep(5);
    }
    return 0;
}
if(pid1>0)
return 0;

}
```

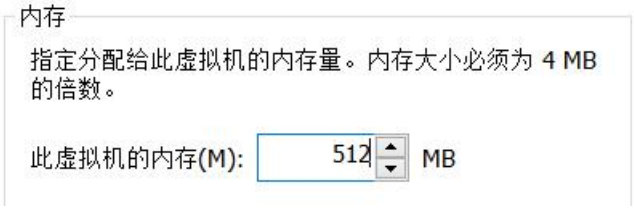
编写的 makefile 文件如下：

```
test:test5-1 test5-2 test5-3
$(test):
        cc -o
clean:
        rm test5-?
~
~
```

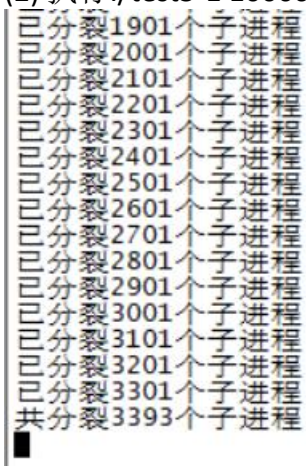
2.虚拟机的内存设置为 512MB，分裂数量达到多少时，分裂子进程会失败：

答：

(1)设置虚拟机内存：



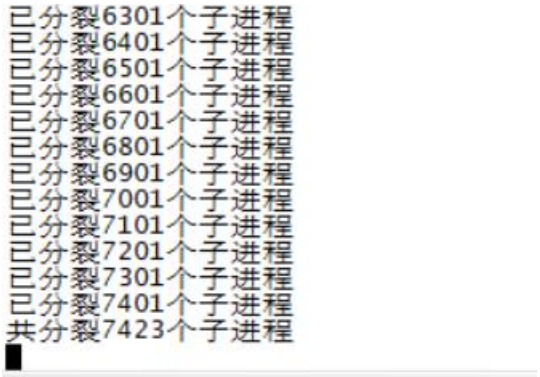
(2) 执行./test5-1 10000,最多为 3393 个子进程：



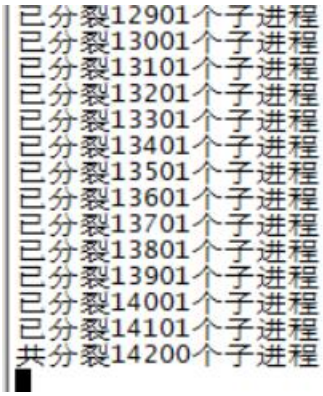
3.虚拟机的内存设置为 1024MB、2048MB 时，分裂最大数量又是多少？

答：

(1)设置为 1024MB，分裂测试结果如下所示,最多 7423：



(2)设置为 2048MB，分裂测试结果如下所示，最多 14200：



4.把 char str[1024]改成 char str[1024*10]，再次测试三种内存下的最大分裂数量？

答：

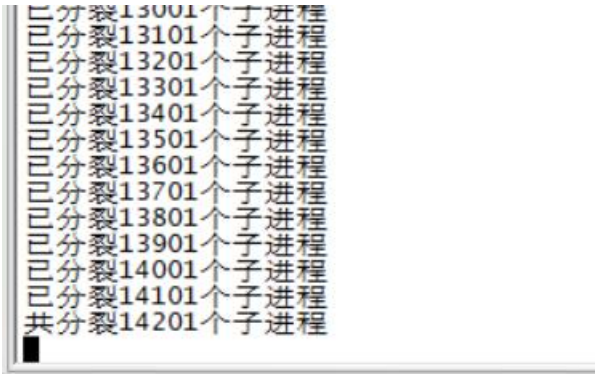
(1)512MB 情况下测试：

```
[root@RHEL74-SVR 05]# ./test5-1 10000
[root@RHEL74-SVR 05]# 已分裂1个子进程
已分裂101个子进程
已分裂201个子进程
已分裂301个子进程
已分裂401个子进程
已分裂501个子进程
已分裂601个子进程
已分裂701个子进程
已分裂801个子进程
已分裂901个子进程
已分裂1001个子进程
已分裂1101个子进程
已分裂1201个子进程
已分裂1301个子进程
已分裂1401个子进程
已分裂1501个子进程
已分裂1601个子进程
已分裂1701个子进程
已分裂1801个子进程
已分裂1901个子进程
已分裂2001个子进程
已分裂2101个子进程
已分裂2201个子进程
已分裂2301个子进程
已分裂2401个子进程
已分裂2501个子进程
已分裂2601个子进程
已分裂2701个子进程
已分裂2801个子进程
已分裂2901个子进程
已分裂3001个子进程
已分裂3101个子进程
已分裂3201个子进程
已分裂3301个子进程
共分裂3396个子进程
```

(2)1024MB 情况下测试：

```
已分裂4501个子进程
已分裂4601个子进程
已分裂4701个子进程
已分裂4801个子进程
已分裂4901个子进程
已分裂5001个子进程
已分裂5101个子进程
已分裂5201个子进程
已分裂5301个子进程
已分裂5401个子进程
已分裂5501个子进程
已分裂5601个子进程
已分裂5701个子进程
已分裂5801个子进程
已分裂5901个子进程
已分裂6001个子进程
已分裂6101个子进程
已分裂6201个子进程
已分裂6301个子进程
已分裂6401个子进程
共分裂6424个子进程
```

(3)2048MB 情况下测试：



5.写 test5-2.c，子进程给 str 赋值后，不要死循环，等待 20 秒后子进程退出，如何做到在小内存情况下分裂完成指定大数量的子进程？

答：

编写实现对应要求的 test5-2.c，内容如下：

```c
#include <stdio.h>
#include <sys/types.h>
#include <signal.h>
#include <string.h>
#include <sys/wait.h>
#include <unistd.h>
int numin=0,numout=0,max=0;
void func_waitpid(int signo) {
    pid_t pid;
    int stat;
    while( (pid = waitpid(-1, &stat, WNOHANG)) > 0 )
        numout++;
    return;
}
int sub()
{
char str[1024*10]={20};
    sleep(20);
}
int main(int argc,char *argv[])
{
    signal(SIGCHLD, &func_waitpid);
    int i,j,num=0;
    for(i=0;argv[1][i];i++)
    num=num*10+argv[1][i]-'0';
    pid_t pid1;
    pid1=fork();
    if(pid1==-1)
    return 0;
    if(pid1==0)
    {
      for(i=0;i<num;)
      {
        if(numin-numout<1500)
        {
          pid_t pid = fork();
          if (pid == 0)
          {
           if(i==num-1)
           printf("子进程最大进程号为%d",(int)getpid());
           sub();
           return 0;
          }
          else if(pid==-1)
          {
           printf("共分裂%d个子进程\n",i+1 );
           break;
          }
         else
         {
          numin++;
          if(!(i%1000))
```

```
            numin++;
            if(!(i%1000))
            {
              printf("已分裂%d个子进程\n",i+1);
              sleep(20);
            }
          i++;
        }
      }
      else
      sleep(20);
      }
      if(i==num)
      printf("共分裂%d个子进程\n",i);
     while(1)
     {
       sleep(5);
     }
     return 0;
   }
    if(pid1>0)
    return 0;
}
```

512MB 情况下测试分裂的进程数测试(上面同等条件下为 3396 个)：

```
已分裂81001个子进程
已分裂82001个子进程
已分裂83001个子进程
已分裂84001个子进程
已分裂85001个子进程
已分裂86001个子进程
已分裂87001个子进程
已分裂88001个子进程
已分裂89001个子进程
已分裂90001个子进程
已分裂91001个子进程
已分裂92001个子进程
已分裂93001个子进程
已分裂94001个子进程
已分裂95001个子进程
已分裂96001个子进程
已分裂97001个子进程
已分裂98001个子进程
已分裂99001个子进程
共分裂100000个子进程
```

6.在 test5-2.c 中加适当语句，看分裂的子进程的最大进程号是多少？

答：

```
已分裂87001个子进程
已分裂88001个子进程
已分裂89001个子进程
已分裂90001个子进程
已分裂91001个子进程
已分裂92001个子进程
已分裂93001个子进程
已分裂94001个子进程
已分裂95001个子进程
已分裂96001个子进程
已分裂97001个子进程
已分裂98001个子进程
已分裂99001个子进程
共分裂100000个子进程
子进程最大进程号为102313
```

7.写 test5-3.c，由守护进程负责回收每个子进程退出，设置两个全局变量作为计数器，一个记录分裂成功的数量，一个记录回收成功的数量，要求分裂完成之后，

且所有子进程都退出后，两个计数器的值相同：
答：
编写的 test5-3.c 程序如下：

```c
#include <stdio.h>
#include <sys/types.h>
#include <signal.h>
#include <string.h>
#include <sys/wait.h>
#include <unistd.h>
int numin=0,numout=0;
void func_waitpid(int signo) {
    pid_t pid;
    int stat;
    while( (pid = waitpid(-1, &stat, WNOHANG)) > 0 )
        numout++;
    return;
}
int sub()
{
    char str[1024*10]={20};
    sleep(20);
}
int main(int argc,char *argv[])
{
    signal(SIGCHLD, &func_waitpid);
    int i,j,num=0;
    for(i=0;argv[1][i];i++)
    num=num*10+argv[1][i]-'0';
    pid_t pid1;
    pid1=fork();
    if(pid1==-1)
    return 0;
    if(pid1==0)
    {

    for(i=0;i<num;)
    {
      if(numin-numout<1500)
      {
         pid_t pid = fork();
         if (pid == 0)
         {
           sub();
           if(i==num-1)
           printf("子进程最大进程号为%d\n",(int)getpid());
           return 0;
         }
         else if(pid==-1)
         {
           printf("共分裂%d个子进程\n",i+1 );
           break;
         }
         else
         {
           numin++;
```

```
            if(!(i%1000))
            {
              printf("已分裂%d个子进程\n",i+1);
              sleep(20);
            }
            if(!(numin%1000))
            printf("分裂%d   回收%d\n",numin,numout);
            i++;
          }
        }
      else
      sleep(20);
      }
    if(i==num)
      printf("共分裂%d个子进程\n",i);
    while(1)
    {
      sleep(5);
      printf("分裂个数%d\n",numin);
      printf("回收个数%d\n",numout);
    }
    return 0;
   }
   if(pid1>0)
   return 0;

}
-- INSERT --
```

测试分裂 10000 个子进程，测试结果如下，达到预期效果：

```
[root@RHEL74-SVR 05]# ./test5-3 10000
[root@RHEL74-SVR 05]# 已分裂1个子进程
分裂1000   回收1
已分裂1001个子进程
分裂2000   回收547
已分裂2001个子进程
分裂3000   回收1566
已分裂3001个子进程
分裂4000   回收2501
已分裂4001个子进程
分裂5000   回收3515
已分裂5001个子进程
分裂6000   回收4797
已分裂6001个子进程
分裂7000   回收5501
已分裂7001个子进程
分裂8000   回收6873
已分裂8001个子进程
分裂9000   回收7867
已分裂9001个子进程
分裂10000   回收8501
共分裂10000个子进程
分裂个数10000
```

```
回收个数8901
分裂个数10000
回收个数8918
分裂个数10000
回收个数8921
分裂个数10000
回收个数8958
分裂个数10000
回收个数9001
分裂个数10000
回收个数9009
分裂个数10000
回收个数9444
子进程最大进程号为12251
分裂个数10000
回收个数10000
分裂个数10000
回收个数10000
分裂个数10000
回收个数10000
分裂个数10000
回收个数10000
分裂个数10000
回收个数10000
```