

**本科实验报告**

课 程 名 称： 操作系统原理实验 课 程 编 号： 08060157 学 生 姓 名： 阮炜霖 学 号： 2020101603 学 院： 信息科学技术学院

系： 计算机科学系

专 业： 网络工程

指 导 教 师： 郝振明

教 师 单 位： 计算机科学系 开 课 时 间：2021 ~ 2022 学年度第 二 学期

## 暨南大学教务处

2022年 6 月 15 日

操作系统原理实验 **课程实验项目目录**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 实验项目 编号 | 实验项目名称 | 实验性质 | 成绩 | 指导教师 |
| 1 | 0806015701 | 实验环境的使用 | 验证 |  | 郝振明 |
| 2 | 0806015702 | 进程的控制 | 设计 |  | 郝振明 |
| 3 | 0806015703 | 进程间的通信 | 设计 |  | 郝振明 |
| 4 | 0806015704 | 生产者消费者问题 | 综合 |  | 郝振明 |
| 5 | 0806015705 | 简单文件系统实现 | 综合 |  | 郝振明 |
| 6 | 0806015706 | 非必须实验。若选择做此项目，则项目名称自定。 |  |  | 郝振明 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

\*此表由学生按顺序填写。

# 暨南大学本科实验报告专用纸

课程名称 操作系统原理实验 成绩评定

实验项目名称 进程的控制 \_\_\_ 指导教师 郝振明

实验项目编号 0806015702 实验项目类型 设计型

学生姓名 阮炜霖 学号 2020101603

学院信息科学技术学院 系 计算机科学系 专业 网络工程

实验时间 2022 年 3 月 17 日 上 午 3、4节 实验地点 N116

**一、 实验目的**

1.实验目的

掌握创建子进程和加载执行新程序的方法，理解创建子进程和加载执行程序的不同。

调试跟踪fork和execve系统调用函数的执行过程。

2.实验要求

任选高级程序设计语言编写源程序，在Linux操作系统下调试通过，测试正确。

**二、 实验内容**

创建子进程，加载执行新程序，调试跟踪 fork 和 execve 系统调用函数的执行过程。

## 三、 实验步骤

（一）使用准备

安装Clion用于集成开发环境，新建一个C/C++程序项目。

（二）编写程序

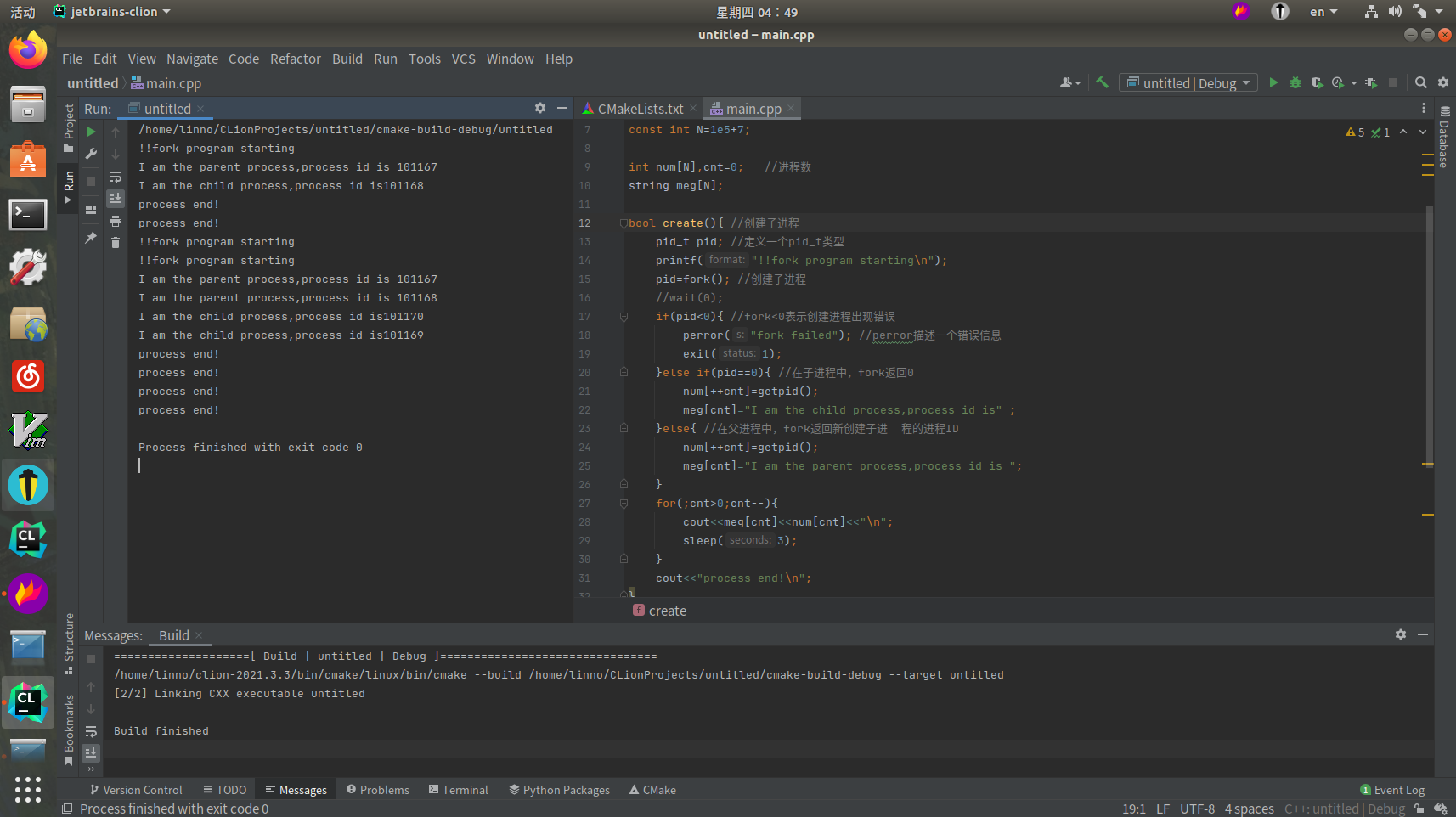
1.在CLion中， 编写以下代码。

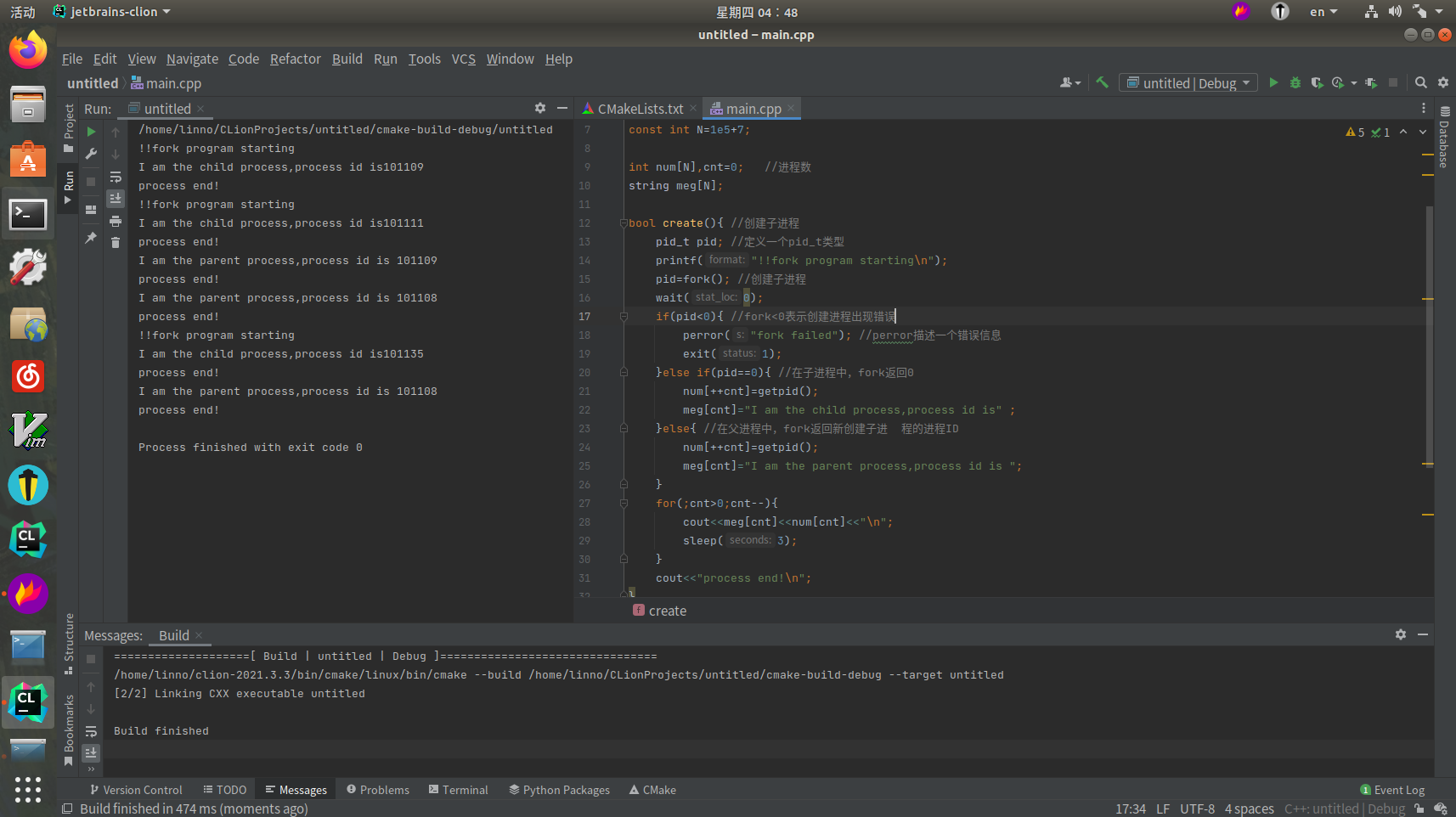
1. #include <iostream>
2. #include <sys/types.h>
3. #include<sys/wait.h>
4. #include <stdio.h>
5. #include <unistd.h>
6. **using** **namespace** std;
7. **const** **int** N=1e5+7;
9. **int** num[N],cnt=0;   //进程数
10. string meg[N];
12. **bool** create(){ //创建子进程
13. pid\_t pid; //定义一个pid\_t类型
14. printf("!!fork program starting\n");
15. pid=fork(); //创建子进程
16. //wait(0);
17. **if**(pid<0){ //fork<0表示创建进程出现错误
18. perror("fork failed"); //perror描述一个错误信息
19. exit(1);
20. }**else** **if**(pid==0){ //在子进程中，fork返回0
21. num[++cnt]=getpid();
22. meg[cnt]="I am the child process,process id is" ;
23. }**else**{ //在父进程中，fork返回新创建子进  程的进程ID
24. num[++cnt]=getpid();
25. meg[cnt]="I am the parent process,process id is ";
26. }
27. **for**(;cnt>0;cnt--){
28. cout<<meg[cnt]<<num[cnt]<<"\n";
29. sleep(3);
30. }
31. cout<<"process end!\n";
32. }
34. **signed** main(){
35. create()&&create()||create();
36. exit(0);
37. **return** 0;
38. }

2. 运行并调试程序

我们按每次进程开始、结束时输出测试语句，并且结束时输出当前存在的进程，这样可以清晰得到每个父进程和子进程的执行顺序。

创建子进程后添加或不添加wait()语句，观察其输出结果的区别。





程序的运行结果如图所示。（可放大查看左边输出结果）

3. 跟踪 fork 函数的执行过程,分析运行结果。

（三）调试分析与测试结果

我们可以很明显观察到输出结果的不同。每个创建的进程都有其pid，并且在统一时间创建的进程ID顺序递增，结束时PID会空出。将输出结果制成图表可以发现，wait()函数使子进程结束之后父进程再继续执行，使得测试结果中的PID更有规律，而没有wait()时父进程和子进程并发执行。

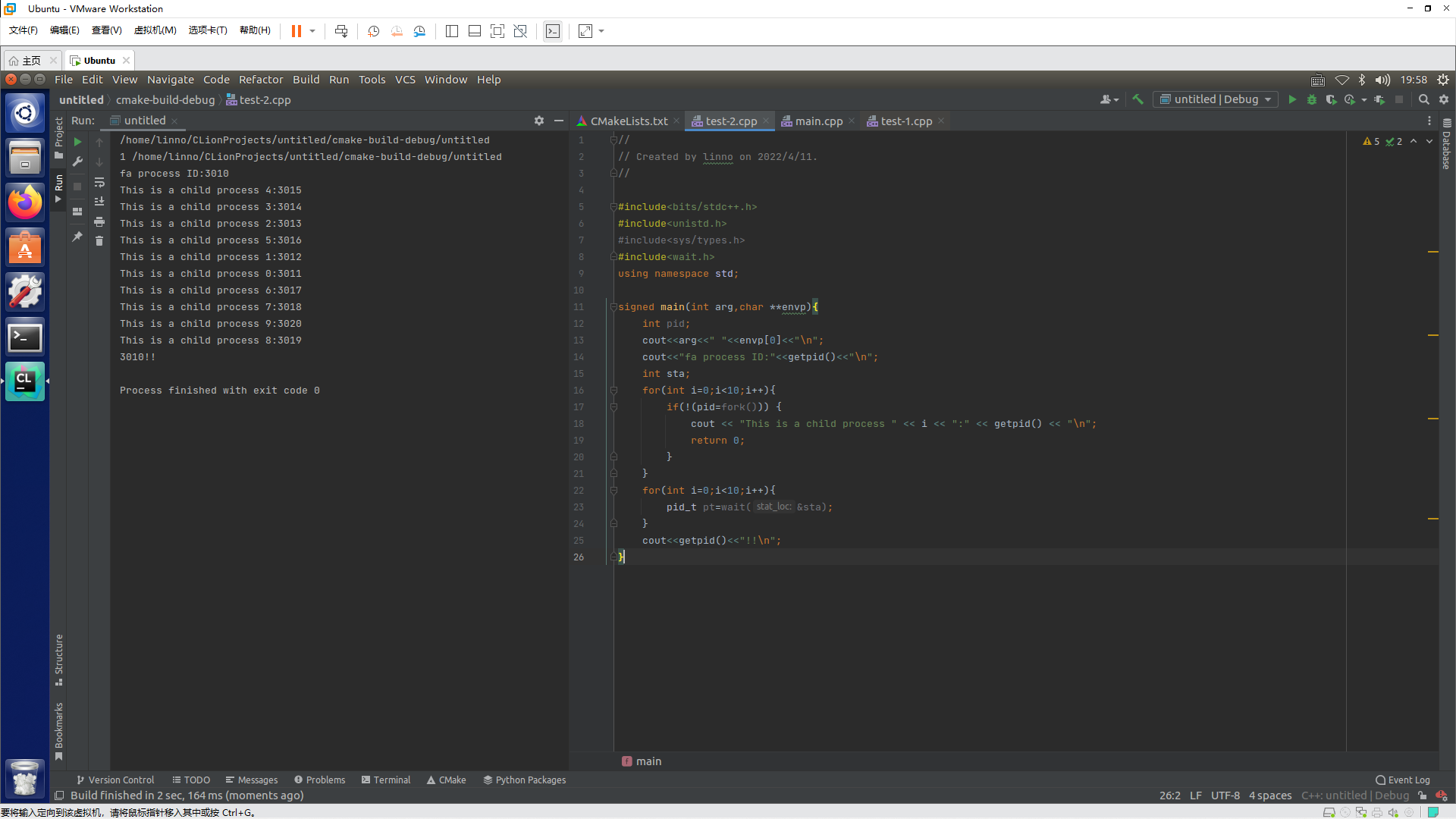
（注意，该表以输出顺序列出，而每次进程结束都会输出进程列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 非等待子进程 | PID | 等待子进程 | PID |
| Parent | 101267 | Child | 101261 |
| Child | 101168 | Child | 101262 |
| Parent | 101167 | Parent | 101261 |
| Parent | 101168 | Parent | 101260 |
| Child | 101170 | Child | 101263 |
| Child | 101169 | Parent | 101260 |

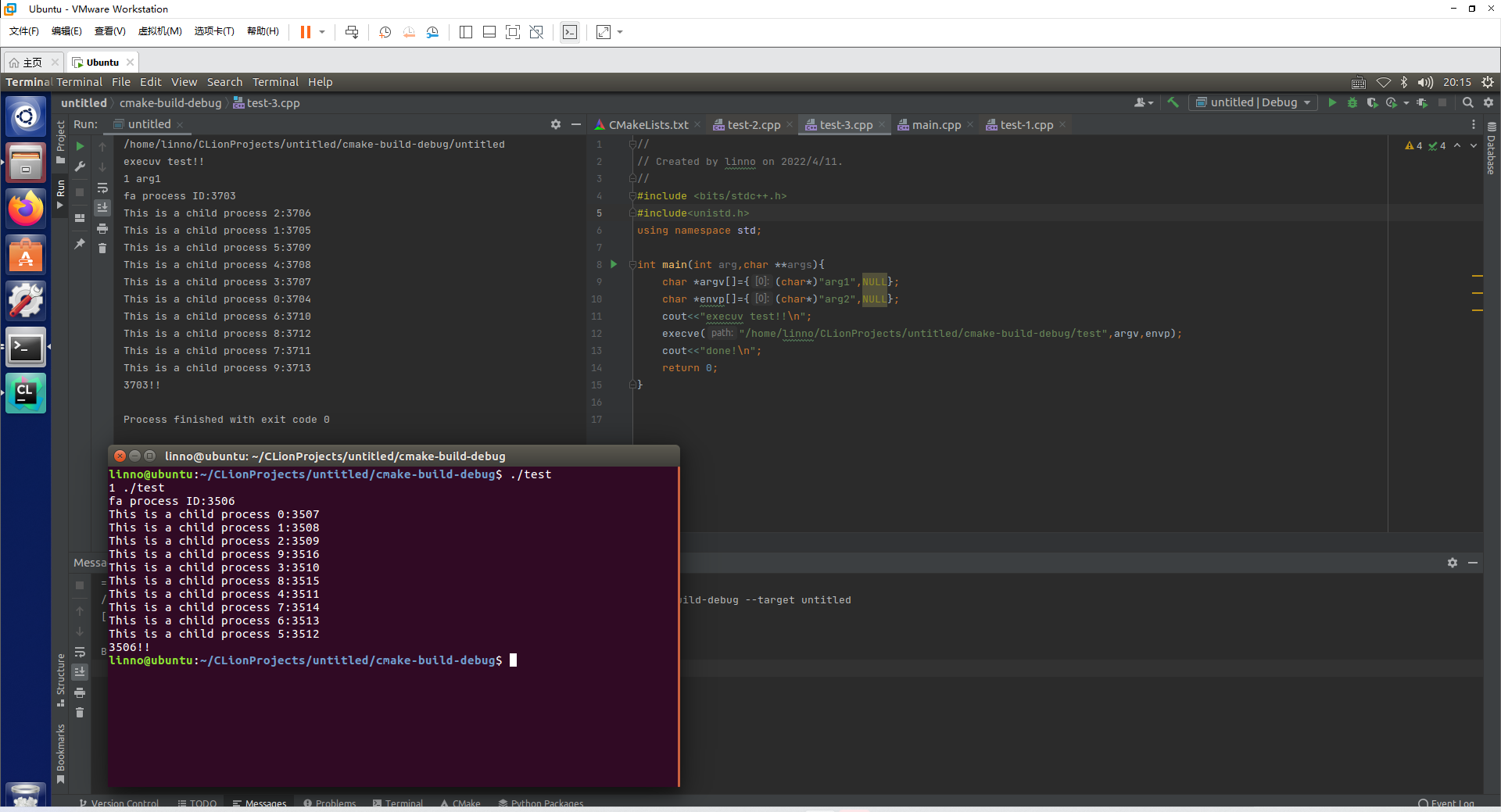
3.调用execve函数观察进程关系

利用execve函数进行调试。通过execve函数启用另外的程序创建新的进程则会使当前程序退出。通过调试结果父进程不会因为子进程的创建而消失，而是在所有子进程结束后父进程才会退出，并输出自己的pid和exit消息。

总结 execve函数的使用方法： execve([执行文件],[传递参数],[环境参数])

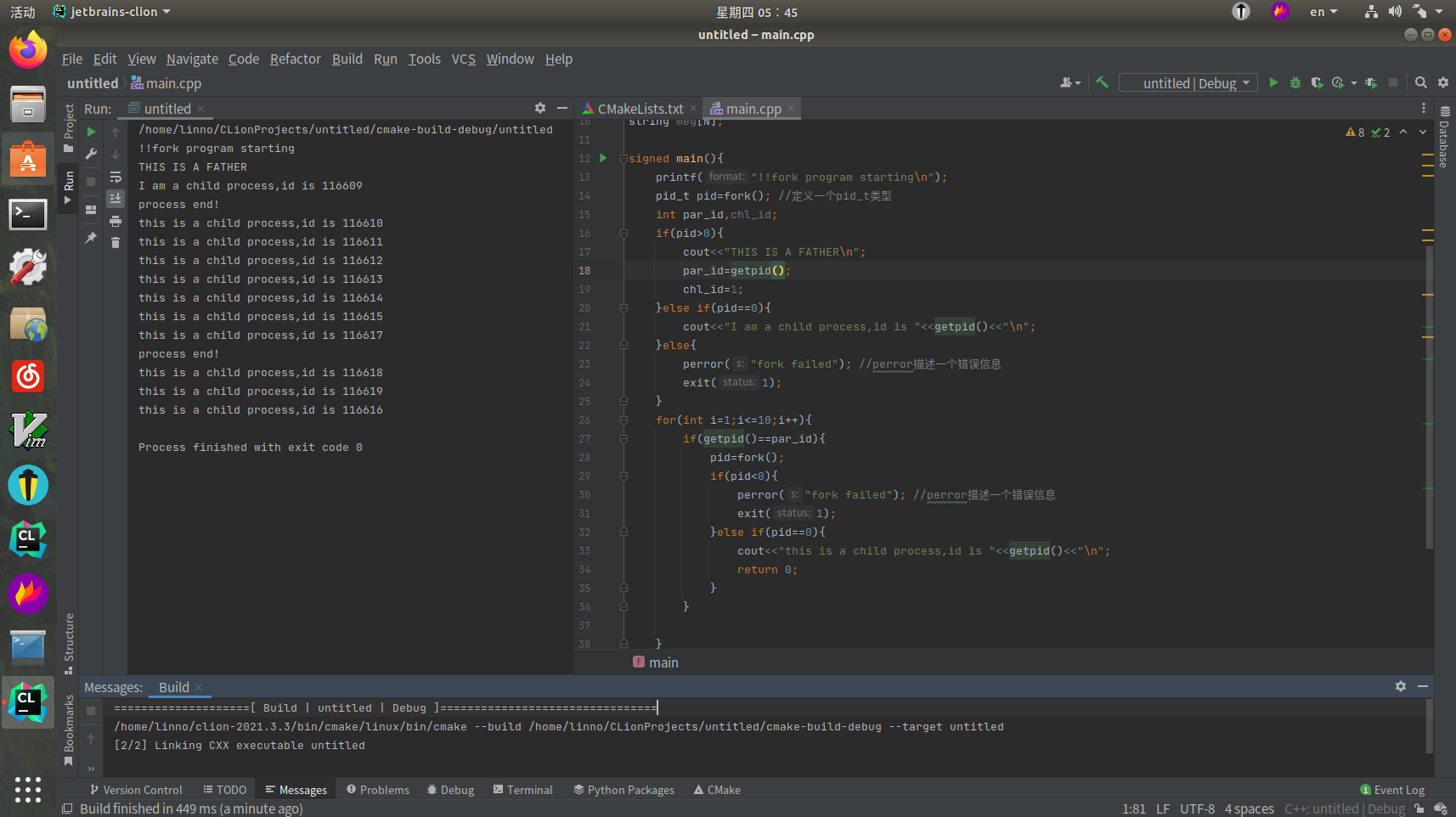
源程序：

调用程序：



## 四、 思考与练习

1.在 Linux应用程序中使用for语句编写一个循环，使父进程能够循环创建10个子进程， 每个子进程在输出自己的pid后退出，父进程等待所有子进程 结束后再退出。



2.代码

1. #include <iostream>
2. #include <sys/types.h>
3. #include<sys/wait.h>
4. #include <stdio.h>
5. #include <unistd.h>
6. **using** **namespace** std;
7. **const** **int** N=1e5+7;
9. **int** num[N],cnt=0;   //进程数
10. string meg[N];
12. **signed** main(){
13. printf("!!fork program starting\n");
14. pid\_t pid=fork(); //定义一个pid\_t类型
15. **int** par\_id,chl\_id;
16. **if**(pid>0){
17. cout<<"THIS IS A FATHER\n";
18. par\_id=getpid();
19. chl\_id=1;
20. }**else** **if**(pid==0){
21. cout<<"I am a child process,id is "<<getpid()<<"\n";
22. }**else**{
23. perror("fork failed"); //perror描述一个错误信息
24. exit(1);
25. }
26. **for**(**int** i=1;i<=10;i++){
27. **if**(getpid()==par\_id){
28. pid=fork();
29. **if**(pid<0){
30. perror("fork failed"); //perror描述一个错误信息
31. exit(1);
32. }**else** **if**(pid==0){
33. cout<<"this is a child process,id is "<<getpid()<<"\n";
34. **return** 0;
35. }
36. }
38. }
39. cout<<"process end!\n";
40. **return** 0;
41. }

### 参考资料:

Linux 内核实验教程 英真时代 编著

……