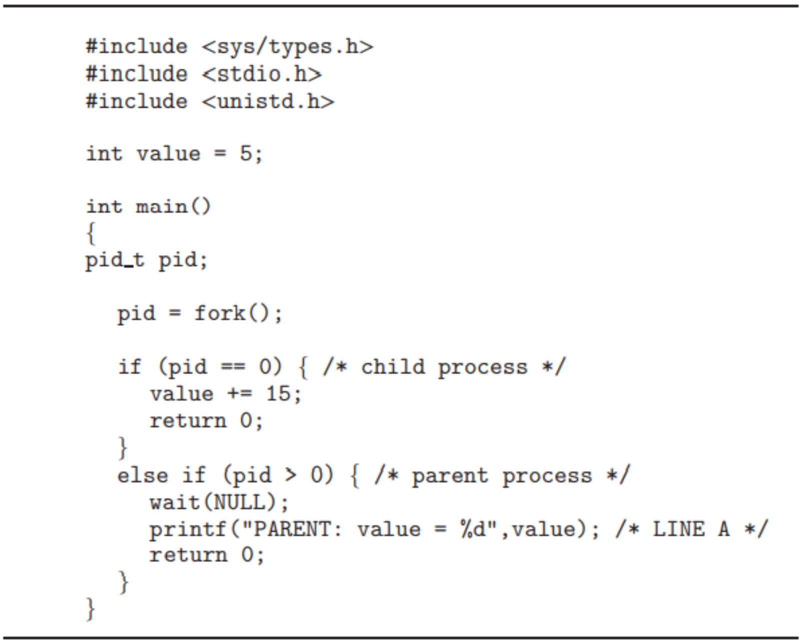
第3章作业

3.1 Using the program shown in Figure 3.30, explain what the output will be at LINE A.



```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>

int value = 5;

int main()
{
pid_t pid;

   pid = fork();

   if (pid == 0) { /* child process */
      value += 15;
      return 0;
   }
   else if (pid > 0) { /* parent process */
      wait(NULL);
      printf("PARENT: value = %d",value); /* LINE A */
      return 0;
   }
}
```

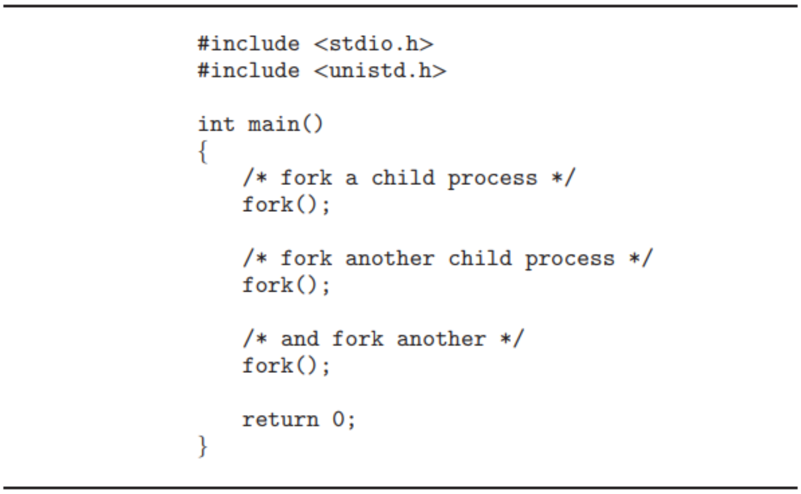
Figure 3.30 What output will be at Line A?

答：输出将会是

```
PARENT: value = 5
```

value为全局变量，存储在数据段，属于初始化数据，子进程会复制父进程的初始化数据，子进程中的value改变不影响父进程。两个进程是相互独立的。

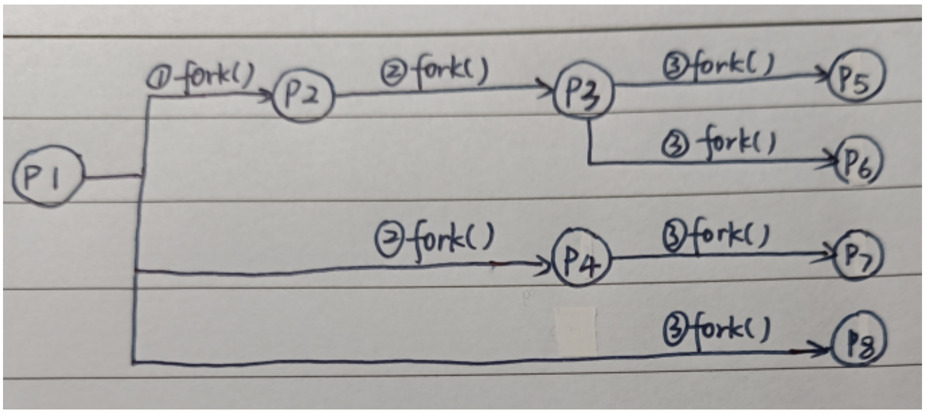
3.2 Including the initial parent process, how many processes are created by the program shown in Figure 3.31?



```
#include <stdio.h>
#include <unistd.h>

int main()
{
    /* fork a child process */
    fork();

    /* fork another child process */
    fork();

    /* and fork another */
    fork();

    return 0;
}
```

Figure 3.31 How many processes are created?

答：8个。



3.4 Some computer systems provide multiple register sets. Describe what happens when a context switch occurs if the new context is already loaded into one of the register sets. What happens if the new context is in memory rather than in a register set and all the register sets are in use?

答：如果需要执行的新进程的上下文已经加载到了几个寄存器组中，那么上下文切换只需简单改变当前寄存器组的指针。如果新进程的上下文在内存中而不是寄存器组中而且所有寄存器都被占用，则需要等有寄存器空闲后在寄存器与内存之间进行数据复制。

3.8 Describe the actions taken by a kernel to context-switch between processes.

答：当进行上下文切换时，内核会将旧进程的状态保存在其PCB中，然后加载经调度而要执行的新进程的上下文。

3.10 Explain the role of the init (or systemd) process on UNIX and Linux systems in regard to process termination.

答：如果父进程没有调用 wait() 就终止，导致子进程成为孤儿进程，这时 init 进程会作为孤儿进程的父进程。init进程定期调用 wait()，以便收集任何孤儿进程的退出状态，并释放孤儿进程的标识符和进程表条目。