

第 3 次作业

李子龙 518070910095

2021 年 3 月 12 日

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

```
1  #include <sys/types.h>
2  #include <stdio.h>
3  #include <unistd.h>
4  int value = 5;
5  int main()
6  {
7      pid_t pid;
8      pid = fork();
9      if (pid == 0) { /* child process */
10         value += 15;
11         return 0;
12     }
13     else if (pid > 0) { /* parent process */
14         wait(NULL);
15         printf("PARENT: value = %d",value); /* LINE A */
16         return 0;
17     }
18 }
```

答: 将会输出 PARENT: value = 5。如果在第 10 行后添加:

```
1  printf("CHILD: value = %d", value);
```

那么将会补充输出: CHILD: value = 20。使用了 fork() 函数后, 新进程复制了原来进程的地址空间, 子进程修改了本进程内的 value 值, 这种复制是真复制而不是引用复制, 因此并不会改变父进程的 value 值, 故原进程的 value 仍然是 5。

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

```

1  #include <stdio.h>
2  #include <unistd.h>
3  int main()
4  {
5      /* fork a child process */
6      fork();
7      /* fork another child process */
8      fork();
9      /* and fork another */
10     fork();
11     return 0;
12 }

```

答: 总进程数是 8 。 fork() 函数之后父子进程都会继续进行下面的代码片段。那么经过第一个 fork() 之后, 变为 2 个进程, 然后变为 4 个进程, 最后变为 8 个进程。

- 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: 对于拥有多寄存器组的机器, 如果新的上下文就在某一个寄存器中的话, 就直接改变当前寄存器组的指针。如果所有的寄存器组都满了, 系统需要像以前一样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: 内核发出中断请求, 然后系统保存当前运行在 CPU 上的进程的上下文, 内核运行结束后, 恢复原进程。

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

答: init 进程定期调用 wait(), 以便收集任何孤儿进程的退出状态, 然后将其作为 init 的子进程, 释放孤儿进程标识符和进程表示条目。