1：输出：PARENT：value=5

原因：父进程通过fork()产生了个子进程，子进程复制拷贝了父进程的内容。即使子进程执行pid==0中的内容改变了value，那也是只改变了子进程内存空间中的value，

对父进程的value没有影响。所有父进程执行pid>0时value输出仍为5。

2：8个进程。

原因：第一次fork()，两个进程，第二次fork(),四个进程，第三次fork(),8个进程。子

进程由于直接拷贝了父进程的内容，所以也拷贝了父进程的程序计数器的内容。Fork()完后，子进程从父进程当前执行的位置接着执行。

3：共享内存段

4：先保存当前进程状态：相关寄存器内容，堆栈指针，程序计数器等到PCB中

选择新的要执行的进程

将要执行的进程的状态从PCB中加载恢复：新进程的寄存器值，文件描述符表，内存表…

恢复新进程的PC，设置为新进程的入口点

到入口点处执行

5：普通管道更适合：父子进程间快速传递处理数据，shell进程可以使用普通管道将命令传

递给其他进程，能够执行复杂的任务。ls | grep “right” ls的输出通过普

通管道给父进程ls，ls再通过普通管道将其传递给另一个子进程grep，

让ls命令的输出成为grep命令的输入，提高命令的执行效率

命名管道更适合：多个进程之间持续，同步通信。分布在不同机器上的进程用命名管道

互相通信

6：A：0 对于子进程而言，fork()返回值pid为0

B：2603 getpid()得到子进程的pid

C：2603 对于父进程而言，fork()返回值为子进程的pid

D：2600