**简单区别回答**：

在任何函数调用return时候，是函数返回。当在main函数中调用就是程序结束；在任何函数调用exit，都会结束程序。

exit（）

函数名: exit()

所在头文件：stdlib.h(如果是”VC6.0“的话头文件为：windows.h)

功 能: 关闭所有文件，终止正在执行的进程。

exit(0)表示正常退出，

exit(x)（x不为0）都表示异常退出，这个x是返回给操作系统（包括UNIX,Linux,和MS DOS）的，以供其他程序使用。

stdlib.h: void exit(int status);//参 数status，程序退出的返回值

\_exit()

直接使进程终止运行，清除其使用的内存空间，并销毁其在内核中的各种数据结构。

return（）

return 表示从被调函数返回到主调函数继续执行，返回时可附带一个返回值，由return后面的参数指定。 return通常是必要的，因为函数调用的时候计算结果通常是通过返回值带出的。 如果函数执行不需要返回计算结果，也经常需要返回一个状态码来表示函数执行的顺利与否（-1和0就是最常用的状态码），主调函数可以通过返回值判断被调函数的执行情况。

1.exit用于结束正在运行的整个程序，它将参数返回给OS，把控制权交给操作系统；而return 是退出当前函数，返回函数值，把控制权交给调用函数。如果main()在一个递归程序中，exit()仍然会终止程序；但return将控制权移交给递归的前一级，直到最初的那一级，此时return才会终止程序。

2.exit是系统调用级别，它表示一个进程的结束；而return 是语言级别的，它表示调用堆栈的返回。

3. 在main函数结束时，会隐式地调用exit函数，所以一般程序执行到main()结尾时，则结束主进程。exit将删除进程使用的内存空间，同时把错误信息返回给父进程。

return和exit()的另一个区别在于，即使在除main()之外的函数中调用exit()，它也将终止程序。

**1、exit函数和return函数的主要区别是：**

    1）exit用于在程序运行的过程中随时结束程序，exit的参数是返回给OS的。**main函数结束时也会隐式地调用exit函数。exit函数运行时首先会执行由atexit()函数登记的函数，然后会做一些自身的清理工作，同时刷新所有输出流、关闭所有打开的流并且关闭通过标准I/O函数tmpfile()创建的临时文件。**exit是结束一个进程，它将删除进程使用的内存空间，同时把错误信息返回父进程；而return是返回函数值并退出函数。通常情况：exit(0)表示程序正常, exit(1)和exit(-1)表示程序异常退出，exit(2)表示表示系统找不到指定的文件。在整个程序中，只要调用exit就结束（当前进程或者在main时候为整个程序）。

    2）**return是语言级别的，它表示了调用堆栈的返回**；return( )是当前函数返回，当然如果是在主函数main, 自然也就结束当前进程了，如果不是，那就是退回上一层调用。在多个进程时。如果有时要检测上个进程是否正常退出。就要用到上个进程的返回值，依次类推。而**exit是系统调用级别的，它表示了一个进程的结束**。

    3）exit函数是退出应用程序，并将应用程序的一个状态返回给OS，这个状态标识了应用程序的一些运行信息。

    4）和机器和操作系统有关的一般是： 0为正常退出，非0为非正常退出；

**2、进程环境与进程控制**

     exit(int n)其实就是直接退出程序，因为默认的标准程序入口为 int main(int argc, char\*\* argv)，返回值是int型的。一般在shell下面，运行一个程序，然后使用命令echo $?就能得到该程序的返回值，也就是退出值，在main()里面，你可以用return n，也能够直接用exit(n)来做。unix默认的正确退出是返回0，错误返回非0。理论上exit可以返回小于256的任何整数。返回的不同数值主要是给调用者作不同处理的。

    单独的进程是返回给操作系统的。如果是多进程，是返回给父进程的。父进程里面调用waitpid()等函数得到子进程退出的状态，以便作不同处理。根据相应的返回值来让调用者作出相应的处理。总的说来，exit()就是当前进程把控制权返回给调用该程序的程序，括号里的是返回值，告诉调用程序该程序的运行状态。

**1）进程的开始：**

     C程序是从main函数开始执行，原型如下:int main(int argc, char \*argv[])；通常main的返回值是int型，正确返回0。如果main的返回值为void，某些编译器会给出警告，此时main的返回值通常是0。

**2）进程终止：**

     C程序的终止分为两种：正常终止和异常终止。

     正常终止分为：return，exit，\_exit，\_Exit，pthreade\_exit。异常中指分为：abort，SIGNAL，线程响应取消。  
     主要说一下正常终止的前4种：即exit系列函数.  
     #include <stdlib.h>     
     void exit(int status);  
     void \_Exit(int status);

     #include <unistd.h>

     void \_exit(int status);

     以上3个函数的区别是：exit()(或return 0)会调用终止处理程序和用户空间的标准I/O清理程序(如fclose)，\_exit和\_Exit不调用而直接由内核接管进行清理。因此，在main函数中exit(0)等价于return 0。

**3）atexit终止处理程序：**

     ISO C规定，一个进程最多可登记32个终止处理函数，这些函数由exit按登记相反的顺序自动调用。如果同一函数登记多次，也会被调用多次。  
     原型如下:

     #include <stdlib.h>

     int atexit(void (\*func)(void));

     其中参数是一个函数指针，指向终止处理函数，该函数无参无返回值。atexit函数本身成功调用后返回0。

     以下面的程序为例:

**[cpp]** [view plaincopyprint?](https://www.cnblogs.com/noble/p/4144166.html)

1. #include <stdlib.h>
2. **static** **void** myexit1()
3. {
4. printf("first exit handler\n");
5. }
6. **static** **void** myexit2()
7. {
8. printf("second exit handler\n");
9. }
10. int main()
11. {
12. atexit(my\_exit2)；
13. atexit(my\_exit1)；
14. atexit(my\_exit1)；
15. printf("main is done\n");
16. **return** 0;
17. }

     运行结果:  
     $ ./a.out  
     main is done  
     first exit handler  
     first exit handler  
     second exit handler

     注意上面的结果，可以发现这些函数由exit按登记相反的顺序自动调用（先myexit1后myexit2）。如果同一函数登记多次, 也会被调用多次（如这里的myexit1）。程序退出的时候利用atexit函数调用了这些处理函数。但是如果用\_exit()退出程序，则它不关闭任何文件，不清除任何缓冲器、也不调用任何终止函数！