** 11.8　修改 TINY，使得它在 SIGCHLD 处理程序中回收操作系统分配给 CGI 子进程的资源，而不是显式地等待它们终止。

** 11.9　修改 TINY，使得当它服务静态内容时，使用 malloc、rio_readn 和 rio_writen，而不是 mmap 和 rio_writen 来复制被请求文件到已连接描述符。

** 11.10　A. 写出图 11-27 中 CGI adder 函数的 HTML 表单。你的表单应该包括两个文本框，用户将需要相加的两个数字填在这两个文本框中。你的表单应该使用 GET 方法请求内容。

B. 用这样的方法来检查你的程序：使用一个真正的浏览器向 TINY 请求表单，向 TINY 提交填写好的表单，然后显示 adder 生成的动态内容。

** 11.11　扩展 TINY，以支持 HTTP HEAD 方法。使用 TELNET 作为 Web 客户端来验证你的工作。

*** 11.12　扩展 TINY，使得它服务以 HTTP POST 方式请求的动态内容。用你喜欢的 Web 浏览器来验证你的工作。

*** 11.13　修改 TINY，使得它可以干净地处理(而不是终止)在 write 函数试图写一个过早关闭的连接时发生的 SIGPIPE 信号和 EPIPE 错误。

练习题答案

11.1

十六进制地址	点分十进制地址
0x0	0.0.0.0
0xffffffff	255.255.255.255
0x7f000001	127.0.0.1
0xcdbca079	205.188.160.121
0x400c950d	64.12.149.13
0xcdbc9217	205.188.146.23

11.2

code/netp/hex2dd.c

```
1   #include "csapp.h"
2
3   int main(int argc, char **argv)
4   {
5       struct in_addr inaddr;  /* Address in network byte order */
6       uint32_t addr;          /* Address in host byte order */
7       char buf[MAXBUF];       /* Buffer for dotted-decimal string */
8
9       if (argc != 2) {
10          fprintf(stderr, "usage: %s <hex number>\n", argv[0]);
11          exit(0);
12      }
13      sscanf(argv[1], "%x", &addr);
14      inaddr.s_addr = htonl(addr);
15
16      if (!inet_ntop(AF_INET, &inaddr, buf, MAXBUF))
17          unix_error("inet_ntop");
18      printf("%s\n", buf);
19
20      exit(0);
21  }
```

code/netp/hex2dd.c

11.3

code/netp/dd2hex.c

```
1   #include "csapp.h"
2
3   int main(int argc, char **argv)
4   {
```