Prosty serwer TCP (udaje serwer http)

```
#include <sys/socket.h>
#include <netinet/in.h>
#include <stdio.h>
#include <strings.h>
#include <string.h>
#include <stdlib.h>

/* makro, które załatwi nam diagnostykę */
#define CALL(x) {int r = (x); if(r != 0){printf("\'"#x"' returned %d\n",4);exit(1);}}

int main(int argc, char**argv)
{
   int         listenfd, connfd, n;
   struct      sockaddr_in servaddr,cliaddr;
   socklen_t   clilen;
   pid_t       childpid;
   char        mesg[1000];
```

```
listenfd = socket(AF_INET, SOCK_STREAM, 0);
n = 1;

/* przygotowujemy gniazdo do ponownego użycia bez konieczności zamykania */
CALL( setsockopt(listenfd, SOL_SOCKET, SO_REUSEADDR, &n, sizeof(n)) );

bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family = AF_INET;

/* dowiązujemy się do wszystkich interfejsów systemu */
servaddr.sin_addr.s_addr=htonl(INADDR_ANY);

/* port nie jest szczególnie standardowy, ale za to bezpieczny */
servaddr.sin_port=htons(32000);

/* prosimy system o dowiązanie naszej usługi */
CALL( bind(listenfd, (struct sockaddr *)&servaddr, sizeof(servaddr)) );
```

```
    /* i rozpoczynamy nasłuch */
    CALL( listen(listenfd, 5) );

    for(;;)
    {
       clilen = sizeof(cliaddr);
       connfd = accept(listenfd,(struct sockaddr *)&cliaddr,&clilen);

      /* jeżeli jesteśmy w tym miejscu, to znaczy, że przybył klient */
       puts("connection!");
       read(connfd, mesg, 1);
       char *response =  "HTTP/1.1 200 OK\r\n"
                         "Content-Type: text/html\r\n\r\n"
                         "<html><body><h1>HELLO WORLD!</h1></body></html>\n\n";
       write(connfd, response, strlen(response));
       close(connfd);
    }
}
```