Simple TCP/IP echo server

This program demonstrates a simple TCP/IP server. It will accept a connection from a client application, receive one line of text, echo that line back to the client and close the connection.

Programming Issues

This program illustrates the classic process for a TCP/IP server program. Summarised, it is as follows:

- Create a socket with a call to `socket()`
- Create and initialize a **socket address strucure** with the IP address set to`INADDR_ANY` (the server will listen on any IP address) and the port to whichever one you wish to use.
- Call `bind()` to bind the socket address to the socket.
- Call `listen()` to indicate that this is a **passive socket**, that we want to accept incoming requests rather than make outgoing ones.
- Enter a loop in which we:
    1. Call `accept()` to wait for an incoming connection
    2. Service the request on our new connection
    3. `close()` our connection, and continue the loop to wait for a new one

Note that it's present form, the server enters an infinite loop and has no mechanism for closing itself. We have to use a command such as `kill` to terminate it.

Usage

Once launched in the background, the server can be used either with a dedicated echo client, or simply by telnetting to it. For the latter approach, the following is a sample session using Linux:

```
[paul@localhost paul]$ ./echoserv 3357 &[paul@localhost paul]$ telnet localhost 3357Trying 127.0.0.1...Connected to localhos
```

After executing telnet and receiving some initialisation output, we enter the words "Echo this line for me, please.", and we see the server process echoing them back to us, then closing the connection.

You can specify the port number to listen to as a command line argument. The program will default to ECHO_PORT (#defined as 2002) if one is not supplied.