Per la corretta visualizzazione della pagina aprire un terminale e digitare:

```
    cd /tmp
    export http_proxy=
    elinks http://localhost:PORTA
```

```
//Marco Zen
//5AI
//27/11/2019

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <signal.h>
#include <fcntl.h>
#include <sys/wait.h>

#define PORT 8888
#define MAX 10

int client[MAX];
int create_socket;

socklen_t addrlen;
struct sockaddr_in address;

void connessione(int x)
{
    printf("PID Figlio %d\n", getpid());
    printf("Client connesso\n");
    write(client[x], "HTTP/1.1 200 OK\n", 16);
    write(client[x], "Content-length: 46\n", 19);
    write(client[x], "Content-Type: text/html\n\n", 25);
    write(client[x], "<html><body><H1>Hello world</H1></body></html>",46);

    printf("Chiusura socket\n\n");
    shutdown(client[x], SHUT_RDWR);
    close(client[x]);
    client[x] = -1;
}

void startServer()
{
```

```
    address.sin_family = AF_INET;
    address.sin_addr.s_addr = INADDR_ANY;
    address.sin_port = htons(PORT);

    if ((create_socket = socket(AF_INET, SOCK_STREAM, 0)) > 0)
    {
        printf("socket()\n");
    }
    else
    {
        perror("socket() fallito\n");
        exit(0);
    }

    if (bind(create_socket, (struct sockaddr *) &address, sizeof(address)) == 0)
    {
        printf("bind()\n");
    }
    else
    {
        perror("bind()");
        exit(0);
    }

    if (listen(create_socket, 1000000) != 0)
    {
        perror("listen() fallito");
        exit(1);
    }
}

int main()
{
    int i;
    int slot = 0;

    printf("PID Padre: %d\n", getpid());
    printf("Server in avvio nella porta %d\n", PORT);

    for(i = 0; i < MAX; i++) //Setto tutti i possibili client a -1
    {
        client[i] = -1;
    }

    startServer();

    while(1)
    {
        addrlen = sizeof(address);
```

```
        client[slot] = accept(create_socket, (struct sockaddr *) &address, &addr]

        if (client[slot] < 0)
        {
            perror("accept() fallito");
            exit(1);
        }
        else
        {
            int pid = fork();

            if(pid == 0) //Codice Figlio, cioè codice del client connesso
            {
                printf("Esecuzione fork()\n");
                connessione(slot);
                wait(NULL);
                exit(0);
            }
        }

        while(client[slot] != -1)
        {
            slot = (slot + 1) % MAX;
        }
    }
    close(create_socket);
    return 0;
}
```