João Pedro Oliveira Loureiro 113109

Sistemas Computacionais - Ficha 8

II

Cliente:

```
int main (int argc, char * const argv[]){
    char buf[4096];
    if(argc != 4){
      printf("Usage: %s server-ip-address port-number filename\n",
             argv[0]);
    exit(1);
  }
FILE *fp = fopen(argv[3], "r");
if (fp == NULL){
      perror("fopen");
      exit(1);
    }
int socket_descriptor = my-connect-to-server(argv[1], argv[2]);
while(1){
int n = fread(buf, 1, sizeof(buf), fp);

    int(n == 0)
          break;

    write(socket_descriptor, buf, n);
  }
fclose(fp);

return 0;
}
```

Servidor:

```
int main(int argc, char *argv[]){
 int new_socket_description, socket_description, n;
 unsigned int clilen;
 struct sockaddr_in cli_addr;

if (argc != 2){
  printf("Usage: %s port_number\n", argv[0]);
  exit 1;
}

 signal(SIGPIPE, SIG_IGN);
 socket_description = create_server_socket(atoi(argv[1]));

 while(1){
  char c;
  char buffer[4096];
  char filename[1024];

  printf("Waiting connections\n");
  clilen = sizeof(cli_addr);
  new_socket_description = accept(socket_description, (struct sockaddr*)
        &cli_addr, &clilen);

  sigaction(SIGCHLD, SIG_IGN);

  n = fork();
```

João Pedro Oliveira Loureiro 113109

Sistemas computacionais - Ficha 8

Servidor (cont.):

```
if (r == 0) {
  printf("Connection from %o S\n", inet_ntoa(*((struct in_addr *)
      &(cli_addr.sin_addr))));

  strcpy(filename, "file XXXXXXXX.bin");
  int fdd = mkstemps(filename, 4);

  FILE *fp = fdopen(fdd, "r+");

  while (1) {
  n = read(new_socket_descriptor, buf, sizeof(buf));

  if(n == 0)
      break;

    fwrite(buf, n, 1, fp);
  }
fclose(fp);
exit(-1);
}
close(new_socket_descriptor);
}

return 0;
}
```

João Pedro Oliveira Lourenço 1131109

Sistemas Computacionais – Ficha 8

## III

```
int main(int argc, char* const argv[]) {

    char buffer[4096];

    char http_msg1[] = "GET /~jcs/sistc/pl/ola.html HTTP/1.1\r\n"
    "Host: www.dee.isep.ipp.pt\r\n\r\n";

    int s = my_connect_to_server("ave.dcc.inp.ipp.pt", "80");

    write(s, http_msg1, strlen(http_msg1));

    int n;
    int flag=0; // Flag que vai ser activa após encontrar linha em branco

    FILE *fp = fdopen(s, "r+");

    while (fgets(buffer, sizeof(buffer), fp) != NULL) {
        if (flag==0){
        int blank = strcmp(buffer, "\r\n");
        if (blank==0) { // strcmp retorna 0 quando s1 = s2
            flag=1;
        }
    }
    else {
        fputs(buffer, stdout);
    }
}
return 0;
}
```