Teste - 2/6/16 - SO

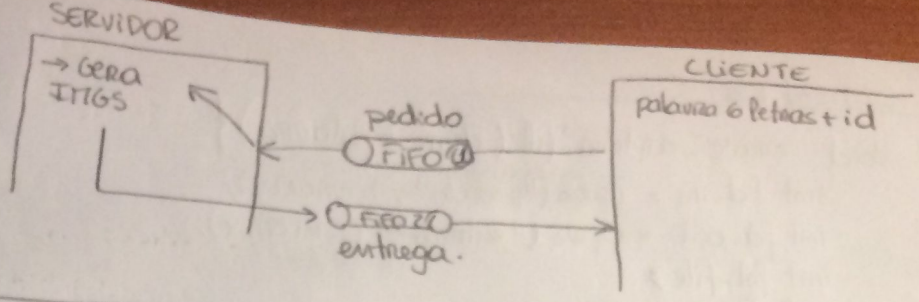
II

```
void hAlarm () {
    char buffer[3] = "OK\n";
    write (1, buffer, 3);
}

void hAlarm2() {
    char buffer[17] = "Passaram 3 secs\n";
    write (1, buffer, 17);
}

void processoTeste () {
    int nVezes = 0;
    signal (SIGALRM, hAlarm);
    while ( nVezes < 4) {
        alarm(4);
        pause();
        nVezes++;
    }
    pause();
}

void ctrl ( int nProc, char **procName) {
    int i, R, count = 0;
    int fds [nProc][2];
    int pids [nProc];
    char buffer[1024], trash[5] = "trash", aux[17] = "Processo Inativo\n";

    signal (SIGALRM, hAlarm2);

    for (i=0; i< nProc; i++) {
        pipe (fds[i]);
        if ( (pids[i] = fork() == 0 ){
            dup2(fds[i][1], 1);
            close (fds[i][0]);
            close (fds[i][1]);
            close (0);

            execlp(procName[i], procName[i], NULL);
            - exit (-1);
        }
    }
```

```
while( count != mProc){
    alarm(4);→ envia ao processo invocador um sinal SIGALRM dentro
                de 4 segundo, mesmo q não esteja a ser executada
    pause();
    for(i=0; i<mProc; i++){
        write(fds[i][1], trash, 5);
        if ((pids[i] != -1) && (r=read(fds[i][0], buffer, 124))==5){
            write(1, aux, 17);
            kill(pids[i], SIGKILL);
            pids[i] = -1;
            count++;
        }
    }
}
```

```
int main (int argc, char **argv){
    crtl(argc-1, argv+1);
}
```

III



CRIA PIPE → FIFO

```
int main() {
  int i;
  if ((i = mkfifo("pedido", 0666)) < 0) {
      perror ("\nERRO");
      return -1;
  }
  return 0;
```

SERVIDOR

```
#define MAX 16384

size_t captch ( char *palavra, char * buffer){
     strcpy (buffer, palavra);
     Return  strlen (buffer);
  }

int main(){
   int fd_in = open ("pedido", O_RDONLY);
   int fd_out = open ("entrega", O_WRONLY);
   int R, i = 0;
   char palavra [1024], buffer [MAX];
   while ( (R = Read (fd_in , palavra+i, 1)) > 0)
         i++;
   palavra [i-1] = '\0';
   R = captch (palavra , buffer);
   write (fd_out , buffer, R);
   close (fd_in);
   close (fd_out);
}
```

CLIENTE

```
void create_captcha_file(char *palavra){
    int fd_im = open("pedido", O_WRONLY);
    int fd_out = open("entrega", O_RDONLY);
    int fd_file;
    int length(palavra), R;
    char filename[length+4];
    char buffer[1024];
    strcpy(fileName, palavra);
    strcat(fileName, ".png");
    fd_file = open(fileName, O_WRONLY | O_CREAT, 0666);
```

Fazer pedido

```
    (write(fd_im, palavra, length);

    while((R = Read(fd_out, buffer, 1024)) > 0){
        write(fd_file, buffer, R);
    }

    close(fd_file);
    close(fd_im);
    Close(fd_out);
}
```

```
int main(int argc, char ** argv){
    create_captcha_file(argv[1]);
}
```

P. Teórica → sobre o tr