Template para Entrega

Nome:Diogo José Real Lavareda

Turma:

**PREENCHA A TABELA: Tamanho do Ficheiro 100MB**

|  |  |  |
| --- | --- | --- |
| TAMANHO do BLOCO | TEMPO CPU lowlevel | TEMPO CPU High Level |
| 4096 | 0.023857 | 0,105859 |
| 1024 | 0.044503 | 0.098666 |
| 256 | 0.157880 | 0.101068 |
| 64 | 0.595811 | 0.102509 |
| 16 | 2.338376 | 0.096460 |
| 4 | 9.436741 | 0.101675 |

## Explicação dos Resultados

**Exercício 1**

( cc -o main main.c soio.c -Wall OU make main )

**Exercício 2– Implementação das funções sofgetc(SOFILE\*) e sofclose(SOFILE\*)**

Exercício Concluída Sim

**Exercício 3– Implementação da função sofflush(SOFILE\*) e o seu test**

Exercício Concluída Sim

Código do ficheiros soio.c

#include "soio.h"

SOFILE \*sofopen(const char \*nome, const char \*mode)

{

SOFILE \*novo = NULL;

int fd;

if (mode[0] == 'r')

fd = open(nome, O\_RDONLY, 0);

else

exit(1); //agora apenas para leitura !

if (-1 == fd)

return NULL; //caso erro

novo = (SOFILE \*)malloc(sizeof(SOFILE));

novo->buf = (char \*)malloc(MAXBUFFSIZE);

novo->fd = fd;

novo->index = 0;

novo->size = 0;

return novo;

}

int sofclose(SOFILE \*fp)

{

close(fp->fd);

free(fp->buf);

free(fp);

return 0;

}

int sofgetc(SOFILE \*fp)

{

if (fp->index < 16 && fp->index > 0)

{

return fp->buf[fp->index++];

}

else

{

fp->index = 0;

if (read(fp->fd, fp->buf, 16) == 0)

{

return EOF;

}

else

{

return fp->buf[fp->index++];

}

}

}

int sofflush(SOFILE \*fp)

{

memset(fp->buf, 0, 16 \* sizeof(char));

fp->index = 0;

fp->size = 0;

return 0;

}