FUNDAÇÃO UNIVERSIDADE DE CAXIAS DO SUL
ÁREA DO CONHECIMENTO DE CIÊNCIAS EXATAS E ENGENHARIAS
FBI4013AB - Programação de Computadores II
PROVA III, 05 DE JULHO DE 2023

Nome: _____

1) (3.3 pontos) Fazer uma função escrita na linguagem de programação C que escreva todos os nomes dos alunos armazenados no arquivo 'alunos.dat' em ordem contrária, ou seja, do último aluno gravado até o primeiro. Abaixo exemplo de um programa que armazena alguns alunos nesse arquivo e algumas funções que podem ser usadas. Considere que o tamanho do arquivo é tal que não é possível carregá-lo todo para um vetor.
fseek(arq, 0, SEEK_SET) // posiciona na posição 0 (deslocamento em bytes) a partir do início do arquivo (SEEK_SET). Outras opções SEEK_END (fim do arquivo), SEEK_CUR (posição corrente).
ftell(arq) // Retorna a posição corrente no arquivo(em bytes) a partir da posição 0 (primeira posição do arquivo)
fread(&variavel, sizeof(variavel), 1, arq) lê para a variável a partir da posição corrente do arquivo. Retorna quantos foram lidos, nesse caso 0 se não foi lido, ou 1 se foi lido.

```
#include <stdio.h>
#include <stdlib.h>

struct st_aluno {
    int codaluno;
    char nomealuno[41];
    float notas[3];
    int faltas;
};
typedef struct st_aluno TIPOALUNO;

void cria_arq_alunos() {
  FILE *arq;
  TIPOALUNO aluno;
  arq=fopen("alunos.dat","wb");
  if(arq==NULL) {
    printf("Problema ao criar alunos.dat\n");
    exit(2);
  }
  aluno=(TIPOALUNO) { 2,"pedro",  {8.6,9.2,7.5}, 0}; fwrite(&aluno,sizeof(aluno),1,arq);
  aluno=(TIPOALUNO) { 3,"ana",    {9.1,9.0,9.4}, 2}; fwrite(&aluno,sizeof(aluno),1,arq);
  aluno=(TIPOALUNO) { 5,"roberto",{6.0,6.2,6.6}, 3}; fwrite(&aluno,sizeof(aluno),1,arq);
  aluno=(TIPOALUNO) { 8,"maria",  {7.9,6.9,8.9}, 1}; fwrite(&aluno,sizeof(aluno),1,arq);
  aluno=(TIPOALUNO) {14,"eduardo",{5.0,4.5,5.1}, 8}; fwrite(&aluno,sizeof(aluno),1,arq);
  aluno=(TIPOALUNO) {18,"leandro",{6.0,6.0,6.0}, 2}; fwrite(&aluno,sizeof(aluno),1,arq);
  aluno=(TIPOALUNO) {32,"paula",  {5.0,9.5,9.8}, 4}; fwrite(&aluno,sizeof(aluno),1,arq);
  fclose(arq);
}
int main() {
  cria_arq_alunos();
  return 0;
}
```

2) (3.4 pontos) Fazer um programa completo escrito na linguagem de programação C que leia um arquivo do tipo texto chamado 'texto01.txt' e gravando todas as informações (exceto as linhas em branco) desse em outro arquivo também do tipo texto chamado 'texto02.txt'. Para facilitar, considere o tamanho máximo de cada linha 1000 caracteres. Assim pode-se ler linha a linha com **fgets(linha, sizeof(linha), arquivo)**. Lembrar que o '\n' também é lido. Veja que essa função retorna NULL quando atingir fim do arquivo, senão retorna um ponteiro para a linha. Para gravar, pode-se utilizar o **fprintf** a função **fputs(linha, arquivograva)**. Outra opção de implementação é ler caractere a caractere com **getc(arquivo)**. Essa função retorna o caracter lido, ou -1 caso seja encontrado fim de arquivo. Aqui pode-se escrever com **putc(caractere, arquivo)**, mas lembrar que precisa controlar linhas em branco e essas não são escritas no arquivo de saída.

3) (3.3 pontos) Considerando que um registro de data e horário é armazenado em uma variável inteira de 32 bits sem sinal, onde os 6 bits menos significativos são o minuto, os 5 bits subsequentes são a hora, a seguir, 5 bits para o dia, depois 4 bits para o mês e finalmente os 12 bits mais significativos para o ano, conforme exemplo. Fazer uma função escrita na linguagem de programação C que receba como parâmetro essa variável (uint32_t), utilize operadores bitwise:
& | ~ ^ << >> para separar as informações e escreva a data no formato: DD/MM/AAAA HH:MM.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
– Ano: 12 bits –												–Mês:4bits–				– Dia: 5 bits –					– Hora: 5 bits –					– Minuto: 6 bits –					
0	1	1	1	1	1	1	0	0	1	1	0	1	1	0	0	0	0	1	1	1	0	0	1	0	1	1	0	1	0	0	0
Exemplo acima: ano 2022												mês 12				dia 7					hora 19					minuto 40					

---- B O A P R O V A ----