Lista de exercício Arquivos

1.) Escreva um programa em C que escreve uma sequencia de caracteres em um arquivo texto e depois retorne ao inicio do arquivo e imprima todos os caracteres que foram escritos no arquivo.

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

int main (void )
{
	int c;
	FILE *pa;
	char *nome = "texto.txt";

	if (( pa = fopen(nome, "w+")) == NULL)
	{
		printf("\n\nNao foi possivel abrir o arquivo para escrita.\n");
		return 1;
	}
	/* Cada caracter digitado ser gravado no arquivo */
	c = getchar();
	while (!feof(stdin))
	{
		fputc(c, pa);
		c = getchar();
	}
	/*fclose(pa); */
	rewind(pa);
	printf("\nTerminei de escrever, agora vou ler.\n");

/*	if (( pa = fopen(nome, "r")) == NULL) {

		printf("\n\nNao foi possivel abrir o arquivo para leitura.\n");

		exit(1);
	}

*/
	c = fgetc(pa);
	while (!feof(pa))
	{
		putchar(c);
		c = fgetc(pa);
	}
	return 0;
}
```

2.) Escreva um programa em C que leia do teclado uma sequencia de caractere e depois escreva esta sequencia em um arquivo. Depois imprima as sequencias de caracteres qe foram gravadas no arquivo.

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#define MAX 80

int main (void )
{
char linha[MAX];

FILE *pa;
char *nome = "texto.txt";

        /* Abre o arquivo para leitura e escrita */
        if (( pa = fopen(nome, "w+")) == NULL)
{
printf("\n\nNao foi possivel abrir o arquivo.\n");
exit(1);
}
        /* Cada linha digitada sera gravada no arquivo */
        fgets(linha, MAX, stdin);
        while (!feof(stdin))
        {
fputs(linha, pa);
fgets(linha, MAX, stdin);
}
rewind(pa);  /* volta ao inicio do arquivo */
printf("\nTerminei de escrever, agora vou ler.\n\n");
fgets(linha, MAX, pa);
while (!feof(pa))
{
puts(linha);
fgets(linha, MAX, pa);
}
   fclose(pa);
   return 0;
}
```

3.) Faça um programa que leia o nome e sobrenome de 30 alunos e armazene em um arquivo, de tal forma que o arquivo tenha um aluno por linha.

```
#include <stdio.h>
#include <stdlib.h>
int main(){
FILE *arq;
char nome[20], sobrenome[20], i;
arq = fopen("t1.txt", "w");
printf("entre nome e sobrenome de 30 alunos : \n");
for (i = 0; i < 30; i++) {
printf("Aluno %d. Entre nome: ");
scanf("%s", nome);
printf("Aluno %d. Entre sobrenome: ");
scanf("%s", sobrenome);
fputs(nome, arq);
fputs(" ", arq);
fputs(sobrenome, arq);
fputs("\n", arq);
}
fclose(arq);
return 0;
}
```

4.) Faça um programa que leia um vetor de inteiros A de tamanho 20 e guarde seus valores em um arquivo, um por linha. Em seguida, reabra o arquivo e leia os elementos para o vetor B, verificando se os valores foram gravados corretamente.

```
#include <stdio.h>
#include <stdlib.h>
int main(){
FILE *arq;
int A[20], B[20], i;
char aux[10];
arq = fopen("t2.txt", "w");
printf("entre 20 numeros: \n");
for (i = 0; i < 20; i++) {
scanf("%d", &A[i]);
fprintf(arq, "%d\n", A[i]);
}
fclose(arq);
arq = fopen("t2.txt", "r");
printf("\nlendo 20 numeros: \n");
for (i = 0; i < 20; i++) {
fscanf(arq, "%d", &B[i]);
printf("%d ", B[i]);
}
fclose(arq);
printf("\n");
return 0;
}
```

5.) Escreva um programa que, dado um arquivo de entrada cujo nome foi fornecido pelo usuario, converta todas as letras minúsculas para maiúsculas. Use as funções abaixo:

int fgetc (FILE* pt_in);
int fputc (int ch, FILE* pt_out);
char toupper (char ch); (ctype.h)

6.) Escreva um programa que leia duas strings str1 e str2 via interação com o usuário e as concatene usando a função de concatenação de strings. Imprima a string concatenada e grave a string em um arquivo. Lembre-se que a string pode conter caracteres brancos.
char* concat(char* str1, char* str2, int* tam_str3);

7.) Escreva um programa que leia duas strings via interação com o usuário e as concatene usando a função do item anterior. Imprima a string concatenada e grave a string em um arquivo cujo nome é dado via interação como usuário. Lembre-se que a string pode conter caracteres brancos.

8.) Faça um programa que receba do usuário um arquivo texto e um caracter. Mostre na tela quantas vezes aquele caractere ocorre dentro do arquivo.

9.) Faça um programa que receba do usuário um arquivo texto. Crie outro arquivo texto ´ contendo o texto do arquivo de entrada, mas com as vogais substituídas por '*'. Segue abaixo a função que conta o número de vogais em um string a ser adaptada para resolver o problema.
A seguinte função conta o número de vogais em uma string:

```
int contaVogais (char s[]) {
  char *vogais;
  vogais = "aeiouAEIOU";
  int numVogais = 0;
  for (int i = 0; s[i] != '\0'; ++i) {
    char ch = s[i];
    for (int j = 0; vogais[j] != '\0'; ++j) {
      if (vogais[j] == ch) {
        numVogais += 1;
        break;
      }
    }
  }
  return numVogais;
}
```

10.) Faça um programa que receba o nome de um arquivo de entrada (leitura) e outro de saída (gravação). O arquivo de entrada contem em cada linha o nome de uma cidade (ocupando 40 caracteres) e o seu numero de habitantes. O programa deverá ler o arquivo de entrada e gerar um arquivo de saída onde aparece o nome da cidade mais populosa seguida pelo seu numero de habitantes.

11.) Abra um arquivo texto, calcule e escreva o numero de caracteres, o número de linhas e o numero de palavras neste arquivo. Escreva também quantas vezes cada letra ocorre no arquivo (ignorando letras com acento). Obs.: palavras são separadas por um ou mais caracteres espaço, tabulação ou nova linha.

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

int main() {
   //Variáveis de contagem
   int comecouPalavra = 0, numPalavras = 0, numLinhas = 0, numCaracteres = 0;
   //Variável usada como descritor do arquivo em disco
   FILE *descritor;
   //Variável que irá receber a string de onde está o arquivo
   char arquivo[50];
   //Variável que armazena caracter para processamento
   char *caracter;
   printf("Digite o arquivo que deseja abrir: ");
   gets(arquivo);
   descritor = fopen(arquivo, "r");
   while (!feof(descritor)) {
       fread(caracter, 1, 1, descritor);
       numCaracteres++;
       if  ((*caracter!=' ') && (*caracter!='\n') && (!comecouPalavra)) {
         comecouPalavra = 1;
       }
       if  (((*caracter==' ') || (*caracter == '\n')) && (comecouPalavra)) {
         comecouPalavra = 0;
         numPalavras++;
       }
       if (*caracter=='\n') {
         numLinhas++;
         }
    }
  printf("\n O número de palavras do arquivo é: %d", numPalavras);
  printf("\n O número de linhas do arquivo é: %d", numLinhas);
  printf("\n O número de caracteres do arquivo é: %d", numCaracteres);
  getchar();
}
```

12.) Faça um programa que permita que o usuário entre com diversos nomes e telefone para cadastro, e crie um arquivo com essas informações, uma por linha. O usuário finaliza a entrada com ‘0’ para o telefone.

13.) Dado um arquivo contendo um conjunto de nome e data de nascimento (DD MM AAAA,) isto e, 3 inteiros em sequência. Faça um programa que leia o nome do arquivo e a data de hoje e construa outro arquivo contendo o nome e a idade de cada pessoa do primeiro arquivo.

14.) Faça um programa que receba como entrada o ano corrente e o nome de dois arquivos: um de entrada e outro de sa´ıda. Cada linha do arquivo de entrada contem o nome de uma pessoa (ocupando 40 caracteres) e o seu ano de nascimento. O

programa devera´ ler o arquivo de entrada e gerar um arquivo de sa´ıda onde aparece o nome da pessoa seguida por uma string que representa a sua idade.

- Se a idade for menor do que 18 anos, escreva “menor de idade”
- Se a idade for maior do que 18 anos, escreva “maior de idade”
- Se a idade for igual a 18 anos, escreva “entrando na maior idade”

15.) Faça um programa que leia um arquivo que contenha as dimensões de uma matriz (linha e coluna), a quantidade de posições que ser ˜ ao anuladas, e as posições a serem anuladas (linha e coluna). O programa le esse arquivo e, em seguida, produz um novo arquivo com a matriz com as dimensões dadas no arquivo lido, e todas as posições especificadas no arquivo ZERADAS e o restante recebendo o valor 1.

Ex: arquivo “matriz.txt”

3 3 2 /*3 e 3 dimensões da matriz e 2 posições que serão anuladas*/

1 0

1 2 /*Posições da matriz que serão anuladas.

arquivo “matriz saida.txt”

saída:

1 1 1

0 1 0

1 1 1

16.) Crie um programa que receba como entrada o numero de alunos de uma disciplina. Aloque dinamicamente dois vetores para armazenar as informações a respeito desses alunos. O primeiro vetor contem o nome dos alunos e o segundo contêm suas notas finais. Crie um arquivo que armazene, a cada linha, o nome do aluno e sua nota final. Use nomes com no máximo 40 caracteres. Se o nome não contêm 40 caracteres, complete com espaço em branco.

17.) Crie um programa que receba como entrada o numero de alunos de uma disciplina. Aloque dinamicamente em uma estrutura para armazenar as informações a respeito desses alunos: nome do aluno e sua nota final. Use nomes com no máximo 40 caracteres. Em seguida, salve os dados dos alunos em um arquivo binário. Por fim, leia o arquivo e mostre o nome do aluno com a maior nota.

18.) Faça um programa para atualizar contas bancárias. O programa deve abrir quatro arquivos binários: (a) contas dos clientes no dia anterior, (b) movimentações no dia (débitos e créditos), (c) contas criadas no dia (contendo o saldo inicial) e (d) contas removidas no dia. O saldo atual de cada conta deve ser atualizado com base nas movimentações diárias. Um novo arquivo bin ´ ario de contas de clientes deve ser criado, contendo o saldo atualizado de cada cliente (com base nas movimentações diárias) e as novas contas. As contas

removidas no dia não devem aparecer neste novo arquivo. Após criar este arquivo o programa é encerrado. Os vetores utilizados devem ser alocados dinamicamente. Os dados de cada cliente são: número da conta (produzido automaticamente pelo sistema), nome e saldo. Uma movimentação é composta de número da conta, tipo da operação (credito ou débito), valor.

19.) De acordo com o exercício anterior grave em um arquivo bin ´ ario (a) todas as movimentações que produziram saldo negativo, (b) tentativas de remover uma conta que não existe.