Para todas as questões, utilize as funções da norma POSIX (open(), close(), write(), read() e lseek()). Compile os códigos com o gcc e execute-os via terminal.

1. Crie um código em C para escrever "Ola mundo!" em um arquivo chamado 'ola\_mundo.txt'.

#include <stdio.h>

#include <fcntl.h>

#include <unistd.h>

#include <stdlib.h>

int main(int argc,char \*\*argv){

int fp,i;

char \*str = "Ola mundo!";

fp = open("ola\_mundo.txt", O\_WRONLY | O\_CREAT,S\_IRWXU);

if(fp==-1){

printf ("Erro na abertura do arquivo.\n");

exit (1);

}

for(i=0; str[i]; i++)

write(fp, &(str[i]), 1);

write(fp, "\n", 1);

close(fp);

return 0;

}

1. Crie um código em C que pergunta ao usuário seu nome e sua idade, e escreve este conteúdo em um arquivo com o seu nome e extensão '.txt'. Por exemplo, considerando que o código criado recebeu o nome de 'ola\_usuario\_1':

$ ./ola\_usuario\_1

$ Digite o seu nome: Eu

$ Digite a sua idade: 30

$ cat Eu.txt

$ Nome: Eu

$ Idade: 30 anos

#include <stdio.h>

#include <fcntl.h>

#include <unistd.h>

#include <stdlib.h>

#include <string.h>

int main(int argc,char \*\*argv){

int fp,i;

char nome\_arq[30],nome[30], idade[5];

printf("Digite o seu nome: ");

gets(nome);

printf("Digite a sua idade: ");

gets(idade);

strcpy(nome\_arq,nome);

strcat(nome\_arq,".txt");

fp = open(nome\_arq, O\_WRONLY | O\_CREAT,S\_IRWXU);

write(fp,"Nome: ",6);

for(i=0; nome[i]; i++)

write(fp, &(nome[i]), 1);

write(fp,"\n",1);

write(fp,"Idade: ",7);

for(i=0; idade[i]; i++)

write(fp, &(idade[i]), 1);

write(fp," anos\n",6);

close(fp);

return 0;

}

1. Crie um código em C que recebe o nome do usuário e e sua idade como argumentos de entrada (usando as variáveis argc e \*argv[]), e escreve este conteúdo em um arquivo com o seu nome e extensão '.txt'. Por exemplo, considerando que o código criado recebeu o nome de 'ola\_usuario\_2':

$ ./ola\_usuario\_2 Eu 30

$ cat Eu.txt

$ Nome: Eu

$ Idade: 30 anos

#include <stdio.h>

#include <fcntl.h>

#include <unistd.h>

#include <stdlib.h>

#include <string.h>

int main(int argc,char \*\*argv){

int fp,i;

char \*nome\_arq,\*nome,\*idade;

nome = malloc(strlen(argv[1]));

idade = malloc(strlen(argv[2]));

nome = argv[1];

idade = argv[2];

nome\_arq = malloc(strlen(argv[1])+4);

strcpy(nome\_arq,nome);

strcat(nome\_arq,".txt");

fp = open(nome\_arq, O\_WRONLY | O\_CREAT,S\_IRWXU);

write(fp,"Nome: ",6);

for(i=0; nome[i]; i++)

write(fp, &(nome[i]), 1);

write(fp,"\n",1);

write(fp,"Idade: ",7);

for(i=0; idade[i]; i++)

write(fp, &(idade[i]), 1);

write(fp," anos\n",6);

close(fp);

return 0;

}

1. Crie uma função que retorna o tamanho de um arquivo, usando o seguinte protótipo: int tam\_arq\_texto(char \*nome\_arquivo); Salve esta função em um arquivo separado chamado 'bib\_arqs.c'. Salve o protótipo em um arquivo chamado 'bib\_arqs.h'. Compile 'bib\_arqs.c' para gerar o objeto 'bib\_arqs.o'.

int tam\_arq\_texto(char \*nome\_arquivo);

#include <stdio.h>

#include <fcntl.h>

#include <unistd.h>

#include <stdlib.h>

int tam\_arq\_texto(char \*nome\_arquivo){

int fp, size;

fp = open(nome\_arquivo, O\_RDWR | O\_CREAT, S\_IRWXU);

size = lseek(fp, 0, SEEK\_END);

close(fp);

return size;

}

1. Crie uma função que lê o conteúdo de um arquivo-texto e o guarda em uma string, usando o seguinte protótipo: void le\_arq\_texto(char \*nome\_arquivo, char \*conteúdo); Repare que o conteúdo do arquivo é armazenado no vetor conteudo[]. Ou seja, ele é passado por referência. Salve esta função no mesmo arquivo da questão 4, chamado 'bib\_arqs.c'. Salve o protótipo no arquivo 'bib\_arqs.h'. Compile 'bib\_arqs.c' novamente para gerar o objeto 'bib\_arqs.o'.

int tam\_arq\_texto(char \*nome\_arquivo);

void le\_arq\_texto(char \*nome\_arquivo, char \*conteudo);

#include <stdio.h>

#include <fcntl.h>

#include <unistd.h>

#include <stdlib.h>

int tam\_arq\_texto(char \*nome\_arquivo){

int fp, size;

fp = open(nome\_arquivo, O\_RDWR | O\_CREAT, S\_IRWXU);

size = lseek(fp, 0, SEEK\_END);

close(fp);

return size;

}

void le\_arq\_texto(char \*nome\_arquivo, char \*conteudo){

int fp,size;

fp = open(nome\_arquivo, O\_RDWR | O\_CREAT, S\_IRWXU);

size = tam\_arq\_texto(nome\_arquivo);

read(fp,conteudo,size);

close(fp);

}

1. Crie um código em C que copia a funcionalidade básica do comando cat: escrever o conteúdo de um arquivo-texto no terminal. Reaproveite as funções já criadas nas questões anteriores. Por exemplo, considerando que o código criado recebeu o nome de 'cat\_falsificado':

$ echo Ola mundo cruel! Ola universo ingrato! > ola.txt

$ ./cat\_falsificado ola.txt

$ Ola mundo cruel! Ola universo ingrato!

#include <stdio.h>

#include <fcntl.h>

#include <unistd.h>

#include <stdlib.h>

#include "bib\_arqs.h"

int main(int argc,char \*\*argv){

char \*str;

int size;

size = tam\_arq\_texto(argv[1]);

str = malloc(size);

le\_arq\_texto(argv[1],str);

printf("%s",str);

free(str);

return 0;

}

1. Crie um código em C que conta a ocorrência de uma palavra-chave em um arquivo-texto, e escreve o resultado no terminal. Reaproveite as funções já criadas nas questões anteriores. Por exemplo, considerando que o código criado recebeu o nome de 'busca\_e\_conta':

$ echo Ola mundo cruel! Ola universo ingrato! > ola.txt

$ ./busca\_e\_conta Ola ola.txt

$ 'Ola' ocorre 2 vezes no arquivo 'ola.txt'.

#include <stdio.h>

#include <fcntl.h>

#include <unistd.h>

#include <stdlib.h>

#include <string.h>

#include "bib\_arqs.h"

int countOccurrences(char \* str, char \* toSearch);

int main(int argc,char \*\*argv){

char \*str\_file;

int size,count;

size = tam\_arq\_texto(argv[2]);

str\_file = malloc(size+1);

le\_arq\_texto(argv[2],str\_file);

count = countOccurrences(str\_file,argv[1]);

printf("'%s' ocorre %d vezes no arquivo '%s'\n",argv[1],count,argv[2]);

return 0;

}

int countOccurrences(char \* str, char \* toSearch)

{

int i, j, found, count;

int stringLen, searchLen;

stringLen = strlen(str); // length of string

searchLen = strlen(toSearch); // length of word to be searched

count = 0;

for(i=0; i <= stringLen-searchLen; i++)

{

/\* Match word with string \*/

found = 1;

for(j=0; j<searchLen; j++)

{

if(str[i + j] != toSearch[j])

{

found = 0;

break;

}

}

if(found == 1)

{

count++;

}

}

return count;

}