FATEC

Evaldo de Araújo
Lana Eduarda Schimitez
Nathan Coliado Santanta
Rafael Cabral Lee

PROJETO SHELL

Americana – São Paulo
2019

Shell

```
/*
Projeto SHELL - SO
Autor: Soffner
Data: 05/2018
*/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>
#include <sys/wait.h>


int main(int argc, char * argv[])
{
    char comando[30];
    char *arg[3];
    int pid;
    char senha[10];
    printf("Digite a senha: \n");
    gets(senha);
    if(strcmp(senha, "fatecso") != 0)
    {
        printf("Senha incorreta! \n");
        exit(1);
    }
    for( ; ; )
    {
        printf("fatec> ");
        gets(comando);
        argv[0]=strtok(comando, "");
        argv[1]=strtok(NULL, "");
        argv[2]=NULL;
        if(strcmp(argv[0], "sair")==0)
            exit(0);
        if(strcmp(argv[0], "apagar")==0) // apagar
        {
            pid=fork();
            if(pid==0)
            {
                execlp("./apagar", "./apagar", argv[1], NULL);
            }
            else
                wait(NULL);
            continue;
        }
        if(strcmp(argv[0], "data")==0)
        {
            pid=fork();
            if(pid==0)
```

```
    {
        execlp("./data", "./data", argv[1], NULL);
    }
    else
        wait(NULL);
    continue;
}
if(strcmp(argv[0], "mudar")==0)
{
    pid=fork();
    if(pid==0)
    {
        execlp("./mudar", "./mudar", argv[1], NULL);
    }
    else
        wait(NULL);
    continue;
}
if(strcmp(argv[0], "local")==0)
{
    pid=fork();
    if(pid==0)
    {
        execlp("./local", "./local", argv[1], NULL);
    }
    else
        wait(NULL);
    continue;
}
 if(strcmp(argv[0], "listar")==0)
{
    pid=fork();
    if(pid==0)
    {
        execlp("./listar", "./listar", argv[1], NULL);
    }
    else
        wait(NULL);
    continue;
}
 if(strcmp(argv[0], "criar")==0)
{
    pid=fork();
    if(pid==0)
    {
        execlp("./criar", "./criar", argv[1], NULL);
    }
    else
        wait(NULL);
    continue;
}
if(strcmp(argv[0], "remover")==0)
{
    pid=fork();
    if(pid==0)
```

```
    {
        execlp("./remover", "./remover", argv[1], NULL);
    }
    else
        wait(NULL);
    continue;
}
if(strcmp(argv[0], "dicas")==0)
{
    pid=fork();
    if(pid==0)
    {
        execlp("./dicas", "./dicas", argv[1], NULL);
    }
    else
        wait(NULL);
    continue;
}
   if(strcmp(argv[0], "copiar")==0)
{
    pid=fork();
    if(pid==0)
    {
        execlp("./copiar", "./copiar", argv[1], NULL);
    }
    else
        wait(NULL);
    continue;
}
if(strcmp(argv[0], "creditos")==0)
{
    pid=fork();
    if(pid==0)
    {
        execlp("./creditos", "./creditos", argv[1], NULL);
    }
    else
        wait(NULL);
    continue;
}
if(strcmp(argv[0], "calculadora")==0)
{
    pid=fork();
    if(pid==0)
    {
        execlp("./calculadora", "./calculadora", argv[1], NULL);
    }
    else
        wait(NULL);
    continue;
}
else
{
    printf("Comando invalido \n");
}
```

```
    }
}
```

Mudar (cd)

```
#include<stdio.h>
#include<unistd.h>
#include <string.h>
int main()
{
    char s[100],diretorio[30] ;
    printf("%s\n", getcwd(s, 100));
    printf("Digite o nome da pasta: ");
    gets(diretorio);
    strcat(diretorio, "/");
    chdir(diretorio);
    printf("%s\n", getcwd(s, 100));
    return 0;
}
```

local (pwd)

```
#include<stdio.h>
#include<unistd.h>
int main()
{
    char s[100];
    printf("%s\n", getcwd(s, 100));
    return 0;
}
```

listar (ls)

```
#include<stdio.h>
#include <stdlib.h>
#include <sys/unistd.h>
#include <unistd.h>
#include <string.h>
#include <dirent.h>
int main(void)
{
    char diretorio[100];
    DIR *dir;
    struct dirent *dent;
    dir = opendir(getcwd(diretorio,100));

    if(dir!=NULL)
    {
        while((dent=readdir(dir))!=NULL)
        {
```

```
 if((strcmp(dent->d_name,".")==0 || strcmp(dent->d_name,"..")==0 || (*dent->d_name) == '.' ))
            {
            }
            else
            {
                printf(dent->d_name);
                printf("\n");
            }
        }
    }
    close(dir);
}
```

Apagar (rm)

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/unistd.h>
#include <unistd.h>
#include <string.h>
int main(void)
{
char arquivo[30];
printf("Digite o nome do arquivo: \n");
gets(arquivo);
if(unlink(arquivo) == 0)
{
printf("Arquivo apagado com sucesso \n");
}
else
{
printf("Erro ao apagar o arquivo \n");
}
}
```

Criar (mkdir)

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

int main()
{
    char dir[200];
    printf("\ndigite o nome da pasta:  ");
    gets(dir);
    strcat(dir, "/");
    printf("\npasta criada com sucesso\n");
    mkdir(dir, 7000);
    return 0;
}
```

Remover (rmdir)

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/unistd.h>
#include <unistd.h>
#include <string.h>

int main(void)
{
char diretorio[30];
printf("Digite o nome do diretorio: \n");
gets(diretorio);
strcat(diretorio, "/");
if(rmdir(diretorio) == 0)
{
printf("\nDiretorio apagado com sucesso \n");
}
else
{
printf("Erro ao apagar o arquivo \n");
}
}
```

Copiar (cp)

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/unistd.h>
#include <unistd.h>
#include <string.h>

int main()
{
char ch, arquivo_origem[20], destino_arquivo[20];
FILE *origem, *destino;
printf("Nome de arquivo para copiar\n");
gets(arquivo_origem);
origem = fopen(arquivo_origem,"r");
if (origem == NULL)
{
printf("Problema ao copiar o arquivo. Refaça a operação.in");
exit(EXIT_FAILURE);
}
printf("Nome da copia\n");
gets(destino_arquivo);
destino =fopen(destino_arquivo, "w");
if (destino == NULL)
{
fclose(origem);
printf("Problema ao copiar o arquivo. Refaça a operação \n");
exit(EXIT_FAILURE);
}
```

```
while ((ch =fgetc(origem)) != EOF)
fputc(ch, destino);
printf("Arquivo Copiado\n");
fclose(origem);
fclose(destino);
return 0;
}
```

Data

```
#include<stdio.h>
#include<time.h>
int main()
{
   printf ("\nData: %s, horas: %s!\n", __DATE__, __TIME__);
   return 0;
}
```

Calculadora

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

int main()
{
    int num1=0, num2=0, soma=0, subtracao=0, multi=0, div=0, esc=0;
    printf("\nDigite um numero: ");
    scanf("%i", &num1);
    printf("\nDigite outro numero: ");
    scanf("%i", &num2);
    printf("O que voce deseja fazer?");
    printf ("\n1- somar\n2-subtracao\n3-multiplicacao\n4-divisao\n");
    scanf("%i", &esc);
    switch (esc)
{
    case 1:
    soma = num1 + num2;
    printf( "A soma e: %i\n", soma );
    break;
case 2:
    subtracao = num1 - num2;
    printf( "A subtracao e: %i\n", subtracao );
    break;
case 3:
    multi = num1 * num2;
    printf( "O produto e: %i \n", multi );
    break;
case 4:
    div = num1 / num2;
    printf( "A divisão e: %i \n", div );
break;
     default :
    printf ("Valor invalido!\n");
```

```
break;
}
   return 0;
}
```

Créditos

```
#include <stdio.h>
int main()
{
   printf("\nEvaldo de Araujo RA-0040961913040");
   printf("\nLana Eduarda Schimitez RA-0040961913021");
   printf("\nNathan Coliado Santana RA-0040961913009");
   printf("\nRafael Cabral Lee RA-0040961913036\n");
}
```

Dicas

```
#include <stdio.h>
int main()
{
   printf("\nsair- sair do programa");
   printf("\napagar- apagar um arquivo");
   printf("\ndata- mostra a data e hora atuais");
   printf("\nmudar- mudar o diretório de trabalho atual");
   printf("\nlocal- mostrar o diretório de trabalho atual");
   printf("\nlistar- mostra todos os arquivos da pasta");
   printf("\ncriar- cria uma nova pasta");
   printf("\nremover- remover um diretorio");
   printf("\ncopiar- copiar arquivo");
   printf("\ncalculadora- abre um programa para calculos simples");
   printf("\ndicas- exibir todos os comandos");
   printf("\ncreditos- nome e RA dos participantes do projeto\n");

}
```