1º)

#stack.h

typedef struct No{

struct No \*prox;

char carac;

}Tno;

typedef Tno \*pilha;

void push(pilha \*L,char letra);

int pop(pilha \*L);

int top(pilha L);

int isfull(pilha L);

int isempty(pilha L);

void create(pilha \*L);

void destroy(pilha \*L);

#stack.c

#include <stdio.h>

#include <stdlib.h>

#include "stack.h"

void create(pilha \*L){

\*L = NULL;

}

void push(pilha \*L,char letra){// inserir no começo

Tno \*novo,\*aux;

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

novo->carac = letra;

novo->prox = \*L;

\*L = novo;

}

int pop(pilha \*L){

Tno \*aux;

char info;

aux = \*L;

\*L = (\*L)->prox;

info = aux->carac;

free(aux);

return info;

}

int top(pilha L){

return L->carac;

}

int isfull(pilha L){

return 0;

}

int isempty(pilha L){

if(L == NULL)

return 1;

else

return 0;

}

void destroy(pilha \*L){

Tno \*aux;

while(\*L!=NULL){

aux = \*L;

\*L = aux->prox;

free(aux);

}

}

#principal.c

int main(){

pilha Stack;

char frase[30];

printf("digite a frase : ");

gets(frase);

create(&Stack);

verificar(&Stack,frase);

destroy(&Stack);

return 0;

}

void verificar(pilha \*L,char string[]){ // -- Verificar a formação da string --

int cont;

int verificador=0;

int qtd;

char aberto;

qtd = strlen(string);

cont = 0;

while(cont < qtd){

if(string[cont] == '[' || string[cont] == '{' || string[cont] == '('){

if(isfull(\*L)==false)

push(L,string[cont]);

}

else if(string[cont] == ']' || string[cont] == '}' || string[cont] == ')'){

if(isempty(\*L)==true){

verificador = 1;

}

else{

aberto = pop(L);

if((string[cont] == ']' && aberto != '[') ||( string[cont] == '}' && aberto != '{')||( string[cont] == ')' && aberto != '(')){

verificador = 1;

break;

}

}

}

cont++;

}

if(verificador == 1 || isempty(\*L)==false){

printf("String Mal formada!\n");

}

else

printf("String Bem formada!\n");

}

2º)

#stack.h

typedef struct No{

struct No \*prox;

int info;

}Tno;

typedef Tno \*pilha;

void push(pilha \*L,int numero);

int pop(pilha \*L);

int top(pilha L);

int isfull(pilha L);

int isempty(pilha L);

void create(pilha \*L);

void destroy(pilha \*L);

#stack.c

#include <stdio.h>

#include <stdlib.h>

#include "stack.h"

void create(pilha \*L){

\*L = NULL;

}

void push(pilha \*L,int numero){// inserir no começo

Tno \*novo,\*aux;

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

novo->info = numero;

novo->prox = \*L;

\*L = novo;

}

int pop(pilha \*L){

Tno \*aux;

int info;

aux = \*L;

\*L = (\*L)->prox;

info = aux->info;

free(aux);

return info;

}

int top(pilha L){

return L->info;

}

int isfull(pilha L){

return 0;

}

int isempty(pilha L){

if(L == NULL)

return 1;

else

return 0;

}

void destroy(pilha \*L){

Tno \*aux;

while(\*L!=NULL){

aux = \*L;

\*L = aux->prox;

free(aux);

}

}

#principal.c

void verificar(pilha \*L,char string[]);

void converter(pilha \*L,int valor,int base);

int main(){

pilha Stack;

int nun,base;

create(&Stack);

printf("Digite um valor : ");

scanf("%i",&nun);

printf("Digite um valor para a base: ");

scanf("%i",&base);

converter(&Stack,nun,base);

destroy(&Stack);

return 0;

}

void converter(pilha \*L,int valor,int base){

int contador=0;

while(1==1){

if(isfull(\*L) != 1){

push(L,(valor%base));

}

valor = valor/base;

if(valor == 0 )

break;

contador ++;

}

printf("O valor convertido para a base %i sera :",base);

while(contador>=0){

printf("%i",pop(L));

contador--;

}

printf("\n");

}

4°)

#principal.c

#include <stdio.h>

#include "stack.h"

#include "types.h"

void colocar(NoStack \*\* pilha,int numero);

int main(){

NoStack \*pilha;

char resp;

int cont=0;

createStack(&pilha);

do{

colocar(&pilha,cont + 1);

cont++;

printf("Deseja inserar mais algum vagao ? (S | N)");

scanf("%c",&resp);fflush(stdin);

}while(resp != 'n');

return 0;

}

void colocar(NoStack \*\* pilha,int numero){

Stack desvio;

int qtd=1,auxNun;

if(numero == 1){

if(isFull(\*pilha) == FALSE){

push(pilha,numero);

}

}

else{

for(qtd=1;qtd<numero;qtd++){

auxNun = pop(pilha);

push(&desvio,auxNun);

}

push(pilha,numero);// --

for(qtd=1;qtd<numero;qtd++){

auxNun = pop(&desvio);

push(pilha,auxNun);

}

}

}

3º)

#principal.c

#include <stdio.h>

#include <string.h>

#include <ctype.h>

#include "stack.h"

void verificar(char string[]);

int main(){

char frase[100];

printf("digite a frase : ");

gets(frase);

verificar(frase);

return 0;

}

void verificar(char string[]){ // -- Verificar a string --

Tno \* Stack;

int qtd\_carac,contador=0;

char letra;

int meio;

qtd\_carac = strlen(string);

if(qtd\_carac%2==0)

meio = qtd\_carac/2;

else

meio = (qtd\_carac/2);

while(contador < qtd\_carac){

if((string[contador]<65 || string[contador]>90) && (string[contador]<97 || string[contador]>122) && (string[contador]!=32)){

// -- Validando caracter --

printf("Caracter invalido!\n");

break;

}

if(contador<meio && string[contador] != 32){

push(&Stack,string[contador]);

}

else if(contador>meio && string[contador] != 32){

letra = tolower(pop(&Stack));

if(letra != string[contador]){

printf("Frase não palindromo!\n");

break;

}

}

contador++;

}

if(contador = qtd\_carac){

printf("Frase palindromo!\n");

}

else if(isempty(Stack)==false){

destroy(&Stack);

}

}

#stack.c

#include <stdio.h>

#include <stdlib.h>

#include "stack.h"

void create(pilha \*L){

\*L = NULL;

}

void push(pilha \*L,char numero){// inserir no começo

Tno \*novo,\*aux;

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

novo->info = numero;

novo->prox = \*L;

\*L = novo;

}

int pop(pilha \*L){

Tno \*aux;

char info;

aux = \*L;

\*L = (\*L)->prox;

info = aux->info;

free(aux);

return info;

}

int top(pilha L){

return L->info;

}

int isfull(pilha L){

return 0;

}

int isempty(pilha L){

if(L == NULL)

return 1;

else

return 0;

}

void destroy(pilha \*L){

Tno \*aux;

while(\*L!=NULL){

aux = \*L;

\*L = aux->prox;

free(aux);

}

}

#types.h

#ifndef false

#define false 0

#endif // !false

#ifndef true

#define true 1

#endif // !true

#ifndef ESTRUTURAS

#define ESTRUTURAS

typedef struct No{

struct No \*prox;

char info;

}Tno;

typedef Tno \*pilha;

#endif

#Stack.h

#include "types.h"

void push(pilha \*L,char numero);

int pop(pilha \*L);

int top(pilha L);

int isfull(pilha L);

int isempty(pilha L);

void create(pilha \*L);

void destroy(pilha \*L);