Resolução - Lista Estrutura de Dados

1 – Temos o histórico de navegação de um navegador, que vai empilhando os resultados que você pesquisou. Os últimos procurados são os primeiros que aparecem, e os mais antigos ficam lá em baixo. Também temos o CTRL + Z, que desfaz a última alteração que você realizou em algum programa, ou seja, o último que entra é o primeiro que sai.

2 – O tipo de dado do ponteiro deve ser do tipo STRUCT, e será relativo ao “próximo nó” da sequência da pilha. A quantidade de memória que será consumida depende do tamanho da struct do nó.

3 – Push (Empilhar) e Pop (Desempilhar).

4 – Significa que a pilha está vazia.

5 –

6 -   
  
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct apelido\_no{

char nome[30];

struct apelido\_no \*proximo;

}No;

void push(char \*item);

void pop();

void imprimir();

No \*topo = NULL;

int main(){

int opcao;

do {

system("cls");

printf("PILHA 1 - Por JeeJ\n");

printf("1 - Empilhar\n");

printf("2 - Desempilhar\n");

printf("3 - Mostrar a pilha\n\n");

printf("Opcao: ");

scanf("%d", &opcao);

switch(opcao){

case 1:{

system("cls");

fflush(stdin);

char n[30];

printf("Escreva um nome: ");

scanf("%29[^\n]", n);

push(n);

}

break;

case 2:

pop();

break;

case 3:

imprimir();

break;

}

} while (opcao != 0);

return 0;

}

void push(char \*item){

No \*novo = malloc(sizeof(No));

strcpy((\*novo).nome, item);

(\*novo).proximo = topo;

topo = novo;

printf("Valor empilhado!\n");

system("pause");

}

void pop(){

system("cls");

if (topo == NULL){

printf("A pilha esta vazia!\n");

} else {

No \*temp = topo;

topo = (\*topo).proximo;

free(temp);

printf("Valor desempilhado!\n");

}

system("pause");

}

void imprimir(){

system("cls");

No \*temp = topo;

while (temp != NULL){

printf("%s\n", (\*temp).nome);

temp = (\*temp).proximo;

}

system("pause");

}

7 -   
  
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct apelido\_no{

char nome[30];

struct apelido\_no \*proximo;

}No;

void push(char \*item);

void pop();

void imprimir();

int mostrarQntd();

No \*topo = NULL;

int main(){

int opcao;

do {

system("cls");

printf("PILHA 1 - Por JeeJ\n");

printf("1 - Empilhar\n");

printf("2 - Desempilhar\n");

printf("3 - Mostrar a pilha\n");

printf("4 - Mostrar quantidade de elementos\n\n");

printf("Opcao: ");

scanf("%d", &opcao);

switch(opcao){

case 1:{

system("cls");

fflush(stdin);

char n[30];

printf("Escreva um nome: ");

scanf("%29[^\n]", n);

push(n);

}

break;

case 2:

pop();

break;

case 3:

imprimir();

break;

case 4:

system("cls");

printf("Quantidade atual de elementos: %d\n", mostrarQntd());

system("pause");

break;

}

} while (opcao != 0);

return 0;

}

void push(char \*item){

No \*novo = malloc(sizeof(No));

strcpy((\*novo).nome, item);

(\*novo).proximo = topo;

topo = novo;

printf("Valor empilhado!\n");

system("pause");

}

void pop(){

system("cls");

if (topo == NULL){

printf("A pilha esta vazia!\n");

} else {

No \*temp = topo;

topo = (\*topo).proximo;

free(temp);

printf("Valor desempilhado!\n");

}

system("pause");

}

void imprimir(){

system("cls");

No \*temp = topo;

while (temp != NULL){

printf("%s\n", (\*temp).nome);

temp = (\*temp).proximo;

}

system("pause");

}

int mostrarQntd(){

int qntd = 0;

No \*temp = topo;

while (temp != NULL){

qntd++;

temp = (\*temp).proximo;

}

return qntd;

}

8 -   
  
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct apelido\_no{

char \*nome;

struct apelido\_no \*proximo;

}No;

void push(char \*item);

void pop();

void imprimir();

int mostrarQntd();

No \*topo = NULL;

int main(){

int opcao;

do {

system("cls");

printf("PILHA 1 - Por JeeJ\n");

printf("1 - Empilhar\n");

printf("2 - Desempilhar\n");

printf("3 - Mostrar a pilha\n");

printf("4 - Mostrar quantidade de elementos\n\n");

printf("Opcao: ");

scanf("%d", &opcao);

switch(opcao){

case 1:{

system("cls");

fflush(stdin);

char n[30];

printf("Escreva um nome: ");

scanf("%29[^\n]", n);

push(n);

}

break;

case 2:

pop();

break;

case 3:

imprimir();

break;

case 4:

system("cls");

printf("Quantidade atual de elementos: %d\n", mostrarQntd());

system("pause");

break;

}

} while (opcao != 0);

return 0;

}

void push(char \*item){

No \*novo = malloc(sizeof(No));

int tamanho = strlen(item) + 1;

(\*novo).nome = malloc(tamanho);

strcpy((\*novo).nome, item);

(\*novo).proximo = topo;

topo = novo;

printf("Valor empilhado!\n");

system("pause");

}

void pop(){

system("cls");

if (topo == NULL){

printf("A pilha esta vazia!\n");

} else {

No \*temp = topo;

topo = (\*topo).proximo;

free(temp);

printf("Valor desempilhado!\n");

}

system("pause");

}

void imprimir(){

system("cls");

No \*temp = topo;

while (temp != NULL){

printf("%s\n", (\*temp).nome);

temp = (\*temp).proximo;

}

system("pause");

}

int mostrarQntd(){

int qntd = 0;

No \*temp = topo;

while (temp != NULL){

qntd++;

temp = (\*temp).proximo;

}

return qntd;

}