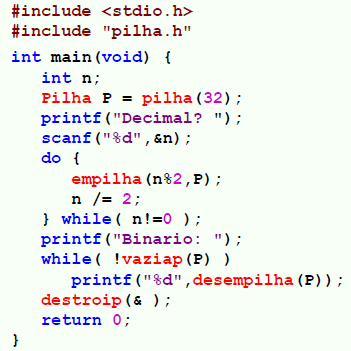
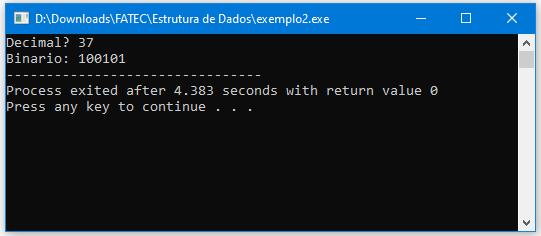
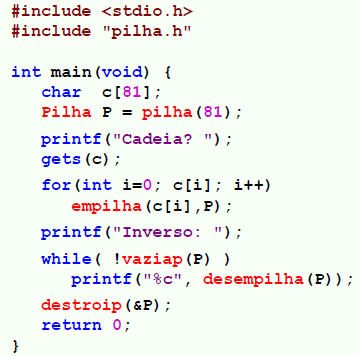
ed-02v2 – 12/08/2021

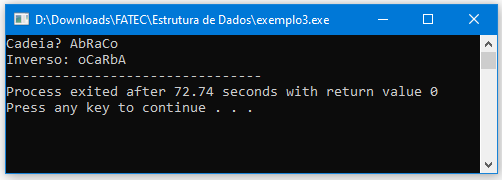
Exemplo 2.

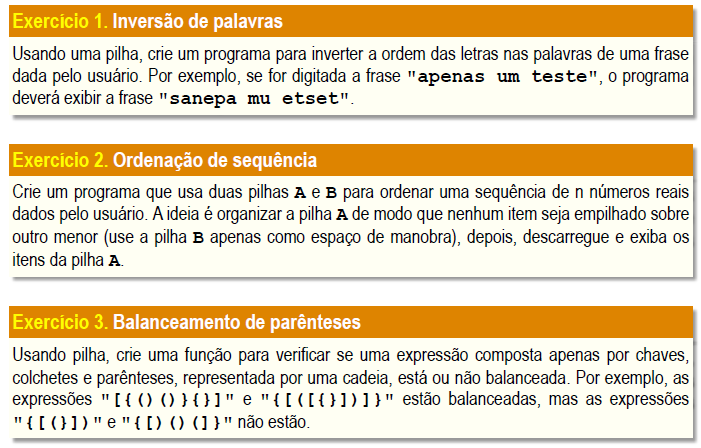


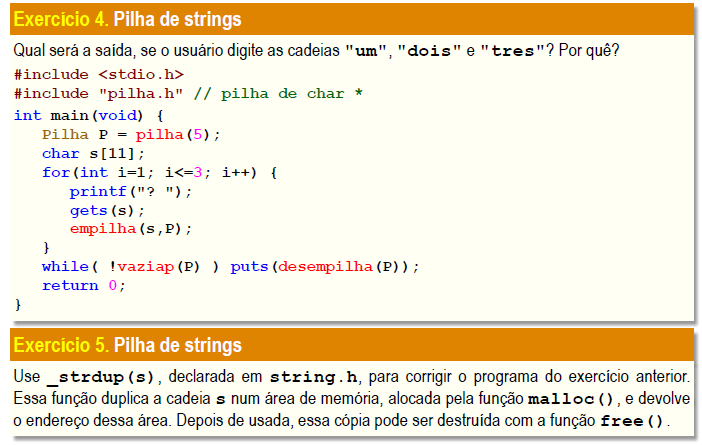


Exemplo 3.









**Exercício 01.**

#include <stdio.h>

#include <string.h>

#include "pilhaChar.h"

int main(void){

char c[70], rev[70];

int i, len, index, palComeco, palFim;

Pilha P = pilha(70);

printf("Frase desejada: ");

gets(c);

len = strlen(c);

index = 0;

palComeco = len - 1;

palFim = len - 1;

while(palComeco > 0){

if(c[palComeco] == ' '){

i = palComeco + 1;

while(i <= palFim){

rev[index] = c[i];

i++;

index++;

}

rev[index++] = ' ';

palFim = palComeco - 1;

}

palComeco--;

}

for(i=0; i<=palFim; i++){

rev[index] = c[i];

index++;

}

rev[index] = '\0';

printf("%s\n", rev);

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

empilha(rev[i], P);

printf("Inverso: ");

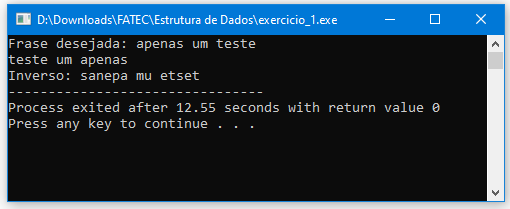
while(!vaziap(P))

printf("%c", desempilha(P));

destroip(&P);

return 0;

}



**Exercício 02.**

#include <stdio.h>

#include <stdlib.h>

#include "pilha.h"

#define TAM 20

float aux, n;

char ask='s';

int main(void){

Pilha A = pilha(TAM);

Pilha B = pilha(TAM);

do{

printf("\nEntrar com um numero: ");

scanf("%d", &n);

empilha(n, A);

printf("Deseja continuar? [s/n]... ");

scanf("%s", &ask);

}while(ask=='s');

while (!vaziap(A)){

aux = desempilha(A);

while (!vaziap(B) && topo(B) > aux){

empilha(desempilha(B), A);

}

empilha(aux, B);

}

while (!vaziap(B)){

empilha(desempilha(B), A);

}

printf("Pilha: \n");

while(!vaziap(A))

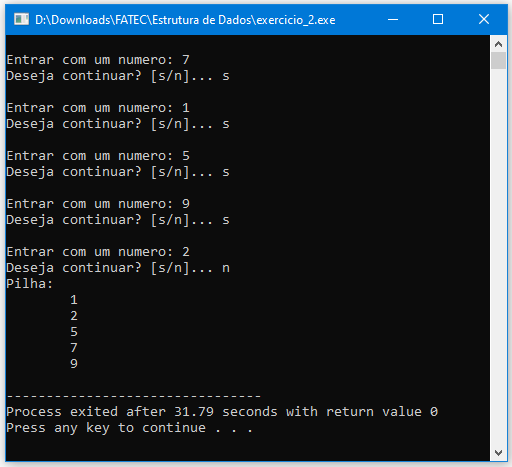
printf("\t%d\n", desempilha(A));

destroip(&A);

destroip(&B);

return 0;

}



**Exercício 03.**

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#include "pilhaChar.h"

#define VAL 20

bool correspondencia(char caractere1, char caractere2){

if(caractere1 =='(' && caractere2 == ')')

return 1;

else if(caractere1 =='[' && caractere2 == ']')

return 1;

else if(caractere1 =='{' && caractere2 == '}')

return 1;

else

return 0;

}

bool caractereBalanceado(char exp[]){

int i = 0;

Pilha T = pilha(VAL);

while(exp[i]){

if (exp[i] == '{' || exp[i] == '(' || exp[i] == '[')

empilha(exp[i], T);

if (exp[i] == '}' || exp[i] == ')' || exp[i] == ']') {

if (vaziap(T))

return 0;

else if (!correspondencia(desempilha(T), exp[i]))

return 0;

}

i++;

}

if (vaziap(T))

return 1; //balanceado

else

return 0; //não balanceado

}

int main(void){

char exp[VAL] = "{()}[";

if(caractereBalanceado(exp))

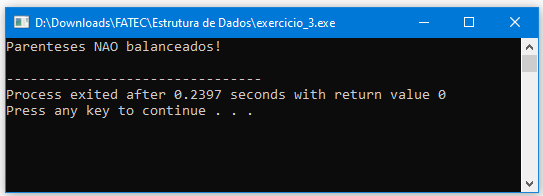
puts("Parenteses balanceados!");

else

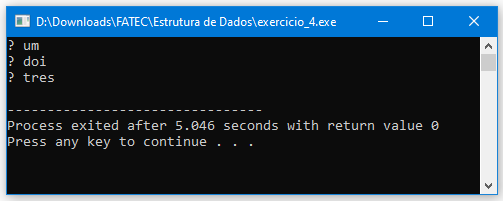
puts("Parenteses NAO balanceados!");

return 0;

}



**Exercício 04.**



Ocorre problemas ao empilhar cadeia de char em determinada Pilha, deste modo não ocorre o empilhamento das entradas.

**Exercício 05.**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "pilhaChar.h"

int i;

int main(void){

system("cls");

Pilha P = pilha(5);

char s[11];

printf("\nEmpilhando e Copiando cadeia de char\n");

for(i=1; i<=3; i++){

printf("? ");

gets(s);

empilha(\_strdup(s), P);

}

while(!vaziap(P)) puts(desempilha(P));

free(\_strdup(s));

printf("\n");

return 0;

}