Ex: 001

/\*

O programa do Exemplo 2 não reconhece "Amor a Roma" como uma cadeia palíndroma. Use a função toupper(), declarada em ctype.h, para resolver esse problema (essa função converte uma letra minúscula em maiúscula).

\*/

#include <stdio.h>

#include <ctype.h>

#include "pilha.h"

#include "fila.h"

int main(void) {

   char s[256];

   Fila F = fila(256);

   Pilha P = pilha(256);

   printf("\nFrase? ");

   gets(s);

   for(int i=0; s[i]; i++)

      if( isalpha(s[i]) ) {

        s[i]=toupper(s[i]);

        enfileira(s[i],F);

        empilha(s[i],P);

      }

   while( !vaziaf(F) && desenfileira(F)==desempilha(P) );

   if( vaziaf(F) ) puts("A frase e palindroma");

   else puts("A frase nao e palindroma");

   destroif(&F);

   destroip(&P);

   return 0;

}



Ex: 002

/\*

Qual a saída exibida pelo programa a seguir?

\*/

#include <stdio.h>

#include "fila.h"

int main(void) {

   Fila F = fila(5);

   for(int i=0; i<=3; i++)

      enfileira('A'+i,F);

   while( !vaziaf(F) )

      printf("%c\n",desenfileira(F));

   destroif(&F);

   return 0;

}

// A B C D



Ex:003

/\*

Esse programa simula o compartilhamento de uma CPU entre vários processos. Qual a saída?

\*/

#include <stdio.h>

#include "fila.h"

#define TIMESLICE 3 // tempo máximo de uso ininterrupto de CPU

int main(void) {

   Fila F = fila(5);

   enfileira(17,F); // 1º processo requer 7s para concluir sua execução

   enfileira(25,F);

   enfileira(39,F);

   enfileira(46,F);

   while( !vaziaf(F) ) {

      int x = desenfileira(F);

      int p = x/10;

      int t = x%10;

      if( t>3 ) enfileira(p\*10+(t-TIMESLICE),F);

      else printf("Processo %d concluido\n",p);

   }

   destroif(&F);

   return 0;

}

/\*

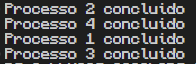
Processo 2 concluido

Processo 4 concluido

Processo 1 concluido

Processo 3 concluido

\*/



Ex: 004

/\*

   Considere n processos a serem executados.

   Uma fila F contendo a ordem de chamada dos processos

   E uma pilha P contendo a ordem de execução ideal dos processos (se um processo i está sobre outro processo j em P, então i deve ser executado antes de j).

   Supondo que cada mudança de ordem dos processos na fila, bem como a execução de um processo, consome 1 segundo, crie a função ttm(F,P) que devolve o tempo total mínimo necessário para executar todos os processos na ordem ideal.

      Por exemplo, se                  F = [2,3,1]

      e a ordem de execução ideal em   P = [1,3,2],

      então o ttm=6 segundos.

      Note que a função ttm() deve funcionar para qualquer quantidade de processos e, no final da execução da função, F e P devem estar vazias.

      2==1? n

      t=1   F=[3,1,2]

      1==3? n

      t=2   F=[1,2,3]

      1==1? s

      t=3   F=[2,3]

      3==2? n

      t=4   F=[3,2]

      3==3? s

      t=5   F=[2]

      2==2? s

      t=6   F=[]

\*/

#include <stdio.h>

#include "pilha.h"

#include "fila.h"

int ttm(Fila F, Pilha P) { // tempo total mínimo

   int t = 0; // tempo total

   while(!vaziap(P) && !vaziaf(F)){

      int x = topo(P);           //1

      int y = desenfileira(F);   //3

      printf("x=%d y=%d\n",x,y); //1 3

      if(x == y){                //1==3? n

         desempilha(P);

      }

      else{

         enfileira(y,F);         //F=[1,2,3]

      }

      t++;                       //t=1

   }

   return t;

}

int main(void) {

 Fila F = fila(3);

 enfileira(2,F);

 enfileira(3,F);

 enfileira(1,F);

 Pilha P = pilha(3);

 empilha(2,P);

 empilha(3,P);

 empilha(1,P);

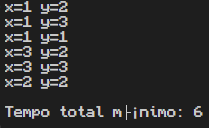
 printf("\nTempo total mínimo: %d\n\n", ttm(F,P) );

 destroif(&F);

 destroip(&P);

 return 0;

}



Ex: 005

/\*

  2. Uma cadeia c composta apenas por letras A e B é uma concha SE cada letra em c pode ser conectada a exatamente uma única outra letra igual a ela em c, por meio de um arco, de modo que nenhum arco intercepte outro arco. Por exemplo, na figura a seguir, a cadeia em (a) é uma concha, mas aquela em (b) não o é

  Com base nesta definição:

   Crie a função concha(c), que recebe uma cadeia c e devolve 1 se c for uma concha (caso contrário, devolve 0).

   Crie a função filtra(F), que recebe uma fila F contendo cadeias compostas apenas por letras A e B e exibe apenas aquelas cadeias em F que são conchas, bem como o total de cadeias em F que são conchas.

\*/

//AAAAAAAA AA ABA

/\*i=10

F=[];

P=[B];

\*/

#include <stdio.h>

#include <string.h>

#include "pilha.h"

#include "fila.h"

int concha(char \*c)

{ // verifica se c é uma concha

  Pilha P = pilha(strlen(c));

  // printf("c= %s\n", c);

  for(int i=0; i<strlen(c); i++){

    if(c[i]!=c[i+1]){

        if(vaziap(P)){

          empilha(c[i],P);

        }

        else if(topo(P)==c[i] || topo(P)==c[i+1]){

          desempilha(P);

        }

        else{

          empilha(c[i],P);

        }

    }

    else{

      i++;

    }

  }

  if(vaziap(P)){

    return 1;

  }

  else{

    return 0;

  }

}

int filtra(Fila F)

{ // exibe apenas as cadeias em F que são conchas

  int total = 0;

  for (int i = 0; i < 20; i++)

  {

    char \*c = desenfileira(F);

    if (concha(c))

    {

      printf("%s e uma concha\n", c);

      total++;

    }

  }

  return total;

}

int main()

{

  Fila F = fila(20);

  enfileira("AAAAAAAABAAABA", F);//nao concha

  enfileira("BBBBBBBBBBBBBB", F);//sim concha

  enfileira("ABBABBABBABB", F);  //sim concha

  enfileira("AAABAAABBABAAA", F);//nao concha

  enfileira("ABAABAAAAAAAAA", F);//sim concha

  enfileira("AABAAAAAABAAAA", F);//sim concha

  enfileira("AAAABB", F);        //sim concha

  enfileira("AABBBBAABAAB", F);  //sim concha

  enfileira("ABAAAAAAAAAB", F);  //nao concha

  enfileira("ABABBBABAB", F);    //nao concha

  enfileira("BBBBBBBBBBBBBB", F);//sim concha

  enfileira("AABAABAAAA", F);    //sim concha

  enfileira("AABAAABBBBBABB", F);//nao concha

  enfileira("BBBAAAAABA", F);    //nao concha

  enfileira("BBBBBBBBBBBBBB", F);//sim concha

  enfileira("AAAAAAAAAAAABB", F);//sim concha

  enfileira("BAAAAABAAAAAAA", F);//nao concha

  enfileira("BBBBABBBBBBA", F);  //sim concha

  enfileira("BBABABAAAAAABB", F);//nao concha

  enfileira("ABAABAABAB", F);    //nao concha

  printf("Total de sim conchas: %d", filtra(F)); // deve exibir 12 conchas

  destroif(&F);

  return 0;

}

