UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação - ICMC
SCC0202 – Algoritmos e Estrutura de Dados I - 2º Sem /2018
Prof. Robson Cordeiro
PAE: Guilherme Queiroz, Leonardo Moraes

Lista de Exercícios: TAD & Pilha

1. O que é um Tipo Abstrato de Dados (TAD) e qual a característica fundamental na sua utilização?
2. Quais as vantagens de se programar com TADs?
3. Considere um programa que simule uma calculadora e realize operações de soma, subtração, divisão e multiplicação de números reais. Defina textualmente as operações do TAD para este programa apresentando as condições de entrada e de saída de dados para que as operações possam ser realizadas.
4. Sejam os TADs **Ponto** e **Circulo** como definido abaixo. Desenvolva um programa cliente (main.c) que, pela ordem: crie um ponto **p** e um círculo **r** definidos pelo usuário (stdin); chame a função *distancia(PONTO *p, CIRCULO *r);* para calcular se **p** é interior ou não a **r**; imprima o resultado. Desenvolva a função *distancia* justificando onde ela deve ser implementada (em que arquivo .c?). Se necessário, desenvolva (implemente) as alterações que julgar pertinentes nos TADs, **sempre justificando**. Um ponto **p** é interior a um círculo **r** se a distância entre **p** e o ponto que define o círculo (ponto_c) é menor que o raio de **r** (raio). A distância entre dois pontos é dada pela equação (1):

$$d = \sqrt{(x2 - x1)^2 + (y2 - y1)^2}$$

```
(ponto.h)
#ifndef PONTO_H
  #define PONTO_H
  typedef struct ponto_ PONTO;
  PONTO *ponto_criar (float x, float y);
  void ponto_apagar (PONTO *p);
...
#endif

(ponto.c)
...
#include "ponto.h"
  struct ponto_{
  float x;
  float y;
};
...
```

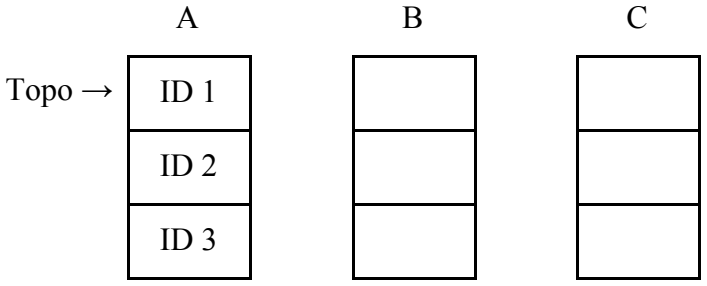
```
(circulo.h)
#ifndef CIRCULO_H
  #define CIRCULO_H
  #include "ponto.h"
  typedef struct circulo_ CIRCULO;
  CIRCULO *circulo_criar (float x, float y,
float raio);
  void circulo_apagar (CIRCULO *circulo);
  float circulo_area (CIRCULO *circulo);
...
#endif

(circulo.c)
...
#include "circulo.h"
#define PI 3.14159
struct circulo_{
  PONTO *ponto_c;
  float raio;
};
...
```

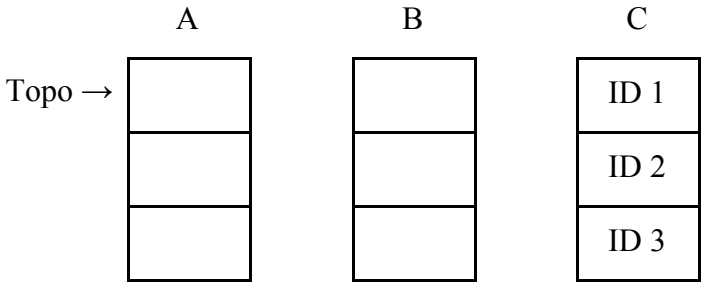
5. Um determinado estacionamento de carros tem o formato de corredor e possui capacidade para dez carros. Infelizmente, há apenas uma passagem de acesso aos carros, localizada em

uma das extremidades do corredor. Quando um cliente decide retirar um carro, os outros carros impedindo sua saída deverão são retirados antes. Após o cliente retirar seu carro, os outros carros são estacionados novamente, mantendo a ordem em que estavam antes da remoção daquele carro. Escreva o TAD e implemente as operações necessárias para processar o fluxo de chegada/saída deste estacionamento (`int estaciona_carro(char* placa)` / `int retira_carro(char* placa)`), considerando que um carro é identificado pela placa. O valor de retorno das funções `estaciona_carro` e `retira_carro` indicam se a operação teve sucesso (valor 1) ou falhou (valor 0).

6. Considere três pilhas A, B e C, e os comandos `void empilha(Pilha *p, Elemento e)` e `Elemento *desempilha(Pilha *p)` que, como os nomes sugerem, servem para empilhar e desempilhar elementos de uma determinada pilha. Supondo que as pilhas estão inicialmente na seguinte configuração:



e que, em nenhum momento um elemento com ID maior pode estar empilhado sobre um elemento de ID menor. Descreva a sequência de comandos (utilize a menor quantidade de operações possível) `empilha/desempilha` necessários para que a configuração de A, B e C passe a ser:



7. Escreva um algoritmo para determinar se uma string de caracteres de entrada é da forma $a^n b^n$ tal que n é um número inteiro maior ou igual a zero. Exemplo de strings na forma $a^n b^n$: ab, aabb, aaabbb. Não é permitido realizar qualquer tipo de contagem de caracteres da string de entrada.

8. Implemente uma calculadora polonesa usando pilha com alocação estática. O programa deve ler um arquivo com uma sequência de expressões em notação polonesa, calcular e imprimir os resultados (na saída padrão).
Notação Polonesa é uma forma de notação para lógica, aritmética e álgebra em que os operadores devem preceder os valores, a notação polonesa de soma é: + 1 2.

Quanto mais "interna" a operação, antes ela deve ser executada. Então, as expressões: * + 1 2 3 seria calculado como (1 + 2) * 3; * 3 + 1 2 seria calculado como 3 * (1 + 2)
A entrada deve ser composta apenas por números inteiros positivos de 0 a 9 e dos seguintes operadores: +, -, / e *.

EXTRAS:

9. Escreva um algoritmo que use uma pilha para inverter a ordem das letras de cada palavra de uma string, preservando a ordem das palavras. Por exemplo, dado o texto ESTE EXERCICIO E MUITO FACIL a saída deve ser ETSE OICICREXE E OTIUM LICAF.

10. Na seguinte sequência, uma letra significa ‘push(letra)’ e um asterisco significa ‘pop’. Dê a sequência de valores devolvidos pelas operações pop quando esta sequência de operações é realizada em uma pilha inicialmente vazia.

E A S * Y * Q U E * * * S T * * * I O * N * * *

11. Escreva o TAD de uma estrutura de dados chamada Pilha Dupla. Esta estrutura de dados deve oferecer a funcionalidade de duas pilhas distintas A e B (alocação sequencial) que devem ser armazenadas em um único vetor compartilhado de tamanho M. Implemente as operações de empilha/desempilha do TAD Pilha Dupla de tal forma que a soma das quantidades de elementos em A e B não extrapole M e que, tanto A quanto B possam armazenar até M elementos.