ED – 2019s2 – Lista 1

Prof. Alexandre - INE/CTC/UFSC

Prazo	**06set2019**
Entrega	*upload* no Moodle dos fontes (`.h` e `.cpp`) necessários à compilação (colocar o nome da dupla nos arquivos)

1. Implemente, em C ou C++, um tipo abstrato de dados (TAD) para o conjunto de números racionais. As operações requeridas são:

   (a) Alocação de um conjunto (vetor) de `N` números racionais. Protótipo sugerido:

```
Racional *criaConjRac(int N);
```

   (b) Geração de uma sequência de `N` racionais seguindo esta regra:

   - Primeiro elemento: $a_1 = \frac{1}{2}$
   - Segundo elemento: $a_2 = m\ a_1 + s = m\ \frac{1}{2} + s$
   - k-ésimo elemento: $a_k = m\ a_{k-1} + s$

   Protótipo sugerido:

```
void seqRac(Racional *conj, int m, int s, int N);
```

   (c) Soma de uma sequência. Protótipo sugerido:

```
Racional *somaTodosRac(Racional *seq, int N);
```

   Material

   - Implementação do TAD Racional:
     - https://www.inf.ufsc.br/~alexandre.goncalves.silva/courses/code/racional.zip

2. Implemente o mapeamento de uma operação devolvendo um vetor de inteiros (alocado dinamicamente) da aplicação de uma função `oper` a cada um dos `N` elementos do vetor `vet` de inteiros da entrada. Segue protótipo:

```
int *mapeamento( int *vet, int N, int (*oper)(int x) );
```

   Exemplo de aplicação e execução:

```
#include <iostream>
#define TAMANHO 10

int quadrado(int x) {
    return x*x;
}

int main() {
    int vetin[TAMANHO] = {1,2,3,4,5,6,7,8,9,10};

    int *vetout = mapeamento(vetin, TAMANHO, quadrado);

    for (int i=0; i < TAMANHO; i++) {
        std::cout << vetout[i] << " ";
    }
    std::cout << std::endl;
    delete [] vetout;
}
```

```
$ ./a.out
1 4 9 16 25 36 49 64 81 100
```

3. Escreva uma função que calcule o resultado final de uma expressão matemática em notação pós-fixa (polonesa). Considera-se cada operando e cada operador separados entre si por um espaço em branco em uma string. Exemplo:

   `char expr[500] = "2.0 5.0 3.0 - 1.0 3.0 + / 1.0 - *";` $\quad /* = 2.0 \left(\frac{5.0-3.0}{1.0+3.0} - 1.0 \right) = -1.0/*$

   Restrições:

   - Os operandos devem ser reais do tipo `float`;
   - Apenas quatro operadores básicos podem ser utilizados (+, -, *, /).

   Dicas:

   - Ler um string por vez (usando, por exemplo, o `sscanf` sobre a expressão) e verificar se o primeiro caracter refere-se ou não a um operador.

   Protótipo:

```
float calculaExpressaoPolonesa(char *expr);
```

4. Duas pilhas A e B podem compartilhar o mesmo vetor, como esquematizado na figura a seguir. Modifique sua implementação de `ArrayStack` (primeiro exercício de VPL no Moodle), de modo a atender o seguinte itens: (i) construtor deve iniciar os valores de `topoA` e `topoB`; (ii) existência de métodos `emptyA` e `emptyB`; (iii) métodos `pushA`, `pushB`, `popA` e `popB`; (iv) uma exceção de pilha cheia, em `full`, só é emitida se todas as posições do vetor estiverem ocupadas.