Armazenamento de dados em memória (revisão de ponteiros e alocação dinâmica), Listas encadeadas, duplamente encadeadas e árvores binárias, e TADs Específicos <STL> (set, map, dictionary, list)

Lab02-00 Revisão Ponteiros – Correção baseada no arquivo main.cpp: Para cada um dos itens seguintes, escreva uma única instrução que realiza a tarefa indicada. Suponha que as variáveis do tipo inteiro `long value1` e `value2` tenham sido declaradas e que `value1` tenha sido inicializado como 200000. As perguntas abaixo não são avaliadas pela saída das impressões. As que não tem saída, não serão avaliadas mas podem ser feitas como revisão.

1. Declare a variável `long_ptr` como um ponteiro para um objeto do tipo `long`.
2. Atribua o endereço da variável `value1` à variável ponteiro `long_ptr`.
3. Imprima o valor do objeto apontado por `long_ptr`.
4. Atribua o valor do objeto apontado por `long_ptr` à variável `value2`.
5. Imprima o valor de `value2`.
6. Imprima o endereço de `value1`.
7. Imprima o endereço armazenado em `long_ptr` . O valor impresso é o mesmo que o endereço de `value1`?
8. Declare um array do tipo `unsigned int` chamado `values` com cinco elementos e inicialize os elementos para os inteiros pares de 2 a 10. Suponha que a constante simbólica `SIZE` foi definida como 5.
9. Declare um ponteiro `v_ptr` que aponta para um objeto do tipo `unsigned int`.
10. Utilize uma instrução `for` para imprimir os elementos do array `values` usando notação de array subscrito.
11. Escreva duas instruções separadas que atribuem o endereço inicial do array values à variável ponteiro `v_ptr`.
12. Utilize uma instrução `for` para imprimir os elementos do array `values` utilizando a notação de ponteiro/deslocamento.
13. Utilize uma instrução `for` para imprimir os elementos do array `values` utilizando a notação de ponteiro/deslocamento com o nome de array como o ponteiro.
14. Utilize uma instrução for para imprimir os elementos do array `values` utilizando subscritos no ponteiro para o array.
15. Referencie o quinto elemento de `values` utilizando a notação de subscrito de array, a notação de ponteiro/deslocamento com o nome de array como o ponteiro, a notação de subscrito de ponteiro e a notação de ponteiro/deslocamento.
16. Mostre que endereço é referenciado por `v_ptr + 3`, e que valor é armazenado nessa localização.
17. Supondo que `v_ptr` aponte para `values[4]`, que endereço é referenciado por `v_ptr -= 4`? Que valor é armazenado nessa localização?

Todas as outras perguntas, correção por doctest estilo Lab01.

1. Implemente um TAD de lista duplamente encadeada. Para o mesmo, implemente as seguintes operações:
   (a) `insere_elemento(int)`. Insere um elemento no fim da lista.
   (b) `insere_primeiro(int)`. Insere um elemento no inicio da lista.
   (c) `get_iesimo(int)`. Retorna um elemento na posição i.
   (d) `remover_elemento()`. Remove um elemento no fim da lista.
   (e) `remover_primeiro()`. Remove o primeiro elemento da lista.
   (f) `inserir_iesimo(int, int)`. Insere um elemento na posição i.
   (g) `remover_iesimo(int)`. Remove um elemento na posição i.
2. Usando o TAD acima, implemente um código para remover elementos consecutivos com um mesmo valor. Isto é, a lista `1<->1<->2<->2<->1<->3<->3<->1` vira `1<->2<->1<->3<->1`.
3. Implemente um código para encontrar o k-ésimo elemento de uma lista encadeada. Sua função deve operar nos dois sentidos. A assinatura é `k_esimo(bool direcao, int k)`. Assuma que `true` é do inicio para o fim, `false` é o contrário.
4. Implemente uma função para checar se uma lista duplamente encadeada é um palíndromo. Tal lista tem a seguinte assinatura `bool checa_palindromo()`.
5. Você está implementando uma classe de árvore binária que, além dos métodos `insere_elemento`, `existe_elemento` e `numero_nos` tem três métodos de assinatura `ListaEncadeada nome(); /* o nome de cada um está abaixo */` que copia todos os elementos da árvore para uma lista. Para copiar elementos, você três possibilidades:
   (a) Inserir o nó atual e depois caminhar para a esquerda e direita (pre_ordem);
   (b) Inserir toda a sub-árvore da esquerda, depois o nó atual, seguido da árvore da direita (em_ordem);
   (c) Inserir toda a sub-árvore da esquerda, depois a árvore da direita, seguido por fim do nó atual (pos_ordem);

Referências

[1] D.J. Barnes and M. Kölling. *Programação orientada a objetos com java: uma introdução prática usando Blue J.* PRENTICE HALL BRASIL, 2004.

[2] P.J. Deitel and H.M. Deitel. *C++ how to Program.* Deitel series. Pearson Prentice Hall, 2010.

[3] G.L. McDowell. *Cracking the Coding Interview: 189 Programming Questions and Solutions.* CareerCup, LLC, 2015.