Lista de Exercícios 3

Universidade Federal de Santa Catarina
Campus de Joinville
Centro Tecnológico de Joinville
EMB5631 – Programação III
Prof. Valdir Pedrinho de Tomin Junior, Eng.

27 de agosto de 2018

Exercícios do Livro

Resolver os seguintes problemas do capítulo 10 do livro (Deitel, H. M.; Deitel, P. J., **C++ Como Programar**, 5ª ed., Pearson, 2015.):

- 10.4 ao 10.10

Dicas:

1. Não é necessário resolver todos os exercícios, mas certifique-se de resolver um número suficiente de exercícios, de cada assunto, para consolidar seus conhecimentos;
2. Se for o caso, procure mais exercícios de determinado assunto; ou
3. Peça ajuda ao professor.

Observação:

As páginas do livro com os problemas propostos estão anexadas.

```
      // corpo vazio
   } // fim do construtor Example

   int getIncrementedData() const
   {
      return data++;
   } // fim da função getIncrementedData
   static int getCount()
   {
      cout << "Data is " << data << endl;
      return count;
   } // fim da função getCount
private:
   int data;
   static int count;
}; // fim da classe Example
```

Respostas dos exercícios de revisão

10.1 a) inicializadores de membro. b) `friend`. c) `new`, ponteiro. d) inicializado. e) `static`. f) `this`. g) `const`. h) construtor-padrão. i) não-`static`. j) antes. k) `delete`.

10.2 Erro: A definição de classe para `Example` tem dois erros. O primeiro ocorre na função `getIncrementedData`. A função é declarada `const`, mas modifica o objeto.
Correção: Para corrigir o primeiro erro, remova a palavra-chave `const` da definição de `getIncrementedData`.
Erro: O segundo erro ocorre na função `getCount`. Essa função é declarada `static`, portanto não tem permissão de acessar nenhum membro da classe não-`static`.
Correção: Para corrigir o segundo erro, remova a linha de saída da definição `getCount`.

Exercícios

10.3 Compare e contraste os operadores de alocação e de desalocação dinâmica de memória `new`, `new []`, `delete` e `delete []`.

10.4 Explique a noção de amizade em C++. Explique os aspectos negativos de amizade como descritos no texto.

10.5 Uma definição da classe `Time` correta pode incluir os dois construtores a seguir? Se não, explique por quê.

```
Time( int h = 0, int m = 0, int s = 0 );
Time();
```

10.6 O que acontece quando um tipo de retorno, mesmo `void`, é especificado para um construtor ou destrutor?

10.7 Modifique a classe `Date` na Figura 10.10 para obter as seguintes capacidades:
a) Dar saída para a data em múltiplos formatos como

```
DDD YYYY
MM/DD/YY
June 14, 1992
```

b) Utilizar construtores sobrecarregados para criar objetos `Date` inicializados com datas dos formatos na parte (a).
c) Criar um construtor `Date` que lê a data de sistema utilizando as funções-padrão de biblioteca do cabeçalho `<ctime>` e configurar os membros `Date`. (Consulte a documentação de referência do seu compilador ou `www.cplusplus.com/ref/ctime/index.html` para obter informações sobre as funções no cabeçalho `<ctime>`.)

No Capítulo 11, seremos capazes de criar operadores para testar a igualdade de duas datas e compará-las para determinar se uma vem antes ou depois da outra.

10.8 Crie uma classe `SavingsAccount`. Utilize um membro de dados `static annualInterestRate` para armazenar a taxa de juros anual para cada um dos correntistas. Cada membro da classe contém um membro de dados `private savingsBalance` para indicar a quantia que os correntistas têm atualmente em depósito. Forneça a função-membro `calculateMonthlyInterest` que calcula os juros mensais multiplicando o `balance` [saldo] pelo `annualInterestRate` dividido por 12; esses juros devem ser adicionados a `savingsBalance`. Forneça uma função-membro `static modifyInterestRate` que configura o `static annualInterestRate` com um novo valor. Escreva um programa de driver para testar a classe `SavingsAccount`. Instancie dois objetos diferentes da classe `SavingsAccount`, `saver1` e `saver2`, com saldos de $ 2.000,00 e $ 3.000,00, respectivamente. Configure o `annualInterestRate` como 3%. Em seguida, calcule os juros mensais e imprima os novos saldos de cada um dos correntistas. Então configure o `annualInterestRate` como 4%, calcule os juros do próximo mês e imprima os novos saldos para cada um dos poupadores.

10.9 Crie a classe `IntegerSet` pela qual cada objeto pode armazenar inteiros no intervalo 0 a 100. Um conjunto é representado internamente como um array de uns e zeros. O elemento do array `a[ i ]` é 1 se o inteiro *i* estiver no conjunto. O elemento do array `a[ j ]` é `false` se o inteiro *j* não estiver no conjunto. O construtor-padrão inicializa um conjunto para o chamado 'conjunto vazio', isto é, um conjunto cuja representação de array só contém zeros.

Forneça funções-membro para as operações comuns de conjuntos. Por exemplo, forneça uma função-membro `unionOfSets` que cria um terceiro conjunto que seja a união teórica de dois conjuntos existentes (isto é, um elemento do array do terceiro conjunto é configurado como 1 se esse elemento for 1 em qualquer um dos conjuntos existentes, ou em ambos, e um elemento do array do terceiro conjunto é configurado como 0 se esse elemento for 0 em cada um dos conjuntos existentes).

Forneça uma função-membro `intersectionOfSets` que cria um terceiro conjunto que seja a intersecção teórica de dois conjuntos existentes (isto é, um elemento do array do terceiro conjunto é configurado como 0 se esse elemento for 0 em qualquer um ou ambos os conjuntos existentes, e um elemento do array do terceiro conjunto é configurado como 1 se esse elemento for 1 em cada um dos conjuntos existentes).

Forneça uma função-membro `insertElement` que insere um novo inteiro *k* em um conjunto (configurando `a[ k ]` como 1). Forneça uma função-membro `deleteElement` que exclui o inteiro *m* (configurando `a[ m ]` como 0).

Forneça uma função-membro `printSet` que imprime um conjunto como uma lista de números separados por espaços. Imprima somente aqueles elementos que estiverem presentes no conjunto (isto é, sua posição no array tem um valor de 1). Imprima `---` para um conjunto vazio.

Forneça uma função-membro `isEqualTo` que determina se dois conjuntos são iguais.

Forneça um construtor adicional que recebe um array de inteiros e o tamanho desse array e utiliza o array para inicializar um objeto configurado.

Agora escreva um programa de driver para testar sua classe `IntegerSet`. Instancie diversos objetos `IntegerSet`. Teste se todas as suas funções-membro funcionam adequadamente.

10.10 Seria perfeitamente razoável para a classe `Time` das figuras 10.18–10.19 representar a hora internamente como o número de segundos desde a meia-noite em vez representá-la com os três valores de inteiro `hour`, `minute` e `second`. Os clientes poderiam utilizar os mesmos métodos `public` e obter os mesmos resultados. Modifique a classe `Time` da Figura 10.18 para implementar a hora como o número de segundos desde a meia-noite e mostrar que não há alteração visível na funcionalidade para os clientes da classe. [*Nota:* Este exercício demonstra com precisão as virtudes do ocultamento da implementação.]