RESOLUÇÃO DA 2ª LISTA DE EXERCICIO

1°/ PROGRAMAÇÃO FUNCIONAL ET APRESENTAÇÃO SUAS VANTAGENS E DESVANTAGENS.

A programação funcional tem uma abordagem diferente. O desenvolvedor deve escrever funções e o software irá funcionar com base na interação entre essas expressões matemáticas. O resultado de uma função serve como parâmetro para outras e assim por diante.

A programação funcional gera um código completo e autossuficiente que pode trazer diversos benefícios ao trabalho dos desenvolvedores. Tendo em vista que o código é mais fácil de ser refeito, o modelo pode receber mudanças e manutenção com facilidade. Isso não ocorre na programação imperativa, em que as transformações manuais podem gerar repetições ou incongruências. Outro benefício é a facilidade nos testes e na busca por bugs. Como as funções são mais fáceis de serem avaliadas de forma isolada, os testes podem ser direcionados para expressões específicas, possibilitando programas mais concisos e imunes a erros. Apesar desses benefícios, a programação funcional também tem desvantagens. O esforço inicial para o desenvolvimento do programa é uma delas, tendo em vista que pode ser maior que nas linguagens tradicionais. Além disso, na programação funcional é mais difícil estimar o tempo e os recursos necessários para o trabalho de desenvolvimento, tornando o trabalho de gestão do projeto um pouco mais complexo.

2°) EMPRESAS QUE USAM AS LINGUAGEMS FUNCIONAIS E SEUS DOMINIOS DE USO

Facebook : linguagem HASKELL na manipulação da base de código PHP .

Google: linguagem na HASKELL infra-estrutura interna de TI.

Microsoft :  [linguagem F#](http://pt.wikipedia.org/wiki/F%E2%99%AF) na plataforma .NET .

3°) DEFINAÇÃO ORIENTAÇÃO A OBJETOS (OO) E APRESENTAÇÃO DOS BENEFÍCIOS

A Programação Orientada a Objetos (POO) foi criada para tentar aproximar o mundo real do mundo virtual: a idéia fundamental é tentar simular o mundo real dentro do computador. O programador é responsável por moldar o mundo dos objetos, e explicar para estes objetos como eles devem interagir entre si. Os objetos "conversam" uns com os outros através do envio de mensagens, e o papel principal do programador é especificar quais serão as mensagens que cada objeto pode receber, e também qual a ação que aquele objeto deve realizar ao receber aquela mensagem em específico. Como vantagens , os sistemas possuem uma divisão de código um pouco mais lógica e melhor encapsulada do que a empregada nos sistemas não orientados a objetos. Isto torna a manutenção e extensão do código mais fácil e com menos riscos de inserção de bugs. Também é mais fácil reaproveitar o código. É mais fácil gerenciar o desenvolvimento deste tipo de software quando temos uma equipe grande. Podemos fazer uma especificação UML antes de iniciar o desenvolvimento do software em si, e em seguida dividirmos o sistema em classes e pacotes, e cada membro da equipe pode ficar responsável por desenvolver uma parte do sistema.

4°) APRESENTAÇÃO E A DETERMINAÇÃO DO NIVEL DE ACESSO; ESCOPO E TIPO DAS VARIAVEIS/ATRIBUTOS

|  |  |  |  |
| --- | --- | --- | --- |
| **VARIAVEIS/ATRIBUTOS** | **NIVEL DE ACESSO** | **ESCOPO** | **TIPO** |
| year (variável) | public | Car | int |
| make (variável) | public | Car | String |
| speed (variável) | public | Car | double |
| num (variável) | private | Car | int |
| tmp (variável) | public | getYear | int |
| r (variável) | public | getYear | Roda |
| Roda (Atributo) | public | getYear | int |

5°) DETERMINAÇÃO DE ERRO RELACIONADO A MEMORIA

Sim, existe um problema em relação á memoria. na linha 21. \*b++=0 : estamos sobrescrevendo o endereço da mémoria antecipadamente alocada no ponteiro b. essa referência sera perdida e não desalocada ao final do código.