Uma Receita para Programar

1. **Analise o problema e a informação disponível sobre mesmo**. Para tal leia o enunciado com cuidado e várias vezes. O enunciado contem muitas indicações sob a forma de melhor estruturar o seu programa. **Represente essa informação sob a forma de classes/objectos e respectivos atributos**. Deve também identificar relações *is-a* (herança), *has-a* (agregação) e dependências entre os diversos objectos.
2. **Escreva um pequeno programa de teste**. Este programa de teste pode ter a forma de uma nova classe, que não irá pertencer ao programa a implementar mas que é extremamente importante para o seu desenvolvimento. No construtor dessa nova classe deve utilizar as classes e respectivos métodos que constituem o programa que se pretende desenvolver. Este código serve como exemplo de utilização e como teste do programa. Caso o código de teste se torne muito longo (o que é muito provável) defina outros métodos na classe de teste como forma de o decompor.
3. Para cada construtor e método que identificou no passo anterior, **escreva um esqueleto de dados (variáveis) para cada método**. Este esqueleto de dados é constituído pelos cabeçalhos dos métodos e por corpos (definições desses métodos) que contêm apenas os dados que serão utilizados (ainda não se sabe exactamente como) por cada método. Serve como um rascunho do passo seguinte.
4. **Defina o corpo de cada método**.
5. **Teste o programa** utilizando e alterando onde necessário o código de teste já desenvolvido no passo 2.

Nota importante: os passos apresentados podem ser percorridos vária vezes pois tipicamente é necessário voltar várias vezes a passos anteriores. Por exemplo, os passos 1, 2 e 3 são tipicamente realizados de forma muito iterativa: ao fazer um exemplo de utilização, no passo 2, utilizando as classes/objectos identificados no passo 1, poderá facilmente concluir que falta uma classe, objecto ou atributo não identificado no passo 1.

Esta receita é uma adaptação da *Design Recipe* (*in* Viera K. Proulx and Tanya Cashorali, "Calculator problem and the design recipe" ACM SIGPLAN Notices Volume 40 , Issue 3 (March 2005) http://doi.acm.org/10.1145/1057474.1057478