Programa de curso: Matlab Intermediário

Motivação

O curso visa reforçar a capacidade da equipe de desenvolver em ambiente Matlab, especialmente em programação orientada a objetos (OOP).

É possível implementar tarefas simples de programação como simples funções. Entretanto, para tarefas de grande complexidade, tratar tudo como funções torna-se muito complexo e suscetível a erros. Neste contexto, a OOP oferece uma série de vantagens:

- **Maior familiaridade ao usuário *(user-friendliness)*:** Depois de criado, a classe requer pouco ou nenhum conhecimento de sua implementação para seu uso pelo usuário. Isso permite que o usuário foque em compreender os métodos que permitem criar e alterar objetos, e não em decorar detalhes do código. **Há também uma redução de redundância e complexidade** por meio da reutilização e herança de código. O programa se torna modular e simples de entender;
- **Compreensão mais profunda de problemas, ao pensarmos em termos de seus objetos:** o que é muito mais natural para alguns problemas. **Identificação de semelhanças e diferenças entre objetos análogos** permite extrair comportamentos comuns (descritos em superclasses) e comportamentos específicos (descritos em subclasses);
- **Maior facilidade de programação:** embora a OOP requeira uma fase maior de planejamento, a versatilidade do código OOP torna a programação muito mais simples a partir de uma certa complexidade de tarefa;
- **Menor necessidade de manutenção e correção de código:** é menor a chance de o usuário alterar algo no código indevidamente, uma vez que ele interage com o objeto via métodos pré-definidos. Além disso, os **objetos podem gerenciar seu estado interno** e identificar assim inconsistências nas entradas fornecidas pelo usuário.

Em resumo, a OOP significa **menos chance de falha humana**, **mais rapidez no atendimento a demandas** e uma **curva de aprendizado mais íngreme** para novos usuários.

Possível programa do curso

1. Conceitos gerais de Matlab (revisão)
    a. Classes fundamentais do Matlab: numéricas (double, uint32 etc), cell, char, logical, struct, table, handles de função; outras classes: datetime
    b. Comandos: if, for, switch
    c. Tratamento de erros: try/catch
    d. Indexação: por ponto, por chaves, por parênteses, lista separada por vírgula; nomes de campo dinâmicos
    e. Criando funções
        i. Criação de uma função simples
        ii. Múltiplas entradas/saídas
        iii. Encapsulando em pastas: o comando import
    f. Escopos de dados (workspaces)
2. Conceitos de OOP
    a. Programação orientada a objetos vs. Programação procedural: diferenças e vantagens da OOP

b. O processo de programação em OOP: identificando comportamentos comuns e específicos
c. Conceitos em OOP: classe, propriedade, objeto, método, evento, atributo, superclasse, subclasse
d. Classes
    i. Propriedades
        1. Atributos de acesso (public/private/protected)
        2. Outros atributos: Dependent, Hidden, Constant
    ii. Métodos
        1. O método construtor de uma classe
        2. Restrição de acesso (public/private/protected)
        3. Métodos set/get
        4. Métodos Static
    iii. Eventos / listeners
e. Atributos de classe
    i. Semântica handle e value
    ii. Abstract, *Hidden, Sealed, HandleCompatible*
f. Herança de classes
    i. Herança simples e múltipla
    ii. Precedência de métodos
    iii. Boas práticas de programação
g. Overloading de operadores
h. Classes pré-definidas do Matlab
    i. Classes gerais: char, double etc.
    ii. Outras classes que personalizam comportamentos específicos

3. O comando actxserver: escrevendo em Excel a partir do Matlab
    a. Diferenças para o xlswrite/xlsread
    b. Comandos básicos

4. Boas práticas de programação em Matlab