Aula 12 Funções

Roteiro

1. Calculadora Financeira
   - Cálculo de Juros de Prestações
   - Cálculo de Prestações
   - Retorno de Aplicação
2. Exercícios

Calculadora Financeira

- Vamos criar um programa com algumas funções de matemática financeira.
- O programa deve ter as seguintes funcionalidades:
  - Dado um montante inicial **mont** aplicado em um fundo com taxa de juros **ju** por período, deve-se calcular o valor aplicado após **per** períodos.
  - Uma variação do item anterior, onde todo mês uma quantia **apl** será aplicada por período.
  - Calculo de prestações: Dado um valor à vista **valorProd** de um produto, deve-se determinar o valor **valorPrest** das prestações que devem ser pagas, assumindo-se **per** períodos e taxa de juros **ju**.
  - Uma inversão do item anterior: Devemos computar os juros cobrados dado o valor das prestações e o valor à vista do produto.

Cálculo de Juros de Prestações

- Vamos criar funções separadas para cada funcionalidade.
- Vamos começar pela última funcionalidade:
  - Computar os juros reais cobrados, quando compramos um produto cujo valor à vista é **valorProd**, com prestações no valor **valorPrest** que devem ser pagas em **per** períodos.
  - O valor dos juros **ju** cobrados satisfaz a equação abaixo:

$$\text{valorProd} \cdot (1 + \text{ju})^{\text{per}} - \text{valorPrest} \cdot \frac{(1 + \text{ju})^{\text{per}} - 1}{\text{ju}} = 0$$

  - Ou seja, devemos achar o valor de **ju** que é um zero da função.

Cálculo de Juros de Prestações

- Vamos utilizar o método de Newton para isso:
  - Dado uma função $f(x)$, podemos achar os zeros dessa função com sucessivas aproximações.
  - Seja x_0 um valor inicial que achamos estar próximo do zero da função.
  - Dado uma aproximação x_n anterior, uma próxima aproximação melhor é computada pela equação:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

No nosso caso:

$$f(ju)' = \mathrm{valorProd} \cdot \mathrm{per} \cdot (1 + \mathrm{ju})^{\mathrm{per}-1} - \mathrm{valorPrest} \cdot \left(\frac{\mathrm{per} \cdot (1 + \mathrm{ju})^{\mathrm{per}-1}}{\mathrm{ju}} - \frac{(1 + \mathrm{ju})^{\mathrm{per}} - 1}{\mathrm{ju}^2} \right)$$

Cálculo de Juros de Prestações

Criamos uma função para avaliar

$$f(ju) = \text{valorProd} \cdot (1 + \text{ju})^{\text{per}} - \text{valorPrest} \cdot \frac{(1 + \text{ju})^{\text{per}} - 1}{\text{ju}}$$

```
double funcaoFx(double valorProd, int per, double valorPrest, double juros){
  double pote, aux =0;
  pote = pow(1+juros, per);
  return valorProd*pote - valorPrest*((pote-1)/juros);
}
```

OBS: Estamos utilizando a função **pow** da biblioteca **math.h** para computar potências.

Cálculo de Juros de Prestações

Criamos uma função para avaliar

$$f(ju)' = \text{valorProd} \cdot \text{per} \cdot (1+\text{ju})^{\text{per}-1} - \text{valorPrest} \cdot \left(\frac{\text{per} \cdot (1+\text{ju})^{\text{per}-1}}{\text{ju}} - \frac{(1+\text{ju})^{\text{per}} - 1}{\text{ju}^2} \right)$$

```
double derivadaFx(double valorProd, int per, double valorPrest, double juros){
  double pote1, pote2, aux;
  pote1 = pow(1+juros, per);
  pote2 = pow(1+juros, per-1);
  aux = valorProd*per*pote2;
  aux = aux - valorPrest*per*pote2/juros;
  aux = aux + valorPrest*(pote1 - 1)/(juros*juros);
  return aux;
}
```

Cálculo de Juros de Prestações

- As sucessivas aproximações são computadas segundo:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

- Podemos fazer $x_0 = 1$, pois provavelmente $0 \leq$ ju ≤ 1.
- Faremos sucessivas aproximações, mas quando parar?
  - Quando acharmos **ju** que faz a equação ser próxima o suficiente de zero:

$$f(ju) = \text{valorProd} \cdot (1 + \text{ju})^{\text{per}} - \text{valorPrest} \cdot \frac{(1 + \text{ju})^{\text{per}} - 1}{\text{ju}} \approx 0$$

Cálculo de Juros de Prestações

Criamos uma função para detectar se estamos próximos o suficiente de zero:

```
int proxSuficiente(double valorProd, int per, double valorPrest, double juros){
  double valorFun = funcaoFx(valorProd, per, valorPrest, juros);
  if(valorFun <= EPSILON && valorFun >= -1 * EPSILON )
    return 1;
  return 0;
}
```

OBS: **EPSILON** é uma constante definida após a seção de bibliotecas com o comando:

```
#define EPSILON 0.000000001
```

Cálculo de Juros de Prestações

Com todas as funções anteriores estamos prontos para aplicar o método de Newton e achar o valor dos juros cobrados.

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

- O nosso algoritmo deverá funcionar da seguinte forma: A cada 1000 aproximações, verifica-se se encontramos um zero da função.

```
xAtu = 1.0
Enquanto não achar zero da função
      Repita 1000 vezes
           xAtu = xAtu - f(xAtu)/f'(xAtu)
```

Cálculo de Juros de Prestações

Agora em C utilizando as funções anteriores:

```
double achaJuros(double valorProd, int per, double valorPrest){
  int i;
  double juros=1.0;

  while(!proxSuficiente(valorProd, per, valorPrest, juros) ){
    for(i=1; i<=1000; i++){
      juros = juros - funcaoFx(valorProd, per, valorPrest, juros)/derivadaFx(valorPr
    }
  }
  return juros;
}
```

Cálculo de Prestações

Temos as demais funcionalidades:

- Calculo de prestações: Dado um valor à vista **valorProd** de um produto, o valor **valorPrest** das prestações que devem ser pagas, assumindo-se **per** períodos e taxa de juros **ju** é:

$$\text{valorPrest} = \frac{(1+\text{ju})^{\text{per}} \cdot \text{valorProd} \cdot \text{ju}}{(1+\text{ju})^{\text{per}} - 1}$$

Retorno de Aplicação

Útimas duas funcionalidades:

- Dado um montante inicial **mont** aplicado em um fundo com taxa de juros **ju** por período, deve-se calcular o valor aplicado após **per** períodos.

$$\text{valorFim} = (1 + \mathbf{ju})^{\mathbf{per}} \cdot \text{mont}$$

- Uma variação do item anterior, onde todo mês uma quantia **apli** será aplicada por período.

$$\text{valorFim} = (1 + \text{ju})^{\mathbf{per}} \cdot \text{mont} + \text{apli} \cdot \left(\frac{(1 + \text{ju})^{\text{per}} - 1}{ju} \right)$$

Exercício

- Para cada uma das fórmulas das funcionalidades faltantes do programa de aplicação financeira, escreva uma função em C.
- Escreva um programa com uma interface que pede qual tipo de informação financeira deseja-se calcular, depois pede os dados de entrada necessários, e por fim imprime o resultado. Use as funções anteriores e as apresentadas nesta aula.