Introdução à Programação Funcional

- Programação de computadores
- Computadores e modelamento
- Programação funcional

Programas de computador (1)

- A finalidade de um computador é receber, manipular e armazenar dados
  - –Processamento de dados
- Possui duas partes:
  - –Hardware e software
- Desenvolvimento de software através de programas

Programas de computador (2)

- Desenvolvimento de um programa:
- Análise: definição dos dados de entrada, processamento e dados de saída
- Algoritmo: descrever o problema com suas soluções
- Codificação: transformar o algoritmo para uma linguagem de programação

Linguagens de programação (1)

- Algoritmos podem ser representados em código diretamente em linguagem de programação
- Linguagens de programação são construídas utilizando palavras reservadas em inglês
- Um programa nada mais é que uma representação de um algoritmo

Linguagens de programação (2)

- Utilizamos uma linguagem de programação para realizar a representação
- Linguagem de máquina – linguagem de baixo nível
- Linguagem de alto nível – mais próximas da linguagem natural

Linguagem de máquina (1)

- A linguagem de programação que um computador é capaz de compreender é composta apenas de números

Linguagem de máquina (2)

- Conjunto de instruções que hardware do computador é capaz de executar
- Cada instrução é expressa por seqüências de bits
- Exemplo de instruções:
  - **00001 - carregar o acumulador**
  - **00010 - armazenar no acumulador**
  - **00100 - somar ao acumulador**
  - **00101 - subtrair do acumulador**

Exemplo de programa

00011	00000001010
10001	00000001011
00001	00000001011
01101	00000001000
01100	00000001000
00100	00000001010
00010	00000001010
01011	00000000001
10010	00000001010
10011	00000000000

Linguagem de máquina (3)

- Em linguagem de máquina, o trabalho é muito tedioso e sujeito a erros
- Para aliviar um pouco esses incômodos, os programas poderiam ser expressos em decimal.
- O programa ainda precisaria ser convertido automaticamente para binário.

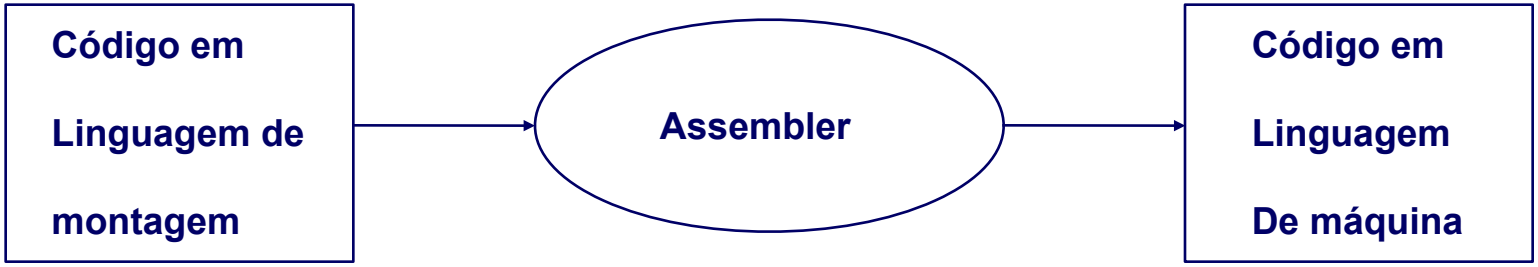
Linguagem assembly

- Linguagem de montagem ou linguagem assembly
- Expressa as instruções de máquina de forma mais clara
- A operação passou a ser expressa por mnemônicos
- Endereçamento simbólico

Exemplo de programa

```
                STZ     soma
Leitura:        READ    numero
                LD      numero
                LZ      escrita
                JN      escrita
                ADD     soma
                ST      soma
                JUMP    leitura
Escrita:        WRITE   soma
                STOP
```

Montador ou assembler



Linguagem de alto nível

- A implementação de programas em assembly ainda é muito complexa
- Dependente do conhecimento das instruções do processador
- Solução: Linguagens de alto nível
  - **Aumenta a produtividade dos programadores**
  - **Aumenta a portabilidade dos programas**

Linguagem FORTRAN

- Em 1954 surgiu a primeira linguagem de propósitos gerais para a solução de problemas matemáticos e científicos
- FORTRAN – FORmula TRANslation
- Toda a linguagem imperativa pode ser expressa em comandos

Programa exemplo

```
      INTEGER Soma, Numero
      Soma = 0
10    READ*, Numero
      IF (Numero.LE.0) GOTO 20
         Soma = Soma + Numero
         GOTO 10
20    WRITE*, Soma
      STOP
      END
```

Linguagens de programação

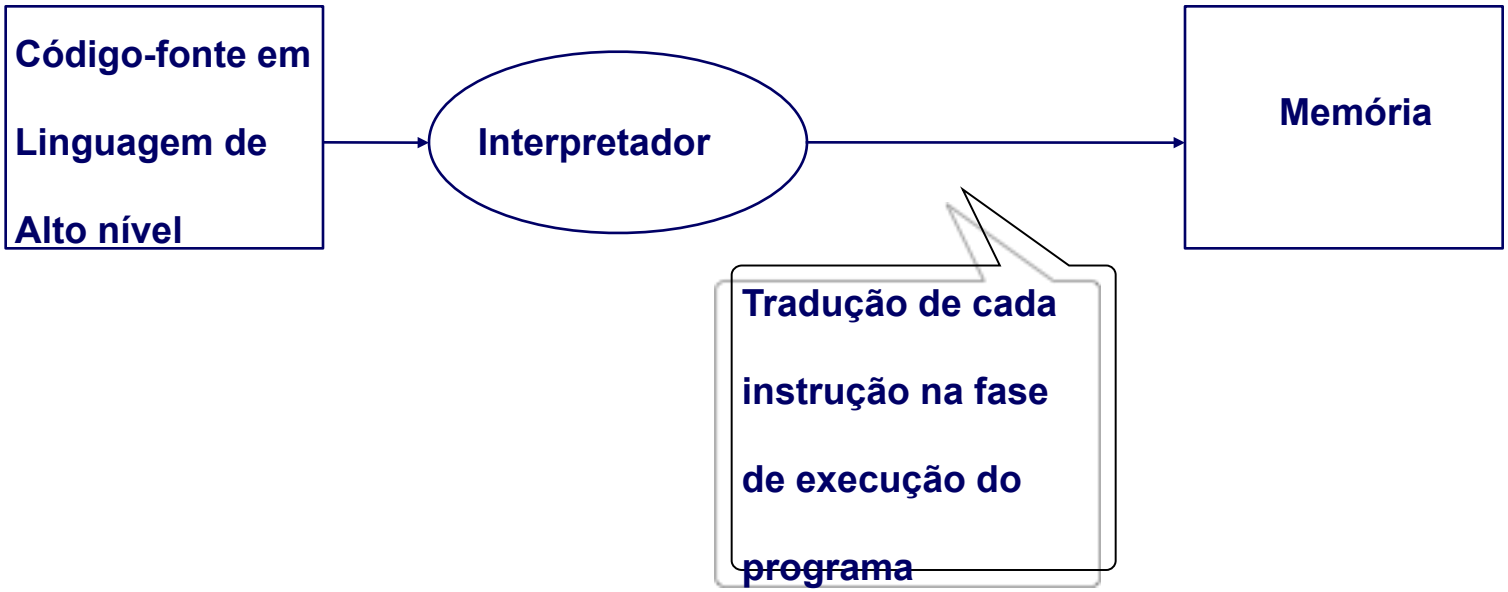
- Com o passar do tempo, linguagens mais bem estruturadas e mais poderosas foram surgindo
- Com diferentes paradigmas para se abordar um problema (funcional, imperativo, OO, concorrente)
- Exemplos:
  - **COBOL, ALGOL, BASIC, PASCAL, Modula-2, PL-1, C, C++, Java, C#, Delphy, LISP, PROLOG, Haskell, etc.**

Alto nível versus Baixo Nível

- É mais fácil programar em linguagens de alto nível
  - **Menos tempo para escrever o programa;**
  - **O programa fica mais curto;**
  - **Fica mais fácil de ler;**
  - **Menos propenso a introdução de erros**
- Programas em linguagens de alto nível são portáveis
- Programas em linguagens de baixo nível ficam amarradas a um tipo de computador.
- Linguagens de baixo nível são usadas em aplicações especiais.

Executando um programa (1)

Executando um programa (2)



Computadores e modelamento (1)

- Um dos objetivos de um computador é manipular informações simbólicas
- Essas informações podem representar uma situação simples,
- como os itens comprados em uma viagem de compras de supermercado

Computadores e modelamento (2)

- Precisamos escrever uma descrição de como as informações são manipuladas.
- Como vimos, isso é chamado de programa
- e é escrito em uma linguagem de programação.

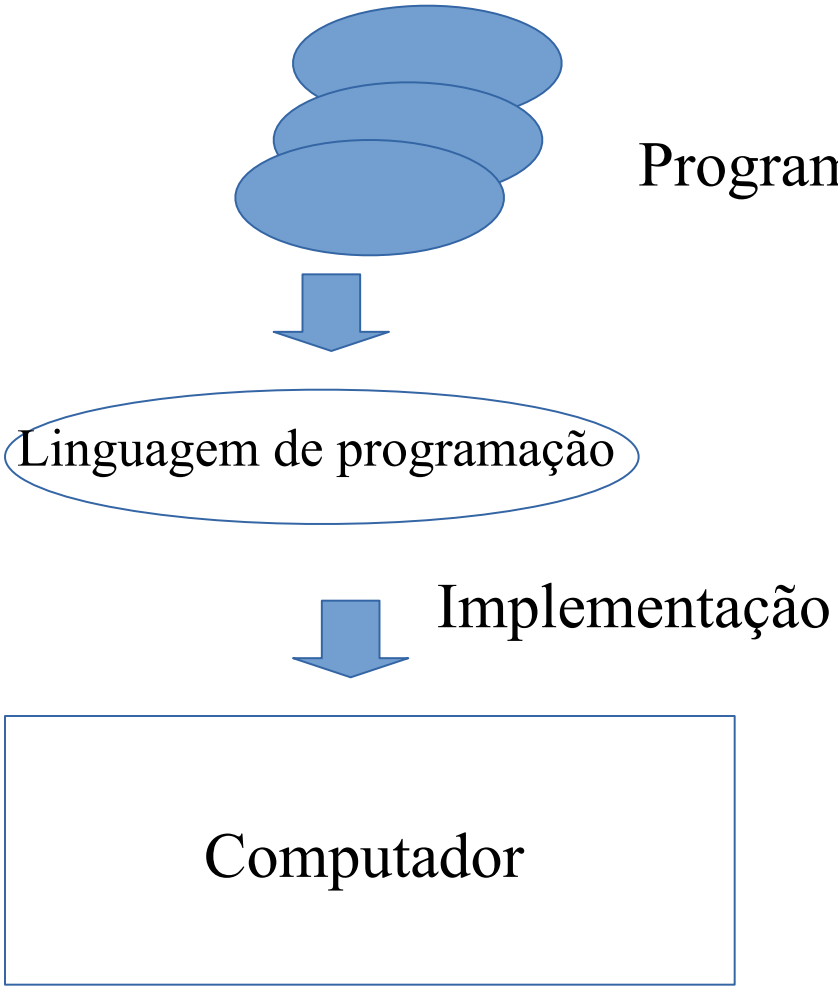
Computadores e modelamento (3)

- Uma linguagem de programação é uma linguagem formal e artificial usada para dar instruções a um computador.
- Em outras palavras, a linguagem é usada para escrever o software que controla o comportamento do hardware.

Computadores e modelamento (4)

- A linguagem de programação é feita para funcionar em um computador por uma implementação,
- que é propriamente um programa e que executa programas escritos na linguagem de nível superior no computador em questão.

Computadores e modelamento (5)



Programação funcional (1)

- Nosso assunto aqui é programação funcional, que é um de vários estilos ou paradigmas de programação diferentes
- Uma maneira muito proveitosa de olhar para a programação é que é a tarefa de modelar situações dentro de um computador.
  - no mundo real ou no imaginário

Programação funcional (2)

- Um programador funcional se concentra nas relações entre valores,
- enquanto um programador 00 se concentra nos objetos.
- Um programador imperativo se concentra em variáveis e comandos

Programação funcional (3)

- Um programa funcional é um conjunto de definições
- Uma definição associa um nome a um valor
- exemplo:
- k = 6.66
- func x = k * x
- O que faz esse programa ?

Programação funcional (4)

- Programar é definir estruturas de dados e funções para resolver determinado problema
- O interpretador da linguagem funcional atua como uma máquina de calcular
- Lê uma expressão, calcula seu valor e mostra o resultado

Programação funcional (5)

- Exemplo 1:
- Um programa para converter valores de temperatura em graus Celsius para graus Fahrenheit
- celfar c = c * 1.8 + 32
- Depois de carregar o programa no interpretador Haskell
- celfar 25
- 77.0

Programação funcional (6)

- Exemplo 2:
- Um programa para converter valores de temperatura em graus Celsius para graus Fahrenheit e de graus Kelvin para graus Celsius
- celfar c = c * 1.8 + 32
- kelcel k = k - 273

Programação funcional (7)

- Depois de carregar o programa no interpretador Haskell
- podemos fazer os seguintes testes:
- celfar 25
- 77.0
- kelcel 0
- -273

Programação funcional (8)

- A um conjunto de associações nome-valor dá-se o nome de ambiente ou contexto
- As expressões são avaliadas no âmbito de um contexto
- O interpretador usa as definições que tem no contexto como regras de cálculo, para simplificar o valor da expressão

Programação funcional (9)

- Exemplo 3:
- Este programa define 3 funções de conversão de temperaturas
- celfar c = c * 1.8 + 32
- kelcel k = k - 273
- kelfar k = celfar (kelcel k)

Programação funcional (10)

- No interpretador:
- kelfar 300
- 80.6
- é calculado pelas regras estabelecidas pelas definições fornecidas pelo programa

Programação funcional (11)

- kelfar 300

==>celfar (kelcel 300)

==>(kelcel 300) * 1.8 + 32

==>(300 - 273) * 1.8 + 32

==> 27 * 1.8 + 32

==> 80.6

Leitura recomendável

- MEDINA & FERTING. Algoritmos e Programação: Teoria e Prática. São Paulo, Novatec Editora, 2005
  - **Capítulo 1 (1.2)**
- MARIA JOÃO FRADE. Programação funcional CC. Notas de aulas. Departamento de Informática. Universidade do Minho 2007/2008