ILP-010 – Linguagem de Programação

ADS – Análise e Desenvolvimento de Sistemas
Prof. Vagner Macedo

Aula 01

Ementa

- Variáveis, constantes, operadores e expressões. Comando de desvio. Controle de fluxo. Vetores. Funções de biblioteca. Estruturas, uniões e tipos definidos pelo usuário. Manipulação de arquivos.

Objetivos gerais

- Solucionar problemas utilizando a lógica de programação e a implementação de programas por meio de uma linguagem de programação (no caso, Java).

Conteúdo Básico

- Apresentação da disciplina e conceitos básicos sobre de Linguagem de Programação.
- Conceitos e primeiros códigos.
- Tipos de dados e variáveis.
- Entrada de dados: tipos primitivos, valores literais e variáveis; escopo de variáveis e conversão de tipos; entrada de dados em Java.
- Operadores e expressões: precedência de operadores; expressões. Depuração de programas.
- Sub-rotinas.

Conteúdo Básico

- Controle condicional (estrutura de controle condicional).
- Controle de repetição (estrutura de controle de repetição).
- Tratamento de erros e exceções.
- Vetores e matrizes.
- Funções e algoritmos recursivos.
- Conjuntos: Set, Map, List; principais interfaces de conjuntos.
- Manipulação de arquivos -> Manipulação de arquivos.

Bibliografia Básica

- 1 - ASCENCIO, A. F. G. e CAMPOS, E. A. V.; Fundamentos da programação de computadores: Algoritmos, Pascal, C/C++ e Java. 3ª. Ed. Pearson;
- 2 - Deitel; Java Como Programar; Ed. Bookman;
- 3 - Sérgio Furgeri; Java 2 Ensino Didático; Ed. Érica.

Conceitos Básicos

- Tecnologia
  - Dentro do campo que nos interessa, a tecnologia tem o objetivo de eliminar as tarefas repetitivas, facilitando o trabalho e tornando-o mais eficiente, como aumentando a produtividade e os benefícios para a empresa. Atualmente a tecnologia também nos auxilia com temas relacionados ao aprendizado, às comunicações, ao lazer e às tarefas rotineiras que possam ser automatizadas.

Conceitos Básicos

- Programa de Computador
  - É uma coleção de instruções que descrevem uma tarefa a ser realizada pelo computador.
  - Também é conhecido como software, aplicativo, *app,* sistema ou simplesmente "programa".
  - É finito, ou seja, tem que ter um início e um fim, podendo ser abreviado em caso de erro de software ou de hardware.

Conceitos Básicos

- Programação de Computador
  - É o processo de escrita, teste e manutenção de um programa de computador.
  - A programação é escrita em uma *linguagem de programação*.

Conceitos Básicos

- Linguagem de programação
  - É um conjunto de regras sintáticas e semânticas usadas para definir um programa de computador.
  - O conjunto de palavras (tokens), composto de acordo com essas regras, constitui o ***código fonte*** de um software. Esse **código fonte** é depois traduzido para código de máquina, que é executado pelo processador.

Conceitos Básicos

- Linguagem de programação
  - Quanto ao arranjo de seus comandos, podem ser, basicamente, de 2 tipos:
    - Linguagens de Baixo Nível:
      - Trabalham passando instruções diretamente ao processador e a seus registradores, respeitando as características da arquitetura de cada computador. Ex.: Assembly.
    - Linguagens de Alto Nível:
      - Utilizam comandos e palavras mais próximas ao linguajar humano, visando facilitar a produtividade e o entendimento pelos programadores. Ex.: Java, C, C++ e C#.

Conceitos Básicos

- Metas da Linguagem de Programação:
  - Permitir que programadores tenham uma maior produtividade;
  - Disponibilizar sintaxe que possa ser mais facilmente entendida por programadores humanos;
  - Possibilitar a escrita de programas mais organizados e com maior rapidez.

História – Algumas linguagens

- 1945 – Plankalkül
  - Primeira linguagem de programação.
- 1954 – Fortran
  - Primeira linguagem amplamente usada.
  - Projetada para o IBM704
- 1958 – Algol
  - Formalizou o conceito de tipo.
  - Cláusula else-if

História – Algumas linguagens

- 1959 – LIST
  - Primeira linguagem funcional.
  - Controle do fluxo de execução através de recursão e expressões condicionais.
- 1960 – COBOL
  - Operadores eram descritos por palavras em inglês.
  - Código e dados separados.
  - Largamente utilizada até os dias de hoje.

História – Algumas linguagens

- 1962 - Linguagens Dinâmicas
  - APL - 1962; Snobol 1964.
  - Tipificação dinâmica e alocação dinâmica.
  - Programas com legibilidade ruim.
- 1964 - Basic
  - Foco voltado à facilidade de aprendizado e uso da linguagem, mesmo por usuários não especializados.
  - Tempo do usuário é mais importante que o tempo de máquina.

História – Algumas linguagens

- 1965 - PL/I
  - Tratamento de exceções e uso de apontadores.
  - Código e dados separados.
  - Largamente utilizada até os dias de hoje.
- 1967 - Simula 67 (baseada em Algol 60 e Simula I)
  - Abstração de dados (conceito de classe).
- 1968 - Algol 68
  - Estruturas de dados definidas pelo usuário.
- 1970 – SQL
  - Linguagem de consultas em banco de dados relacionais.

História – Algumas linguagens

- 1971 – Pascal
  - Recebeu o nome do matemático francês Blaise Pascal.
  - Voltada para o ensino de programação estruturada.
  - Simples, pouco extensa e sem recursos novos.
  - Ainda é largamente utilizada para o ensino de programação.
- 1972 – C
  - Projetada para programação de sistemas.
  - Conjunto poderoso de operadores.
  - Inicialmente parte integrante do sistema operacional UNIX, difundindo o seu uso.

História – Algumas linguagens

- 1972 - Prolog
  - Não procedural.
  - Pode ser resumida como sendo um banco de dados inteligente que usa um processo de inferências para inferir a verdade a partir de dadas consultas.
- 1978 - Fortran 77
  - Manipulação de strings.
  - Instruções lógicas de controle de laço e cláusula else.
- 1980 – Smalltalk
  - Primeira implementação completa de uma LP com conceitos de POO.
  - Pioneira no uso de interfaces gráficas com o usuário (GUI).

História – Algumas linguagens

- 1983 – Ada
  - Suporte para abstração de dados.
  - Tratamento de exceções.
  - Concorrência.
  - Primeiro compilador disponível apenas 5 anos após o término do projeto da linguagem.
- 1985 – C++
  - Suporte a POO para a linguagem C (evolução desta e de Simula 67).
  - Tratamento de exceções.
  - Complexa e extensa.
  - Suporta POO e Programação Estruturada.

História – Algumas linguagens

- 1990 - Fortran 90
  - Arranjos dinâmicos.
  - Apontadores, recursão, comando de seleção múltipla (case).
  - Verificação de tipos de parâmetros.
- 1990 - Java
  - Baseada em C++.
  - Suporta somente POO.
  - Não possui ponteiros, mas referências.
  - Suporte a concorrência.

História – Algumas linguagens

- 1991 – Python
  - Tipagem dinâmica.
  - Baseado nas bibliotecas C.
  - Suporta POO.
  - Utilizada com sucesso em *machine learning* recentemente.
  - Amplamente usada em ambientes Cloud (GCP, AWS, Azure e OCI)
- 1993 – R
  - Utilizada na ciência de dados.
  - Muito empregada para cálculos estatísticos e gráficos.
  - Forte uso com o avanço do Big Data.

História – Algumas linguagens

- 1995 – Javascript
  - Implementada originalmente para navegadores Netscape.
  - Criada inicialmente para interagir no lado cliente.
  - Linguagem mais utilizada em lado cliente atualmente.
  - Tipagem dinâmica.
  - Variação de lado servidor criada em 2009 (Node.js).
- 1995 - Ada 95
  - Suporta a POO.
  - Bibliotecas mais flexíveis.

História – Algumas linguagens

- 1995 - PHP
  - Largamente utilizada na Internet.
  - POO e Tipagem dinâmica.
  - Semelhança com C++ e Perl.
- 1996 – UML
  - Linguagem de modelagem.
- 2000 – C#
  - Principal linguagem da plataforma .NET.
  - delegate substituiu a função do ponteiro em C.

Compilação e Interpretação

- Compilação
  - Traduz (para linguagem de máquina) todo o código fonte de **uma única vez**, para só depois executar;
  - Diz-se que o programa foi ***compilado*** e que o mecanismo utilizado para a tradução é um ***compilador***;
  - Exemplos: Pascal e C.

Compilação e Interpretação

- Interpretação
  - Traduz (para linguagem de máquina) trechos do código fonte à medida em que vai sendo executado;
  - Diz-se que o programa foi ***interpretado*** e que o mecanismo utilizado para a tradução é um ***interpretador***;
  - Exemplos: Javascript, Python ou Perl.

Compilação e Interpretação

- Programas ***interpretados*** são geralmente mais lentos do que os ***compilados,*** mas também são, geralmente, mais flexíveis, já que podem interagir com o ambiente mais facilmente.

Compilação e Interpretação

- Mas... nem sempre é tão simples:
  - Existem as linguagens compiladas para um código de máquina de uma máquina virtual, com as VMs Java e Parrot.
  - E também há códigos-fontes que, ao invés de serem interpretados linha a linha, têm blocos "compilados" para a memória, técnica esta conhecida como JIT (compilação *just-in-time*).

Sintaxe e Semântica

- **Sintaxe**
  - A *forma* ou *estrutura* das expressões, comandos e unidades de programa
- **Semântica**
  - O *significado* das expressões, comandos e unidades de programa.

Sintaxe e Semântica

- **Ex. de Semântica:**
  - A sessão do terreno ocorreu na semana passada.
  - A seção de cinema já começou.
  - A cessão de laticínios fica ao final do corredor.
  - A ceção de teatro foi cancelada.

Exemplos de Sintaxe

- **JavaScript**

```
function somarDoisValores (a, b)  {
  return a + b;
 };
```

- **C/C++/Java/C#/D**

```
public int SomarDoisValores (int a, int b) {
  return a + b;
}
```

- **Cobol**

```
compute c = a + b.
```

- **Visual Basic**

```
function SomarDoisValores(byval a as integer, byval b as integer) as integer
  SomaDeDoisValores = a + b
end function
```

Características de LP

- Simplicidade
  - Resultado de uma combinação de um número relativamente pequeno de construções primitivas.
- Ortogonalidade
  - Número reduzido de construções primitivas e um conjunto consistente de regras para combiná-las.
  - Quanto mais ortogonal uma linguagem é, menos exceções ou regras são necessárias para escrever os programas.

Características de LP

- Legibilidade
  - Está relacionada à leitura dos códigos. Quanto mais fácil para ler o programa, mais fácil para entender o código (e descobrir erros).
  - Ex.: “*” em C pode ser multiplicador, passagem por referência, declaração de ponteiros...
- Redigibilidade
  - Relacionada à facilidade em escrever programas.

Características de LP

- Confiabilidade
  - Refere-se aos mecanismos fornecidos pela LP para incentivar a construção de programas confiáveis (por exemplo, Tratamento de Erros e Exceções).

- Eficiência
  - Performance em tempo de execução.
  - Características de verificações antes de algum funcionamento ou antes do programa ser iniciado.

Características de LP

- Facilidade de aprendizado
  - Características de comandos e lógica simples e inteligível.

- Reusabilidade
  - Possibilidade de reutilização do mesmo código em diversas aplicações.
  - Quanto mais reusável for um código, maior será a produtividade de programação.

Características de LP

- Modificabilidade
  - Refere-se às facilidades de alterar o programa (em função de novos requisitos) sem que tais modificações impliquem mudanças em outras partes do programa.

- Portabilidade
  - Independência do programa gerado pela LP em funcionar de forma idêntica em diversos tipos de hardware ou sistemas operacionais.

Conclusões

- Não existe a melhor ou pior Linguagem de Programação!
- Sempre há uma Linguagem de Programação mais adequada para cada tipo de projeto.

Obrigado!