Linguagens e Compiladores

Aula 1:

a) Critérios
b) Noções de Compiladores e Interpretadores
c) Usos da tecnologia de compilação
d) Compiladores, filtros e pré-processadores
e) Estruturação lógica de compiladores
f) Exemplo
g) Bibliografia

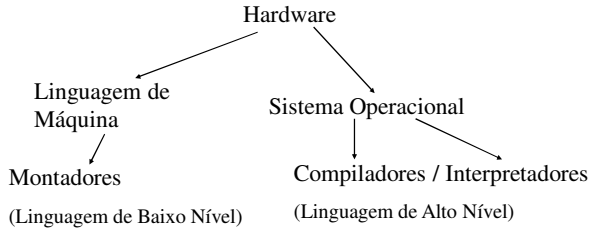
Ricardo Luis de Azevedo da Rocha

a) Critério de Aprovação

- 2 provas
- 1 trabalho prático em dupla (compilador)
- É necessário obter aprovação nas provas e também no compilador, pode-se ir para REC tanto pela média das notas como pelo compilador
- $M_F=(P_1+ 2.P_2+C)/4$, em caso de aprovação em ambos (teoria e prática), qualquer outro caso a nota será suficiente apenas para a REC
- A REC será referente às notas inferiores a 5

b) Noções de Compiladores

Programação de Sistemas



Definição

- Compilador: Um programa de computador que traduz código escrito em uma determinada linguagem (código-fonte) em código escrito em outra linguagem (linguagem-objeto).

  Exemplos:
  - Traduzir de C++ para C
  - Traduzir de Java para JVM
  - Traduzir de C para C (Por que? Para tornar o código mais eficiente, menor, mais rápido, ou mesmo medir desempenho).

c) Usos da Tecnologia de Compilação

- Uso mais comum: traduzir um programa em linguagem de alto nível para código objeto para um processador particular
- Melhorar o desempenho de programas (otimizar)
- Interpretadores: tradutores diretos “on-line”, exemplos Perl, Java
- Paralelização ou vetorização automática
- Ferramentas de depuração
- Segurança: Java VM usa compilação para demonstrar segurança de código
- Formatadores de texto (TeX para PS, PS para PDF)
- Simulação de arquiteturas, Tolerância a falhas, Assinaturas

Chave do Processo

- Habilidade de extrair as propriedades de um programa (análise), e, opcionalmente, transformá-lo (síntese). Exemplo:

```
    X = ...
    i = 1
    While (i <= 100) do
       i = i + 1
       j = i * 4
       N = j ** 2  /* Exponenciação
       Y = X * 2.0
       A[i] = X * 4.0
       B[j] = Y * N
       C[j] = N * Y * C[j]
    End While
L1: Print B[1:400]
    Print C[1:400]
    Stop
```

Resultado da Otimização de Código do Exemplo

```
    j = 4          /* (i = 1) * 4: propagação da const
    Y = X * 2.0    /* mudança do invariante do laço
    While (j <= 400) do
       j = j + 4   /* substituição da variável (forte)
       N = j * j   /* redução (forte)
                   /* eliminação de A
       B[j] = Y * N
       C[j] = B[j] * C[j] /* eliminação de sub-expressão
    End While
L1: Print B[1:400]
    Print C[1:400]
    Stop
                   /* Também aloca registros j, N, Y
```

d) Compiladores, Filtros e Pré-Processadores

- Tradutor genérico: Fonte → Objeto
- Compilador: Alto nível → Outra linguagem
- Montador: Baixo nível → Ling. máquina
- Filtro: Alto nível → Alto nível (conversões)
- Pré-Processador: Semelhante ao Filtro, porém antecede a etapa de compilação, por exemplo, expandindo macros, etc.

Formalização das Linguagens de Programação

- Objetivo: eliminar ambigüidades
- Meio: notação formal
- Característica: linguagens de programação são mais restritivas que as linguagens naturais
- Inviável tratar a comunicação homem-máquina em linguagem puramente natural
- Uso de gramáticas, reconhecedores e dispositivos geradores, descritos em uma *metalinguagem*

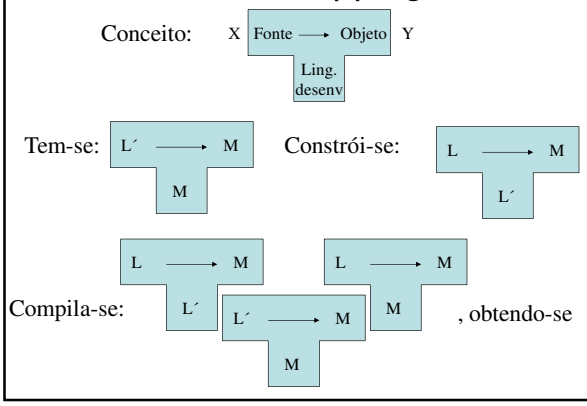
Estratégia para Obtenção de Compiladores

- Três alternativas:
  - Manual: Realizar toda a codificação através de técnicas especiais (os primeiros compiladores foram construídos desta forma)
  - Mista: Utilizar em parte algum programa de auxílio e desenvolver o restante
  - Semi-Automática: Utilizar programas mais modernos e completos, modificando pouco a estrutura gerada
- Definir a linguagem fonte de tradução e a linguagem objeto (máquina objeto), além da linguagem de desenvolvimento

Tipos de Compiladores

- Com relação à máquina hospedeira:
  - Auto-residente: se a máquina hospedeira é a mesma máquina para a qual deve ser gerado código
  - Compilador-Cruzado: se a máquina hospedeira é diferente da máquina para a qual o código deve ser gerado
- Método de "*bootstrapping*" – auto-compiláveis

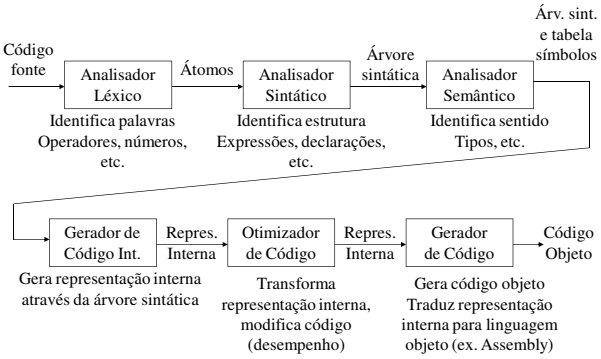
"Bootstrapping"



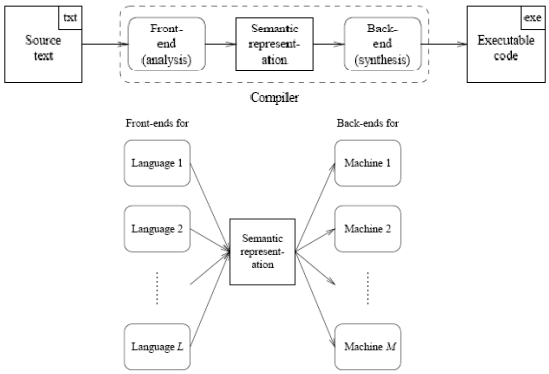
e) Estruturação Lógica de Compiladores

- Analisador léxico
- Analisador sintático
- Analisador semântico
- Gerador de código intermediário
- Otimizador de código
- Gerador de código objeto

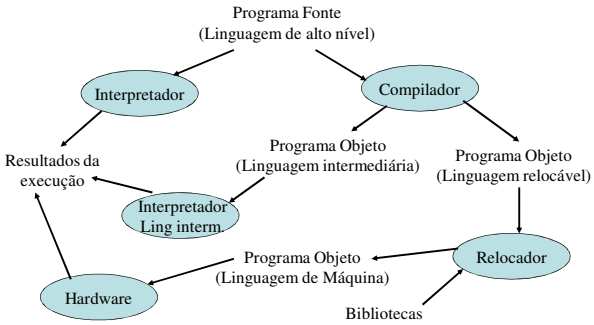
Estrutura Lógica



Front-end e *Back-end*



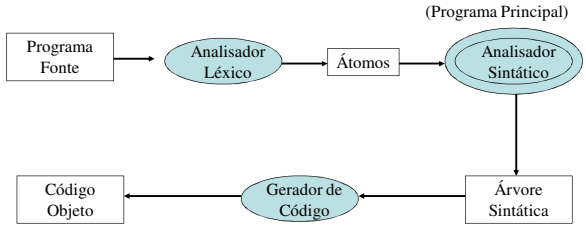
Processo de Execução de Linguagens de Alto Nível



Organização Física

- Quanto ao número de passos:
  - Em um único passo (ex. Delphi, Turbo Pascal)
  - Em múltiplos passos (ex. compiladores COBOL para Mainframes, compiladores da Microsoft mais antigos)
- Quanto à hierarquia interna
  - Dirigidos por sintaxe (mais comum)
  - Dirigidos pela análise léxica
  - Dirigidos pela geração de código

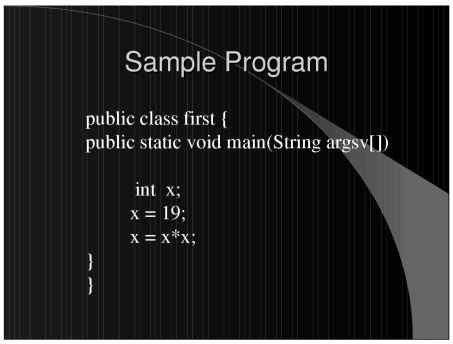
Dirigidos por Sintaxe



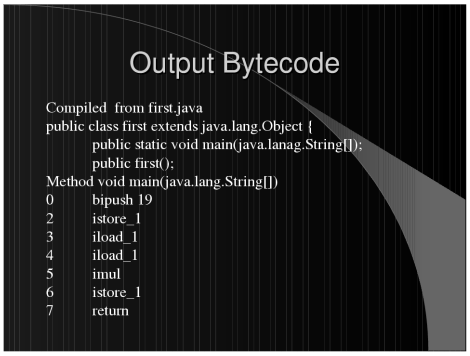
Considerações Custo-benefício

- Tarefa de compilação de uma linguagem de alto nível não é trivial
- Sem compiladores não é possível utilizar uma linguagem de alto nível (justificativa de construção)
- Número de linhas de programa geradas é constante para cada programador, e os programas ficam menores
- Número de erros diminui pela verificação dos compiladores
- Engenharia de Software, considerações de eficiência
- Dificuldade: Um compilador Pascal com otimização demanda aproximadamente 30 Homens-Ano

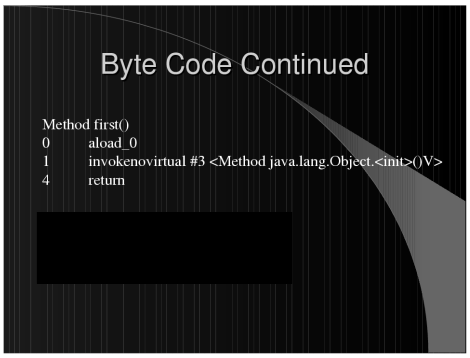
f) Exemplo



Compilação



Continuação



g) Bibliografia

- "Introdução à Compilação", J. J. Neto, LTC, 1987 (**Livro-Base**)
- "Compilers: Principles, Techniques and Tools", Alfred Aho, Ravi Sethi and Jeffrey Ullman, Addison-Wesley, 2007
- "Introdução à compilação", I. Ricarte, ed Campus, 2008
- "Projeto Moderno de Compiladores – Implementação e Aplicações", D. Grune, H. Bal, C. Jacobs, K. G. Langendoen, ed. Campus, 2002
- "Compiladores: princípios e práticas", K. C. Louden, ed. Thompson, 2004
- "Implementação de Linguagens de Programação: Compiladores", A. M. A. Price & S. S. Toscani, ed. Sagra-Luzzatto, 2002
- Trembley & Sorenson – The theory and practice of compiler writing, McGraw-Hill, 1985
- Robert W. Sebesta – Programming language concepts, Addison-Wesley, 2007.