PCS2056 – Linguagens e Compiladores

Assunto: Tradução dos comandos
5a. parte do projeto – definição das rotinas semânticas.
Entregar no dia 29/11/2016

Objetivo: Completar a especificação do código gerado pelo compilador e das rotinas do ambiente de execução da linguagem de alto nível.

Palavras-chave:
estruturas de controle de fluxo: desvios, if-then, if-then-else, decisões múltiplas, while, do-until comandos imperativos: chamadas de procedimentos, atribuição de valores, entrada e saída

Atividades de Projeto:

1) Definir detalhadamente o mapeamento de cada um dos comandos da linguagem de alto nível que você está implementando para a forma de código-objeto de nível mais baixo, a ser produzido cada vez que tais comandos forem encontrados no programa-fonte pelo compilador. Isso deve completar a especificação do código-objeto que deve ser gerado pelo compilador que está sendo construído.
2) Definir, para cada um dos comandos da linguagem de alto nível a ser compilada, as rotinas do ambiente de execução que forem necessárias para o correto funcionamento do código gerado em tempo de execução, completando dessa forma a especificação do conjunto das rotinas que compõem o ambiente de execução da linguagem implementada.
3) Verificar as especificações acima desenvolvidas em conjunto as especificações elaboradas nas aulas anteriores, testando sua consistência e compatibilidade mútua. Isso é importante para que nas próximas atividades as rotinas semânticas possam ser construídas e integradas com o reconhecedor sintático para formar o compilador desejado.

Bibliografia complementar:
Para servirem como referência para o projeto e implementação de linguagens de programação, os livros abaixo, todos relativamente recentes, enriquecem e diversificam a bibliografia clássica apresentada anteriormente e fornecem um vasto material complementar para o estudo desse assunto.

sobre linguagens de programação:
H. Ledgard e M. Marcotty – The Programming Language Landscape – SRA, 1986
R. Sebesta – Concepts of Programming Languages – Addison Wesley, 2002 (5th edition)
T. W. Pratt e M. V. Zelkowitz – Programming Languages - Design and Implementation – Prentice Hall, 1999 (3rd edition)
D. Appleby e J. J. VandeKopple – Programming Languages Paradigm and Practice – McGraw-Hill, 1997 (2nd edition)

sobre compiladores:
J. Holmes – Object-oriented compiler construction – Prentice Hall, 1995 (conceitos e teoria)
J. Holmes – Building your own compiler with C++ – Prentice Hall, 1995 (implementação)
A. W. Appel – Modern compiler implementation in Java - basic techniques – Cambridge, 1997
A. W. Appel – Modern compiler implementation in ML - basic techniques – Cambridge, 1997
A. W. Appel – Modern compiler implementation in C - basic techniques – Cambridge, 1997
J.P. Tremblay e P.G. Sorenson – The theory and practice of compiler writing – McGraw-Hill, 1985
R. Wilhelm e D. Maurer – Compiler Design – Addison Wesley, 1995

PCS2056 – Linguagens e Compiladores

Assunto: Formas intermediárias de código objeto.

Objetivo: Apresentar formas intermediárias de programas e a forma da MVN a ser usada no compilador para a disciplina.

Palavras-chave:
formas intermediárias
máquinas abstratas
MVN

Conceitos
formas intermediárias: linguagens de saída de um passo de compilação, e entrada do passo seguinte.
máquinas abstratas – hardware virtual com conjunto de instruções aderente à linguagem de alto nível
MVN – código objeto do compilador para a disciplina.

Exercícios:

1) Partindo de um reconhecedor sintático de expressões aritméticas, já construído anteriormente, acrescente rotinas semânticas que gerem como saídas as expressões equivalentes em formato MVN.
2) Codifique uma expressão relativamente longa usando a forma da MVN.
3) Procure na literatura ou na Internet descrições de máquinas abstratas importantes, como a máquina virtual da linguagem Java, a máquina P (cuja linguagem é o P-code), etc. Estude uma delas, e identifique as vantagens de sua utilização. Por que não se costuma usar algum hardware real que a interprete?