Processamento de Linguagens
MiEI (3ºano)

Trabalho Prático nº 2 (Yacc)

Ano lectivo 15/16

1 Objectivos e Organização

Este trabalho prático tem como principais **objectivos**:

- (genericamente) aumentar a experiência em *engenharia de linguagens* e em *programação generativa (gramatical)*;
- (especificamente) desenvolver processadores de linguagens segundo o método da *tradução dirigida pela sintaxe*, suportado numa gramática tradutora;
- (especificamente) desenvolver um **compilador** gerando código para uma **máquina de stack virtual**;
- (especificamente) utilizar *geradores de compiladores* baseados em *gramáticas tradutoras*, concretamente o Yacc.

e como **objectivos** secundários:

- aumentar a experiência de uso do ambiente Linux, da linguagem imperativa C (para codificação das estruturas de dados e respectivos algoritmos de manipulação), e de algumas ferramentas de apoio à programação;
- rever e aumentar a capacidade de escrever *gramáticas independentes de contexto* que satisfaçam a condição LR() usando BNF-puro.

Para o efeito, esta folha contém apenas 1 enunciado.
O programa desenvolvido será apresentado aos membros da equipa docente, totalmente pronto e a funcionar (acompanhado do respectivo relatório de desenvolvimento) e será defendido por todos os elementos do grupo, numa semana de Junho, em dia a combinar.
O **relatório** a elaborar, deve ser claro e, além do respectivo enunciado, da descrição do problema, das decisões que lideraram o desenho da linguagem/gramática e as regras de tradução para Assembly da VM (incluir as especificações Yacc), deverá conter exemplos de utilização (programas-fonte diversos e respectivo código produzido). Como é de tradição, o relatório será escrito em LaTeX.
O relatório será submetido através do sistema de submissão próprio, até à data de fim de prazo: **Sábado, 28 de Maio**.

2 Enunciado

Pretende-se que comece por definir uma linguagem de programação imperativa simples (LPIS), a seu gosto.
Apenas deve ter em consideração que a LPIS terá de permitir:

- *declarar e manusear* variáveis atómicas do tipo *inteiro*, com os quais se podem realizar as habituais operações aritméticas, relacionais e lógicas.
- *declarar e manusear* variáveis estruturadas do tipo *array (a 1 ou 2 dimensões) de inteiros*, em relação aos quais é apenas permitida a operação de indexação (índice inteiro).

- *efetuar* instruções algorítmicas básicas como a *atribuição de expressões a variáveis.*
- *ler* do *standard input* e *escrever* no *standard output.*
- *efetuar* instruções para controlo do fluxo de execução—*condicional* e *cíclica*—que possam ser aninhadas.
- *definir e invocar subprogramas* sem parâmetros mas que possam retornar um resultado atómico (opcional).

Como é da praxe neste tipo de linguagens, as variáveis deverão ser declaradas no início do programa e não pode haver re-declarações, nem utilizações sem declaração prévia. Se nada for explicitado, o valor da variável após a declaração é 0 (zero).
Nota: o desenho da LPIS tem de ser validado com o docente antes de avançar para a fase seguinte.

Desenvolva, então, um compilador para LPIS, com base na GIC criada acima e recurso ao Gerador Yacc/ Flex.
O compilador de LPIS deve gerar **pseudo-código**, Assembly da Máquina Virtual VM cuja documentação completa está disponibilizada no Bb.

Muito Importante:
Para a entrega do TP deve preparar um conjunto de testes (programas-fonte escritos na sua linguagem) e mostrar o código Assembly gerado bem como o programa a correr na máquina virtual VM. Esse conjunto terá de conter, no mínimo, os 6 exemplos que se seguem:

- lidos 3 números, escrever o maior deles.
- ler N (valor dado) números e calcular e imprimir o seu somatório.
- contar e imprimir os números pares de uma sequência de N números dados.
- ler e armazenar os elementos de um vetor de comprimento N; imprimir os valores por ordem crescente após fazer a ordenação do array por trocas diretas.
- ler e armazenar os elementos de uma matriz NxM; calcular e imprimir a média e máximo dessa matriz.
- invocar e usar num programa seu uma função 'potencia(Base,Exp)' cujo código Assembly lhe será fornecido.