UNIVERSIDADE ABERTA
www.uab.pt

Ministério da Ciência, Tecnologia e Ensino Superior

U.C. 21018

Compilação

08 de Julho de 2013

-- INSTRUÇÕES --

- O estudante deverá responder à prova na folha de ponto e preencher o cabeçalho e todos os espaços reservados à sua identificação, com letra legível.
- No fim da prova, poderá ficar na posse do enunciado.
- Verifique no momento da entrega da(s) folha(s) de ponto se todas as páginas estão rubricadas pelo vigilante. Caso necessite de mais do que uma folha de ponto, deverá numerá-las no canto superior direito.
- Em hipótese alguma serão aceites folhas de ponto dobradas ou danificadas.
- Exclui-se, para efeitos de classificação, toda e qualquer resposta apresentada em folhas de rascunho.
- Os telemóveis deverão ser desligados durante toda a prova e os objectos pessoais deixados em local próprio da sala de exame.
- Utilize unicamente tinta azul ou preta.
- A prova é constituída por **3** páginas (esta página de rosto e duas com as questões), contém 5 questões, sem consulta, e termina com a palavra **FIM**. Verifique o seu exemplar e, caso encontre alguma anomalia, dirija-se ao professor vigilante nos primeiros 15 minutos da mesma, pois qualquer reclamação sobre defeito(s) de formatação e/ou de impressão que dificultem a leitura não será aceite depois deste período.

Duração: 150 minutos

1ª Questão (4 valores)

Considere a seguinte gramática:

```
Magic -> ( Rabbit , Magic ) | ( Rabbit )
Rabbit -> Rabbit > Carrot
Carrot -> Carrot < Lettuce
Lettuce -> bunny | [ Rabbit ]
```

Construa as tabelas de acções e saltos do analisador sintáctico ascendente LR, pelo método LR Canónico.

2ª Questão (4 valores)

Apresente a especificação *flex* para um analisador léxico que identifique os tokens num documento XML:

- Tag inicial – começa por < e termina em >. No interior só pode ter letras e o hífen (-).
- Tag final – começa por </ e termina em >. No interior só pode ter letras e o híden (-).
- Entidade – Uma das seguintes:
  - `<` que representa o símbolo <
  - `>` que representa o símbolo >
  - `&` que representa o símbolo &
  - `&apos;` que representa o símbolo ‘
  - `&quot;` que representa o símbolo “

O programa deverá converter a notação XML para uma notação funcional, em que as tags correspondem a functores e o texto entre as tags iniciais e finais é o argumento da função. Além disso, as entidades correspondentes a caracteres especiais devem ser convertidas no correspondente símbolo. Por exemplo, o documento XML

```
<expr> 3 &gt; 1 </expr>
```

deverá ser convertido em:

```
expr(3 > 1)
```

3ª Questão (4 valores)

Apresente o código intermédio, em notação TAC (*three address code*) correspondente ao seguinte excerto de código em linguagem C:

```
int a[10];
int i, j;

for (i = 0; i < 10; i++)
  for (j = 0; j < 10; j++)
    a[i*10+j] = i+j*i;
```

4ª Questão (4 valores)

Apresente a especificação *bison* para um analisador sintáctico de expressões lógicas, que devolva o resultado da avaliação da expressão analisada. As constantes lógicas são identificadas pelo analisador léxico, sendo que o analisador sintáctico deverá considerar as seguintes operações:

- Conjunção: o resultado de `x e y` é **1** se `x` e `y` forem **verdadeiros**. Caso contrário é **0**.
- Disjunção: o resultado de `x ou y` é **1** se `x` ou `y` for **verdadeiro**. Caso contrário é **0**.
- Disjunção exclusiva: o resultado de `x xou y` é **0** se `x` e `y` tiverem o mesmo valor lógico. Caso contrário é **1**.

Considere ainda que o operador `e` tem precedência sobre `ou` e este sobre `xou`, e que pode usar parêntesis para alterar a ordem de avaliação.

5ª Questão (4 valores)

Optimize o seguinte código gerado em TAC (*three address code*), explicitando o tipo de optimização que está a fazer:

```
a = 4 / 2
t1 = a - 2
t2 = t1 + a
t3 = t2 - 2
t4 = t3 / 2
t5 = t3
t6 = t5 + b
c = 4 + t6
```

FIM