MFES – Verificação Formal de Software
MI/MEI

9 de Julho de 2009 – Duração: 2 horas

Teste

1. Considere o seguinte programa E da linguagem imperativa simples estudada:

```
z := y;
w := 1;
while (z >= 1) do
   w := w * x;
   z := z - 1
```

   (a) Este programa obedece à seguinte especificação:
   Pré-condição: o valor de y é um número inteiro não negativo
   Pós-condição: o valor de w é igual a x^y
   Calcule (usando o algoritmo VCGen) as condições de verificação que garantem a correcção de E face a esta especificação (terá para isso que identificar um invariante para o ciclo).

   (b) Mostre agora que o seguinte procedimento é correcto face ao seu contrato.

```
pre y >= 0
post r = x^y
proc exp(x, y) =
  z := y;
  w := 1;
  while (z >= 1) do
     w := w * x;
     z := z - 1;
  r := w
```

   (c) Utilize o VCGen sensível à segurança do programa (*safety-aware*) para provar que o programa seguinte, que invoca o procedimento anterior, é também ele correcto.

```
pre forall i; i >= 0 && i < 10 ==> valid_index(u, i);
post forall i; i >= 0 && i < 10 ==> u[i] = i^2;
proc main =
 i := 0;
 while (i < 10) do
   u[i] = call exp(i, 2);
```

2. Explique em que diferem a verificação de *correcção total* e de *correcção parcial* de programas quando se utiliza a ferramenta Frama-C.

3. Sendo ϕ uma frase em Lógica de Primeira Ordem, qual o significado de afirmação *"ϕ é válida"*? Como caracteriza o problema de decisão associado? Relacione a sua resposta com a utilização de demonstradores automáticos de teoremas.

4. Quais os aspectos distintivos entre a Lógica de Primeira Ordem e a Lógica de Ordem Superior? Indique algum tipo de propriedade que não seja possível descrever em Lógica de Primeira Ordem, mas que a Lógica de Ordem Superior consegue captar.

5. Descreva, por palavras suas, o isomorfismo de Curry-Howard que estabelece ligação entre a teoria de tipos e a lógica. Refira ainda a importância deste resultado na construção de ferramentas de prova assistida.

6. Indique qual o objecto de prova gerado pela seguinte prova em Coq.

```
Variables A B C : Prop.

Theorem ex : (A -> B -> C) -> (A -> B) -> A -> C.
Proof.
intros x y z.
apply x.
exact z.
apply y.
assumption.
Qed.
```