Exercícios - Semântica Operacional

1. Usando a semântica operacional *big-step* de ARITH
   (a) encontre o numeral n tal que $(4 + 1) + (2 * 2) \Downarrow n$
   (b) prove que $(3 + 2) \times (1 + 4) \Downarrow 25$
2. Usando a semântica Operacional *small-step* de ARITH prove que $(1 + 2) + (4 + 3) \rightarrow^* 10$. Mostre a derivação completa de cada passo da avaliação.
3. Considere a linguagem Bexp aumentada com a seguinte expressão:

$$b ::= \ldots \mid \texttt{if } b_1 \texttt{ then } b_2 \texttt{ else } b_3$$

   (a) Defina a semântica operacional *big-step* de Bexp.
   (b) Usando a sua semântica mostre a derivação completa para a seguinte expressão:

$$\begin{array}{l}\neg(\texttt{if} \quad\quad\quad\quad (\texttt{false} \wedge \texttt{false}) \\ \texttt{then} \quad (\texttt{if true then } (\texttt{false} \wedge \texttt{true}) \texttt{else false}) \\ \texttt{else} \quad\quad\quad\quad \neg\texttt{true})\end{array}$$

4. Prove as seguintes propriedades sobre a linguagem IMP:
   (a) para todo a, σ existe um único n tal que $a, \sigma \Downarrow n$
   (b) para todo b, σ existe um único t tal que $b, \sigma \Downarrow t$
   (c) para todo C, σ existe um único σ' tal que $S, \sigma \Downarrow \sigma'$
5. Use as regras da semântica operacional *small step* de IMP e construa uma derivação para o julgamento $C, \sigma \rightarrow \texttt{skip}, \sigma'$ onde C pode ser qualquer comando que precise de pelo menos 10 regras diferentes para ser executado.
6. Aumente a sintaxe de IMP incluindo o seguinte elemento em Com

$$C ::= \ldots \mid \texttt{repeat } C \texttt{ until } b$$

   Esse comando excuta C até que a condição b seja verdadeira. Aumente e/ou modifique a semântica operacional de IMP de forma a tornar precisa essa descrição informal do significado de comando `repeat`.
7. Essa é uma extensão de IMP para transferências de controle não-locais parecidas com exceções em Java e ML. A sintaxe é modificada adicionando o seguinte elemento em Com

$$C ::= \ldots \mid \texttt{try } C_1 \texttt{ catch } C_2 \mid \texttt{throw}$$

   Usualmente um comando `try` C_1 `catch` C_2 é equivalente ao comando C_1, ou seja normalmente o comando C_2 não é executado. O controle passa para o comando de *catch* C_2 somente se um comando `throw` é executado em C_1. Por exemplo, ao final da execução de comando

```
x := 1;
try
    y := 2;
    x := 2
catch
    skip
```

   o estado é tal que x e y possuem o valor 2. Já ao final de

```
x := 1;
try
    y := 2;
    throw;
    x := 2
catch
    skip
```

o estado é tal que x possui valor igual a 1 e y igual 2, e

```
x := 1;
try
    y := 2;
    throw;
    x := 2
catch
    y := 1
```

resulta em um estado no qual x e y possuem valor igual a 1.

8. A semântica original de IMP assume que todo indentificador que aparece no programa possui um valor associado na memória. Essa suposição permitiu que tivéssemos a regra (big step)

$$\frac{}{x, \sigma \Downarrow \sigma(x)} \quad \text{(VAR)}$$

que poderia ter sido escrita também na forma

$$\frac{\sigma(x) = n}{x, \sigma \Downarrow n} \quad \text{(VAR)}$$

mas onde, graças a suposição, teríamos a certeza que a premissa sempre seria verdadeira (ou seja $\sigma(x)$ sempre seria definido e igual a um inteiro).

Vamos considerar agora que isso não é verdade, ou seja, existem identificadores com valores associados e outros com valores indefinidos. A semântica precisa ser modificada/estendida para lidar com esse fato? caso positivo apresente uma proposta.

9. Modifique a sintaxe de expressões aritméticas da linguagem adicionando a seguinte produção:

$$a \quad ::= \quad \ldots \mid x{+}{+}$$

O valor produzido pela avaliação da expressão $x++$, em um dado estado s, é o mesmo pela avaliação da expressão x no estado σ. Mas a avaliação de $x++$ produz ainda, além de um valor, o efeito colateral de incrementar o valor mapeado para x na memória. Modifique/estenda a semântica da linguagem de acordo

10. Modifique a sintaxe de expressões aritméticas da linguagem adicionando a seguinte produção:

$$a \quad ::= \quad \ldots \mid {+}{+}x$$

A avaliação de $++x$ produz o efeito colateral de incrementar x. O valor de $++x$ é o valor de x no estado resultante após o incremento. Modifique/estenda a semântica da linguagem de acordo.

11. Modifique a sintaxe de expressões aritméticas da linguagem adicionando a seguinte produção:

$$a \quad ::= \quad \ldots \mid \texttt{do}\ C\ \texttt{result is}\ a$$

A expressão a é avaliada no estado resultante da execução do comando C. Modifique/estenda a semântica da linguagem de acordo.

12. Implemente a semântica operacional *big-step* e *small-step* de IMP em OCAML.