

1 Using the denotational semantics specification for constant declarations on pp. 26–27 of the notes, compute $\mathcal{E}[\text{let val } x=2*x; \text{ val } x=5 \text{ in } x]$.

2 Consider the following CFG G which generates a language of tours:

- Syntax

$$\begin{array}{lll} T & \longrightarrow & \varepsilon \mid T M ; \quad (\text{Tour}) \\ M & \longrightarrow & \text{rotate } D \mid \text{walk} \quad (\text{Move}) \\ D & \longrightarrow & \text{left} \mid \text{right} \quad (\text{Directional-Change}) \end{array}$$

Design a semantic function $\mathcal{T}: \text{Tour} \rightarrow \mathbf{Z}_{\perp} \times \mathbf{Z}_{\perp} \times \mathbf{Z}_{\perp}$ which computes the final coordinates (x, y) and direction θ in degrees of a Tour beginning at $(0,0)$ facing 0 degrees (the positive x -axis) and consisting of Moves which **rotate** the tourist **left** ($\Delta\theta = +90$ degrees) or **right** ($\Delta\theta = -90$ degrees) or **walk** the tourist forward ($r = 1$ unit). Complete the denotational semantics specification with supporting semantic functions $\mathcal{M}: \text{Move} \rightarrow ?$ and $\mathcal{D}: \text{Directional-Change} \rightarrow ?$ and semantic elements as necessary.

3 Suppose $\{P\}S\{Q\}$ and $Q \wedge \neg B \implies P$. Prove the following program specifications:

- $\{P\}$ **repeat** S **until** B $\{Q \wedge B\}$
- $\{P\}$ **begin** S ; **while** $\neg B$ **do** S **end** $\{Q \wedge B\}$

4 Prove the following program specification and its termination:

```
{n ≠ 0}
while not odd(n) do
  n := n div 2
{odd(n)}
```

5 Prove the following program specification:

```
{m ≥ 0 ∧ n ≥ 0}
begin
  SUM := 0;
  for i := 1 to m do
    for j := 1 to n do
      SUM := SUM + 1
    end
  end
{SUM = m * n}
```

6 Consider the following **for**-loop in which x , e_1 , and e_2 are of type integer; no assignments are made to x within S ; and S terminates.

```
for x := e1 to e2 do S
```

Prove termination of the **for**-loop, using the fact that it is semantically equivalent to

```
begin
  t1 := e1;
  t2 := e2;
  if t1 ≤ t2 then
    begin
      x := t1;
      S;
      while x <> t2 do
        begin
          x := x+1;
          S
        end
      end
    end
  end
end
```