Elements of Programming Languages
Tutorial 1: Abstract syntax trees, evaluation and typechecking
Solution notes

1. **Pattern matching.**

   (a)

```
def evens[A](l: List[A]): List[A] = l match {
  case Nil => Nil
  case x::Nil => List(x)
  case x::_::l2 => x::evens(l2)
}
```

   (b)

```
def allplus(e: Expr): Boolean = e match {
  case Plus(e1,e2) => allplus(e1) && allplus(e2)
  case Times(e1,e2) => false
  case Num(_) => true
}
```

   (c)

```
def consts(e: Expr): List[Int] = e match {
  case Plus(e1,e2) => consts(e1) ++ consts(e2)
  case Times(e1,e2) => consts(e1) ++ consts(e2)
  case Num(n) => List(n)
}
```

   (d) **For this example, observe that the Times case needs to recursively reverse.**

```
def revtimes(e: Expr): Expr = e match {
  case Num(n) => Num(n)
  case Plus(e1,e2) => Plus(revtimes(e1),revtimes(e2))
  case Times(e1,e2) => Times(revtimes(e2),revtimes(e1))
}
```

   (e)

```
def printExpr(e: Expr): String = e match {
  case Num(n) => n.toString
  case Plus(e1,e2) =>
    "(" + printExpr(e1) + "␣+␣" + printExpr(e2) + ")"
  case Times(e1,e2) =>
    "(" + printExpr(e1) + "␣*␣" + printExpr(e2) + ")"
}
```

2. **Evaluation derivations.**

   (a) 6×9

$$\frac{\overline{6 \Downarrow 6} \quad \overline{9 \Downarrow 9}}{6 \times 9 \Downarrow 54}$$

(b) $3 \times 3 + 4 \times 4 == 5 \times 5$

$$\dfrac{\dfrac{\dfrac{\overline{3 \Downarrow 3} \quad \overline{3 \Downarrow 3}}{3 \times 3 \Downarrow 9} \quad \dfrac{\overline{4 \Downarrow 4} \quad \overline{4 \Downarrow 4}}{4 \times 4 \Downarrow 16}}{3 \times 3 + 4 \times 4 \Downarrow 25} \quad \dfrac{\dfrac{\overline{5 \Downarrow 5} \quad \overline{5 \Downarrow 5}}{5 \times 5 \Downarrow 25}}{}}{3 \times 3 + 4 \times 4 == 5 \times 5 \Downarrow \texttt{true}}$$

(c) $\texttt{if}\ 1 + 1 == 2\ \texttt{then}\ 2 + 3\ \texttt{else}\ 2 * 3$

$$\dfrac{\dfrac{\dfrac{\overline{1 \Downarrow 1} \quad \overline{1 \Downarrow 1}}{1 + 1 \Downarrow 2} \quad \overline{2 \Downarrow 2}}{1 + 1 == 2 \Downarrow \texttt{true}} \quad \dfrac{\overline{2 \Downarrow 2} \quad \overline{3 \Downarrow 3}}{2 + 3 \Downarrow 5}}{\texttt{if}\ 1 + 1 == 2\ \texttt{then}\ 2 + 3\ \texttt{else}\ 2 * 3 \Downarrow 5}$$

(d) $(\texttt{if}\ 1 + 1 == 2\ \texttt{then}\ 3\ \texttt{else}\ 4) + 5$

$$\dfrac{\dfrac{\dfrac{\dfrac{\overline{1 \Downarrow 1} \quad \overline{1 \Downarrow 1}}{1 + 1 \Downarrow 2} \quad \overline{2 \Downarrow 2}}{1 + 1 == 2 \Downarrow \texttt{true}} \quad \overline{3 \Downarrow 3}}{\texttt{if}\ 1 + 1 == 2\ \texttt{then}\ 3\ \texttt{else}\ 4 \Downarrow 3} \quad \overline{5 \Downarrow 5}}{(\texttt{if}\ 1 + 1 == 2\ \texttt{then}\ 3\ \texttt{else}\ 4) + 5 \Downarrow 8}$$

3. **Typechecking derivations.**

(a) $\vdash 6 \times 9 : \texttt{int}$

$$\dfrac{\overline{\vdash 6 : \texttt{int}} \quad \overline{\vdash 9 : \texttt{int}}}{\vdash 6 \times 9 : \texttt{int}}$$

(b) $(\vdash \texttt{if}\ 1 + 1 == 2\ \texttt{then}\ 3\ \texttt{else}\ 4) + 5 : \texttt{int}$

$$\dfrac{\dfrac{\dfrac{\overline{\vdash 1 : \texttt{int}} \quad \overline{\vdash 1 : \texttt{int}}}{\vdash 1 + 1 : \texttt{int}} \quad \overline{\vdash 2 : \texttt{int}}}{\vdash 1 + 1 == 2 : \texttt{bool}} \quad \overline{\vdash 3 : \texttt{int}} \quad \overline{\vdash 4 : \texttt{int}}}{\vdash \texttt{if}\ 1 + 1 == 2\ \texttt{then}\ 3\ \texttt{else}\ 4 : \texttt{int}} \quad \overline{\vdash 5 : \texttt{int}}}{\vdash (\texttt{if}\ 1 + 1 == 2\ \texttt{then}\ 3\ \texttt{else}\ 4) + 5 : \texttt{int}}$$

4. **($\star$) Nondeterminism.**

(a) The properties mentioned in lecture 2 were totality (every expression evaluates to some value) and uniqueness (an expression evaluates to at most one value). The totality property still holds if we add nondeterminism, but the uniqueness property is violated.

(b) Typechecking rule should ensure that both nondeterministic choices have the required type:

$$\dfrac{\vdash e_1 : \tau \quad \vdash e_2 : \tau}{\vdash e_1 \Box e_2 : \tau}$$

(c) Possible values:

i. 3, 6, 4 and 8

ii. 1 and 2

An example is

$$\dfrac{\dfrac{\overline{\texttt{true} \Downarrow \texttt{true}} \quad \dfrac{\overline{2 \Downarrow 2} \quad \overline{2 \Downarrow 2}}{2 + 2 \Downarrow 4}}{\texttt{true} \Box \texttt{false} \Downarrow \texttt{true}}}{\texttt{if true} \Box \texttt{false then}\ 2 + 2\ \texttt{else}\ 4 \Downarrow 4}$$

$$\dfrac{\dfrac{\overline{\texttt{false} \Downarrow \texttt{false}}}{\texttt{true} \Box \texttt{false} \Downarrow \texttt{false}} \quad \overline{4 \Downarrow 4}}{\texttt{if true} \Box \texttt{false then}\ 2 + 2\ \texttt{else}\ 4 \Downarrow 4}$$

They are different derivations because the first one evaluates the "then" branch, while the second one evaluates the "else" branch.

A simpler, but valid example could be something like $(2 + 2) \Box 4$.