1. **Syntax (I).**
   The following inference rules give the syntax of a simple language.

$$\frac{}{d \in \mathsf{N}}\,(d \in \{0 \ldots 9\}) \qquad \frac{n \in \mathsf{N}}{dn \in \mathsf{N}}\,(d \in \{0 \ldots 9\}) \qquad \frac{n \in \mathsf{N}}{n \in \mathsf{Z}} \qquad \frac{z \in \mathsf{Z}}{-z \in \mathsf{Z}} \qquad \frac{z \in \mathsf{Z}}{z \in \mathsf{Q}} \qquad \frac{z \in \mathsf{Z} \quad n \in \mathsf{N}}{z.n \in \mathsf{Q}}$$

   (a) Give BNF rules for a non-terminal symbol q (and other non-terminals as you require) such that q generates all strings in Q.

   (b) Give derivations trees for the following assertions.

      (i) $42 \in \mathsf{Z}$

      (ii) $-12 \in \mathsf{Q}$

      (iii) $3.14 \in \mathsf{Q}$

2. **Syntax (II).**
   The following is the BNF grammar for a simple expression language. Give inference rules for membership in a set E, such that E contains all the strings that could be generated by non-terminal e.

$$\begin{aligned} e &::= p \mid p + e \\ p &::= a \mid a \times p \\ a &::= x \mid \texttt{(}e\texttt{)} \end{aligned}$$

3. **Syntax (III).**
   Fully parenthesize the following λ-calculus expressions.

   (a) $\lambda f.\lambda x.f\;x\;x$

   (b) $\lambda f.\lambda x.f\;(f\;x)$

   (c) $(\lambda x.\lambda y.x)\;y$

   (d) $\lambda x.\lambda y.x\;y$

4. **Evaluation (I).**
   Consider the following simple arithmetic language:

$$\begin{aligned} &z \in \mathbb{Z} \\ &t ::= z \mid t + t \mid t \times t \mid \textsf{ifeven}\ t\ \textsf{then}\ t\ \textsf{else}\ t \end{aligned}$$

   The following rules give an evaluation relation for that language, assuming 4-bit unsigned numbers.

$$\frac{}{z \Downarrow n}\,(z \equiv n \bmod 16) \qquad \frac{t_1 \Downarrow n_1 \quad t_2 \Downarrow n_2}{t_1 + t_2 \Downarrow n_3}\,(n_1 + n_2 \equiv n_3 \bmod 16) \qquad \frac{t_1 \Downarrow n_1 \quad t_2 \Downarrow n_2}{t_1 \times t_2 \Downarrow n_3}\,(n_1 \times n_2 \equiv n_3 \bmod 16)$$

$$\frac{t_1 \Downarrow n_1 \quad t_2 \Downarrow n_2}{\textsf{ifeven}\ t_1\ \textsf{then}\ t_2\ \textsf{else}\ t_3 \Downarrow n_2}\,(n_1 \equiv 0 \bmod 2) \qquad \frac{t_1 \Downarrow n_1 \quad t_3 \Downarrow n_3}{\textsf{ifeven}\ t_1\ \textsf{then}\ t_2\ \textsf{else}\ t_3 \Downarrow n_3}\,(n_1 \equiv 1 \bmod 2)$$

   Give derivation trees for the evaluation of the following expressions.

   (a) $(4 \times 3) + 4$

   (b) $(6 \times 6) + 4$

(c) ifeven $4 + 3$ then 6×5 else 3×8

5. **Evaluation (II).**
Assume the language from the previous question. We want to develop a new relation $\Downarrow_{\mathsf{p}}$ which characterizes whether the result of evaluating an expression is even (E) or odd (O) (or, possibly, may be either). The first rules for this relation are as follows:

$$\frac{}{z \Downarrow_{\mathsf{p}} \{\mathsf{E}\}}\ (z \equiv 0 \bmod 2) \qquad \frac{}{z \Downarrow_{\mathsf{p}} \{\mathsf{O}\}}\ (z \equiv 1 \bmod 2)$$

$$\frac{t_1 \Downarrow_{\mathsf{p}} S_1 \quad t_2 \Downarrow_{\mathsf{p}} S_2}{t_1 + t_2 \Downarrow_{\mathsf{p}} \bigcup\{s_1 \hat{+} s_2 \mid s_1 \in S_1, s_2 \in S_2\}} \quad \text{where} \quad \begin{array}{c|cc} \hat{+} & \mathsf{E} & \mathsf{O} \\ \hline \mathsf{E} & \mathsf{E} & \mathsf{O} \\ \mathsf{O} & \mathsf{O} & \mathsf{E} \end{array}$$

(a) Give the evaluation rule for $t_1 \times t_2$.

(b) Give the evaluation rule for ifeven t_1 then t_2 else t_3

6. **Evaluation (III).**
Suppose we extend our simple language with variables, function abstraction and application:

$$t ::= \cdots \mid x \mid \lambda x.t \mid t\ t$$

In *call-by-value* evaluation, the argument to a function is evaluated before it is called. In *call-by-name* evaluation, in contrast, the argument to a function is not evaluated before evaluating the function. The difference is illustrated in the evaluation rules below; call-by-value is on the left, and call-by-name is on the right.

$$\frac{t_1 \Downarrow_{\mathsf{cbv}} \lambda x.t \quad t_2 \Downarrow_{\mathsf{cbv}} w \quad t[w/x] \Downarrow_{\mathsf{cbv}} v}{t_1\ t_2 \Downarrow_{\mathsf{cbv}} v} \quad \frac{t_1 \Downarrow_{\mathsf{cbn}} \lambda x.t \quad t[t_2/x] \Downarrow v}{t_1\ t_2 \Downarrow_{\mathsf{cbv}} v}$$

Given the following definition:

$$spin = (\lambda f.f\ f)\ (\lambda f.f\ f)$$

write the result of evaluating each of the following definitions under call-by-name and call-by-value interpretations, or write "diverge" if they diverge (i.e., run forever).

Expression	cbn	cbv
ifeven 1 then $spin$ else 0		
ifeven $spin$ then 4 else 0		
$(\lambda x.\lambda y.x)\ 4\ spin$	4	diverge
$(\lambda x.\lambda y.y)\ 4\ spin$		

7. **Fixed points.**
Suppose we extend our language with a *fixed point* construct to capture recursive definition:

$$t ::= \cdots \mid \mathsf{fix}\ t$$

with the evaluation rule:

$$\frac{}{\mathsf{fix}\, t \Downarrow \lambda x.t\,(\mathsf{fix}\, t)\, x}$$

Rewrite the following recursive definitions to used the fixed point construct instead.

(a) $add = \lambda m.\lambda n.\mathsf{if}\ m = 0\ \mathsf{then}\ n\ \mathsf{else}\ incr\,(add\,(m-1)\,n)$

(b) $even = \lambda m.\mathsf{if}\ m = 0\ \mathsf{then}\ True\ \mathsf{else}\ \mathsf{if}\ m = 1\ \mathsf{then}\ False\ \mathsf{else}\ \neg(even\,(m-1))$