Final sol.

Q1 (a) There are several problems due to common prefixes; e.g.,

$I \to CI \mid C$: $\quad a \in FIRST(CI) \cap FIRST(C)$

(b) They are resolved using the common prefix procedure:

$I \to CI \mid C$ replaced by $I \to CJ \quad J \to I \mid \varepsilon$

$N \to DN \mid D$ —"— $N \to DM \quad M \to N \mid \varepsilon$

$T_s \to T \mid T, T_s$ —"— $T_s \to TZ \quad Z \to \varepsilon \mid , T_s$

There is one less obvious common prefix:

$T \to A \mid S \qquad S \to A(T_s)$

replaced by: $T \to AY \qquad Y \to \varepsilon \mid (T_s)$

Q2 The "mark-and-sweep" technique uses the existing links in the heap. Therefore we cannot use a balanced tree.

(a) 2^{40} – in the worst case, the entire heap is a singly linked list

(b) As explained above, we cannot control the size of the stack. So, "pointer-reversal" is a very effective way to reduce the memory required.

Q3

$T \equiv \lambda xy.x$

$F \equiv \lambda xy.y$

$if \equiv \lambda cte.cte$

$cons \equiv \lambda xyf.fxy$

$car \equiv \lambda l.lT$

$cdr \equiv \lambda l.lF$

$null? \equiv \lambda l.l\,(\lambda xy.F)$

$nil \equiv \lambda l.T$

$\boxed{\begin{array}{l} T \equiv True \\ F \equiv False \end{array}}$

(a) $cons\ A\ nil$

$$\left.\begin{array}{l} \equiv (\lambda xyf.fxy)\ A\ (\lambda l.T) \\ \Rightarrow_\beta (\lambda yf.fAy)(\lambda l.T) \\ \Rightarrow_\beta \lambda f.fA(\lambda l.T) \end{array}\right] (1)$$

$$\left.\begin{array}{l} T\ A\ B \equiv (\lambda xy.x)\ A\ B \\ \Rightarrow_\beta (\lambda y.A)\ B \\ \Rightarrow_\beta A \end{array}\right] (2)$$

$$\left.\begin{array}{l} F\ A\ B \equiv (\lambda xy.y)\ A\ B \\ \Rightarrow_\beta (\lambda y.y)\ B \\ \Rightarrow_\beta B \end{array}\right] (3)$$

$car\ (cons\ A\ nil)$

$\overset{(1)}{\Rightarrow^*_\beta} (\lambda l.lT)\ (\lambda f.fA(\lambda l.T))$

$\Rightarrow_\beta (\lambda f.fA(\lambda l.T))\ T$

$\Rightarrow_\beta T\ A\ (\lambda l.T)$

$\overset{(2)}{\Rightarrow^*_\beta} A$

(b) $cdr\ (cons\ A\ nil)$

$\overset{(1)}{\Rightarrow^*_\beta} (\lambda l.lF)(\lambda f.fA(\lambda l.T))$

$\Rightarrow_\beta (\lambda f.fA(\lambda l.T))\ F$

$\Rightarrow_\beta F\ A\ (\lambda l.T)$

$\overset{(3)}{\Rightarrow^*_\beta} \lambda l.T$

(c) $if\ (null?\ nil)\ T\ F$

$\equiv if\ ((\lambda l.l\,(\lambda xy.F))\ (\lambda l.T))\ T\ F$

$\Rightarrow_\beta if\ ((\lambda l.T)(\lambda xy.F))\ T\ F$

$\Rightarrow_\beta if\ T\ T\ F$

$\equiv (\lambda cte.cte)\ T\ T\ F$

$\Rightarrow_\beta (\lambda te.Tte)\ T\ F$

$\overset{(2)}{\Rightarrow^*_\beta} \underbrace{(\lambda te.t)}_{T}\ T\ F$

$\equiv T\ T\ F$

$\overset{(2)}{\Rightarrow^*_\beta} T$

Q4

(a)

```
(define bin-strict-and
   (lambda (x y)
      (let ((xt (if x #t #f)) (yt (if y #t #f)))
         (and xt yt))))
```

(b)

```
(and #f (2 2))  => #f
(bin-strict-and #f (2 2))  => error ; 2 not a procedure
```

(c)

```
(define strict-and
   (lambda l
      (let x ((l1 l))
         (if (null? l1)
             #t
             (and (x (cdr l1)) (car l1))))))
```

Q5 a

$$\forall X \forall Y \forall M \forall F \big(brothers(X,Y) \vee \neg mother(M,X) \vee \neg mother(M,Y) \vee \neg father(F,X) \vee \neg father(F,Y) \vee \neg boy(X) \vee \neg boy(Y)\big)$$

b $$\forall X \forall Y \Big(\big(brothers(X,Y) \leftarrow \exists M \exists F \big(mother(M,X) \wedge mother(M,Y) \wedge father(F,X) \wedge father(F,Y) \wedge boy(X) \wedge boy(Y)\big)\big)\Big)$$

Q6

(a)

```
1. member3(H,[H|T]) :- not(member3(H,T)).
2. member3(H,[_|T]) :- member3(H,T).
3. not(X) :- X, !, fail.
4. not(_).
```

(b) put: m = member3