PART 1 (12 points)

Q1. (2)

```
1
2
3
```

Q2. (2)

```
(1 (2) 3)
```

Q3. (2)

```
```

Q4. (2)

```
```

Q5. (2)

```
```

Q6. (1)

```
```

If you wish to write a note on any problem, put a check in the box next to the problem number, and use the space provided at the end of the answer sheet. Notes written outside the answer boxes on this page will be ignored.
PART 2 (22 points)

Q9. (5) 
(map empty-unit
  (subunits unit))

Q10. (5) 
(lambda (sub sofar)
  (+ (unit-adults sub) sofar))

Q11. (2) 
0

Q12. (2) 
new-subunits

Q13. (5) 
(lambda (x)
  (not (null? x)))

Q14. (3) 
(map (lambda (x)
  (big-units x cutoff))
  (subunits unit))

PART 3 (16 points)

Q15. (3) 
Values: number
Types: 

Q16. (3) 
Values: compound proc
Types: boolean, number -> number

Q17. (2) 
Values: error, 20 not applicable
Types: 

Q18. (3) 
Values: compound proc
Types: anytype -> number

Q19. (3) 
Values: error, wrong number arguments to "three"
Types: 

Q20. (2) 
Values: compound proc
Types: number -> string

Note: this is a hard problem!
Not only should we include big subunits at the current level,
but we need to recurse deeper
to get big units further down
in the tree.
PART 4 (30 points)

Q21.(7)  
(if (null? lst)
    nil
  (append (reverse (cdr lst))
    (list (car lst)))))

Q22.(3)  
(cons (cadv lst)
  (cons (car lst)
    (caddr lst))))

Q23.(10)  
(if (= n o)
    lst
  (swap-first-and-second
    (cons (car lst)
      (nth-to-front
        (cadv lst)
        (- n 1))))))

Q24.(10)  
(if (= n (length lst))
  lst
  (helper (nth-to-front
    lst
    n)
    (+ n 1)))

PART 5 (20 points)

Q25.(2)  
C

Q26.(2)  
A

Q27.(2)  
C

Q28.(2)  
C

Q29.(3)  
D

Q30.(3)  
C

Q31.(3)  
E

Q32.(3)  
C