J Bolton Ch. 14

**Programming Languages: Principles and Paradigms**

14.1, 14.2, 14.6, *14.14 (in Haskell only), *14.32-14.34

**1.**

(a).

```
((\x . x*x)5)
(5 * 5)
25
```

(b).

```
((\y . ((\x . x + y + z)3))2)
\y . (3+y+z)2
3+2+z
```

(c).

```
((\v . (\w . w))((\x . x)(y(\z .z))
((\v . (\w . w))y(\z .z)
(\w . w)(\z . z)
```

**2.**

Same results except example c, where the outer functions are not applied so nothing happens in the inner functions.

**6.**

```
(m-expression '(plus (times (variable a) (variable b))
                     (value 2))
              '((a 2) (b4)))
= applyBinary '(plus (times (variable a) (variable b))
                     (value 2))
              '((a 2) (b4)))
= (+ (m-expression '(times (variable a) (variable b)) '((a 2) (b 4)))
     (m-expression '(value 2) ((a 2)(b 4))
= (+ (* (m-expression (variable a) '((a 2) (b 4))) (m-expression (variable b)) '((a 2) (b 4)))
     (m-expression '(value 2) ((a 2)(b 4))
= (+ (* (get 'a '((a 2) (b 4))) etc
= (+ (* 2 (get 'b '((a 2) (b4))) etc
= (+ (* 2 4)
  (m-expression '(value 2) ((a 2)(b 4))
= (+ (* 2 4) 2)
= (+ 8 2)
= 10
```