Esteban Serna
Coms 342
HW4

1. ASCII Values

```
(define a 97)
(define A (- a 32))
(define b 98)
(define B (- b 32))
(define c 99)
(define C (- c 32))
```

2. Right Angel

```
(define isRightAngled
        (lamda (B L H)
          (if (= (+ (* BB)(* LL)(* HH)))
                #t
           #f))
```

3.

a. Factorial

```
(define factorial
    (lambda (n)
      (if (= n 1)
        1
        (+ ( factorial (- n 1)n))))
```

b. Fibonacci

```
(define fib
        (lambda (n)
          (if(= n1)
            1
           (if ( = n 2 )
            1
            (+ (fib (- n 1))(fib (- n 2))  )))))
```

4.

a.

```
(define max
    (lambda (n)
        (if (null? n)
          0
        (if (null ? (car list )) (car list)
          (if (> (car list)(max(cdr list)))(car list)
                (max(cdr list))  )))))
```

Esteban Serna
Coms 342
HW4

b.

```
(define helper
        (lambda (n)
          (if ( = n 0 )
            #t
           (if ( = n 1 )
            #f
            (helper(- n 2)  ) ) ) )

(define even
        (lambda (n)
        (if (null? n ) (list)
        (if (helper(car  n ))(cons (car n)(even(cdr x))  ) ) ) )
```

5.

a.

```
(define helper
        (lambda ((a b ) (list x y)) )
(define pairs
        (list ( pair( 5 1 ) ( 4 6 ) ( 8 7 ) (10 15)) )
```

b.

```
(define firstSum
        (lambda (list n )
          (if  (null? n )
            0
           (+(car(car n)) (firstSum (cdr n)) )))
```

6.

a.
```
(define filter
        (lambda(test_op lst)
          (if(null? lst) (list)
            (if(test_op (car lst))
                (cons(car lst)(filter test_op (cdr lst)))
                (filter test_op (cdr lst)) ))))
```

b.
```
(define foldl
  (lambda (op zero_ele lst)
    (if (null? lst)
      zero_ele
      (foldl op (op (car lst )  zero_ele) (cdr lst)))))
```

Esteban Serna
Coms 342
HW4

7.

a.
```
(define pair
   (lambda ((x y)
        (list x y ) )))
(define pairs
        ( pair ( 1 3)(4 2 )( 5 6) )
```

b.
```
(define nth
   (lambda (op lst x )
        (if (< x 0) -1
        (if (null? lst) -1
        (if (= n 0)
                (op(car(car(lst)) (car(cdr(car lst))))
                (nth op (cdr lst)( x 1) )))))
```

c. sad