Ejercicios de Racket

1) (**EVALUADO**) Se tiene una cuenta de ahorros en la cual se deposita un capital C al inicio de un periodo de un año. Por cada año que transcurra se recibe un interés I sobre el capital depositado. Es decir, al terminar el año se cuenta con el capital inicial más interés ganado. Construya una función `(int Cap, I, N)` que calcula el monto que se recibe al depositar un capital a un interés dado durante un cierto número de años. Debe producir los resultados de acuerdo con el siguiente cuadro:

Capital	Interés	Años	Resultado
2000	0.10	0	2000
2000	0.10	1	2200
2000	0.10	2	2420
2000	0.10	3	2662

2) Escriba un programa que determine la suma de los dígitos de un número entero positivo N mayor que 0. Sugerencia: utilizar las funciones de `quotient` y `remainder`. A continuación se presentan algunos ejemplos:

```
> (sumd 124)
7

> (sum 12480)
15
```

3) Construya una función que se llame `(duplicar E L)`. Esta función debe sustituir cada uno de los elementos de la lista L por el elemento E. A continuación algunos ejemplos.

```
> (duplicar 'a '(1 3 4))
(a a a)

> (duplicar 'a '())
#f
```

4) Construya una función que se llame `(doblar E L)`. Esta función debe duplicar cualquier parición del elemento E en la lista L. Por ejemplo:

```
> (doblar 'a '(b c d a e f a))
(b c d a a e f a a)

> (duplicar 'es '(hoy es jueves))
(hoy es es jueves)
```

5) Construya utilizando la función *map*, una función que se llame `(primeros L)`. Esta función recibe en L una lista de listas y debe devolver una lista con el primer elemento de cada una de ellas. Por ejemplo:

```
> (primeros '((1 2 3)(1 2 3)))
(1 1)

> (primeros '((a b c)(d e f)(g h i)))
(a d g)
```

Como complemento a esta función construya otra que se llame `segundos`, con el mismo funcionamiento, solamente que con los segundos elementos de cada lista

6) (**EVALUADO**) Construya una función que se llame `merge`. Esta función recibe dos listas ordenadas y produce otra lista ordenada con todos los elementos de las primeras dos listas A continuación se presentan algunos ejemplos

```
> (merge '(1 2 3 4) '(5 6 7 8))
(1 2 3 4 5 6 7 8)

> (merge '(1 2 3) '(1 2 3 4))
(1 1 2 2 3 3 4)
```

7) Construya una función que se llame `permutaciones` esta recibe una lista de enteros y devuelve todos los posibles ordenamientos (permutaciones) de esa lista. Por ejemplo:

```
> (permutaciones '(1 2 3))
((1 2 3)(1 3 2)(2 1 3)(2 3 1)(3 1 2)(3 2 1))
```

8) (**EVALUADO**) Construya una función que se llame `sub-conjunto?` Esta función recibe dos argumentos y debe producir un valor #t cuando el primer argumento es subconjunto del segundo y #f en caso contrario. Por ejemplo:

```
> (sub-conjunto? '() '(a b c d e f))
#t

> (sub-conjunto? '(a b c) '(a b c d e f))
#t

> (sub-conjunto? '(a b x) '(a b c d e f))
#f
```

9) (**EVALUADO**) Utilice la función *map* para definir la función *eliminar_elemento* que recibe un elemento E y una lista L y retorna la lista L sin el elemento E. Si el elemento no existe, retorna la lista L original.

```
> (eliminar_elemento 3 '(1 2 3 4 5))
(1 2 4 5)

> (eliminar_elemento 0 '(1 2 3 4 5))
(1 2 3 4 5)
```