UNIVERSIDAD NACIONAL DE RÍO CUARTO
FACULTAD DE CS. EXACTAS, FCO-QCAS Y NATURALES
DEPARTAMENTO DE COMPUTACIÓN
ASIGNATURA: **INTRODUCCIÓN A LA ALGORÍTMICA Y PROGRAMACIÓN**
Año: 2018.

Práctico Nº 10

Tema: Recursividad. **Duración**: 4 clases

Ej. 1.a) Defina una función recursiva que reciba como parámetro un número natural y devuelva la sumatoria de los números desde el uno hasta el número dado incluido.

1.b) Defina una función recursiva que reciba como parámetro un número natural y devuelva la productoria de los números pares desde el uno hasta el número dado incluido.

1.c) Defina una función recursiva que reciba como parámetro un número natural y retorne como resultado el mismo número expresado en el sistema binario.

1.d) Defina una acción recursiva que reciba como parámetro un número natural como dato y otro como resultado y devuelva, en este último, la sumatoria de los números desde el uno hasta el número dado incluido.

1.e) Defina una acción recursiva que reciba como parámetro un número natural como dato y otro como resultado y devuelva, en éste último, la productoria de los números pares desde el uno hasta el número dado incluido.

Ej. 2) Dada una lista simplemente encadenada del siguiente tipo:

tipo TElemento = <nro ∈ Z, sig ∈ puntero a TElemento>

escribir las siguientes funciones o acciones recursivas:

2.a) ***LongElem***, función que dado un puntero al primer elemento de la lista retorne la cantidad de elementos, es decir la longitud de la misma.

2.b) ***Suma***, función que dado un puntero al primer elemento de la lista retorne la suma de los elementos de la misma.

2.c) ***MasNum***, acción que dado un número entero y un puntero al primer elemento de la lista, modifica la lista original sumándole a cada elemento de la lista el número entero pasado como parámetro.

Ej. 3.a) Defina una función recursiva que reciba como parámetro una LSE de números enteros y retorne como resultado la cantidad de números pares que contiene.

3.b) Idem pero con una acción recursiva (agregando un parámetro para el resultado).

Ej. 4.a) Defina una acción recursiva que reciba como parámetro un arreglo de hasta 30 números enteros, y la cantidad de valores cargados; y retorne en una variable, a llamar ***suma***, la sumatoria de todos los números contenidos en el arreglo.

4.b) Defina una acción recursiva que reciba como parámetro un arreglo de hasta 30 números enteros y la cantidad de valores cargados; y retorne en una variable a llamar Producto, el producto de todos números contenidos en el arreglo, entre la posición 1 y n, siendo n un parámetro que se pasa y que puede tomar el valor 1 hasta 30 como máximo.

Ej. 5) Usando una función recursiva que reciba como parámetro una LSE de cadenas de caracteres y retorne como resultado Verdadero si existe en ella un nombre pasado como parámetro en la variable ***nom***, en otro caso que devuelva Falso.

Ej. 6.a) Defina una función recursiva que reciba como parámetro un arreglo de enteros y la cantidad de valores que contiene, y retorne el promedio de los números contenidos en el arreglo.

6.b) Resuelva el mismo problema pero desarrollando una acción recursiva.

Ej. 7) Defina una función recursiva que reciba como parámetros dos listas simplemente encadenadas (LSE) de enteros y retorne como resultado la suma de todos los elementos de las dos listas.

Ej. 8) Suponiendo que solo se tiene definidas sobre los números naturales las funciones primitivas *pred* y *suc* como :

Suc(x) = x +1

Pred(x)= x – 1 si x>0 y *Pred*(x)= 0 si no es >0

Defina recursivamente las siguientes funciones sobre los números naturales:

8.a) *Suma* NxN→N {la suma de naturales, es decir *Suma*(x,y)=x+y}

8.b) *Mult* NxN→N {el producto de naturales, es decir *Mult*(x,y)=x*y}

8.cc) *Monus* NxN→N {definida como *Monus*(x,y)= x - y, si x>y *Monus*(x,y)=0 si no(x<y) }

Ej. 9) Defina una acción recursiva que reciba como parámetro la LSE llamada Original de números enteros y una lista vacía a denominar Destino, y retorne como resultado la Destino con todos los elementos de la primer lista (Original) de números enteros:

9.a) que queden en el mismo orden.

9.b) que queden en orden inverso.

Ej. 10) Defina una acción recursiva que reciba como parámetro un arreglo de 30 números enteros y la cantidad de valores cargados; y retorne en una variable a llamar ***pares*** el valor Verdadero si todos los números contenidos en el arreglo son pares sino que retorne Falso.

Ej. 11) Defina una acción recursiva que reciba como parámetro un arreglo de 30 caracteres y la cantidad de valores cargados; y retorne en una variable a llamar ***contA*** la cantidad de letras “a” que hay en el arreglo.

Ej. 12) Defina una acción recursiva que reciba como parámetro un número natural *n*, si *n* es par la acción debe dar por salida todos los números pares comprendidos entre *n* y 0, y si *n* es impar debe dar por salida todos los números impares entre *n* y 0.

Ej. 13.a) Desarrollar una función recursiva que reciba como parámetro un arreglo de 30 caracteres y la cantidad de valores cargados; y retorne cuantas palabras comienzan con la letra “m”. Para esto tenga en consideración que cada elemento de un arreglo es solo una letra y las palabras comienzan con espacio en blanco (ej.: ~~b~~**La~~b~~casa~~b~~de~~b~~la~~b~~montaña~~b~~azul**, en esta cadena, donde cada carácter podría ser cada uno un elemento de un arreglo de caracteres, el símbolo ~~b~~ representa a un espacio en blanco. Para identificar si una palabra empieza con “m”, es necesario ver si antes de la “m” hay un espacio en blanco (~~b~~), o si la primera letra del arreglo es una “m”.

13.b) Generalice la función anterior para que la letra pueda entrarse como parámetro.

Ej. 14.a) Desarrollar una función recursiva que reciba como parámetro un arreglo de 30 caracteres y la cantidad de valores cargados; y retorne cuantas palabras comienzan con la sílaba “ca”. Para esto tenga en consideración que cada elemento de un arreglo es solo una letra y las palabras comienzan con espacio en blanco (ej.: ~~b~~**La~~b~~casa~~b~~de~~b~~la~~b~~montaña~~b~~azul**, en esta cadena, donde cada carácter podría ser cada uno un elemento de un arreglo de caracteres, el símbolo ~~b~~ representa a un espacio en blanco. Para identificar si una palabra empieza con “ca”, es necesario ver si antes de la “c” hay un espacio en blanco (~~b~~), o si la primera sílaba del arreglo es una “ca”).

14.b) Generalice la función anterior para que la sílaba pueda entrarse como parámetro.

Ej.15) Dada una estructura simplemente encadenada del siguiente tipo:

TElemento = <nro ∈ Z, sig ∈ puntero a TElemento>

Desarrolla las siguientes funciones o acciones recursivas:

15.a) *Mayor*, función que dado un puntero al primer elemento de la estructura retorne el mayor valor de la misma.

15.b) *Cuad*, acción que dado un puntero al primer elemento de la estructura da por la salida el cuadrado de cada número de la misma.

15.c) *MasMayor* acción que dado un puntero al primer elemento de la estructura modifica la estructura original sumándole a cada elemento el mayor número de la estructura.

Ej. 16) Dada las siguientes funciones, resueltas como **recursión en aumento**, hallar para cada caso una solución de **recursión en cola**.

16.a)
```
Función factorial (dato n ∈ N) → N
{Def: (n0=0 ∧ fact(n0)=1) ∨ (n0>0 ∧ fact(n0)=1*2*..*n0)}
Inicio
  según
    n=0: ← 1
    n>0: ← n * factorial(n-1)
 fsegún
Fin
```

16.b)
```
TLista = puntero a TElem

TElem = <info ∈ m, next ∈ puntero a TElem>

Función long (dato L ∈ TLista) → N
Inicio
 según
   L=nil: ←  0
   L≠nil: ←  1 + long((^L).next)
  fsegún
Fin
```

16.c)
```
Función contOcu (dato a ∈ arreglo [1..254] de Carácter, u ∈ (0..254), c ∈ carácter) → (0..254)
{Def: ((contOcu(a,u,c,res)=0 ∧ el arreglo a esta vacío o no existe ningún carácter igual a c) ∨
(contOcu(a,u,c,res)<>0 ∧ existe uno o más caracteres en a que son iguales a c))  ∧ c=Co }
Inicio
  según
    u=0: ← 0
    a[u]=c: ← 1+ contOcu(a, u-1,c)
   a[u]<>c: ← contOcu(a, u-1,c)
  fsegún
Fin
```

Plan de Clases

Clase 1: 1.a), 1.c), 2.b), 3.a),
Clase 2: 4.a), 6.a), 8.a), 8.b)
Clase 3: 10), 11), 13.a)
Clase 4: 15.a), 16.a), 16.b), 16.c)

Pasar a C el Ejercicio 13.b). Fecha de entrega: **en la fecha que indique el profesor a cargo de los Trabajos Prácticos.**