ANÁLISIS Y DISEÑO DE ALGORITMOS I

Práctico Nº 3. Tipos de Datos Abstractos

1. Para cada uno de los siguientes Tipos de Datos Abstractos:

- Escriba una especificación algebraica en lenguaje *Nereus*.
- Clasifique sus operaciones según su signatura en constructoras básicas, constructoras modificadoras u observadoras.
- Escriba algunos términos del álgebra.
- Reescriba los términos del inciso anterior en función de las constructoras básicas.
- Para los incisos a, b, f y g implemente en C++. Para los incisos restantes discuta las distintas alternativas de implementación propuestas en cada inciso en base a la complejidad temporal de sus funciones. Seleccione una de ellas e implemente en C++.

a) TDAs *Persona*, *Libro.*

b) TDAs *Punto*, *Segmento* y *Círculo.*

c) *TDAs conjunto*, *lista*, *pila* y *cola*.
Posibles representaciones:
- arreglo
- listas vinculadas

d) *TDA árbol binario de Búsqueda*.
Posibles representaciones:
- arreglo
- punteros (hijo izquierdo, hijo derecho).

e) *TDA árbol n-ario*.
Posibles representaciones:
- arreglo con punteros al padre
- arreglo de punteros a hijos,
- hijo más a la izquierda, hermano derecho.
- punteros a listas de hijos (listas vinculadas)

f) *TDA heap*.
Implemente utilizando la representación basada en arreglo.

g) TDA *Polinomio* de una variable con coeficientes enteros,
$p(x) = a_0 + a_1 x + ... + a_{n-1} x^{n-1} + a_n x^n$, $n \in N$, $\forall i : 0 \leq i \leq n: a_i \in Z$,
con las siguientes operaciones:
- *cero* : representa el polinomio $p(x) = 0$
- *añadir*: añade un término (coeficiente-exponente) a un polinomio
- *evaluar*: calcula el valor del polinomio en un punto
- *coeficiente*: devuelve el coeficiente asociado a un término de exponente dado dentro de un polinomio
- *suma:* dado dos polinomios, devuelve la suma de los mismos.

2. Especifique en lenguaje *Nereus* e implemente en C++ el tipo de dato *Cadena de Caracteres*, que además de la funcionalidad básica (crear, agregar, insertar en una posición dada, longitud, es_vacía), ofrezca el siguiente conjunto de funciones:

- *concatenar:* concatena dos cadenas de caracteres,
- *duplicar:* duplica una secuencia elemento a elemento, por ejemplo, duplicar (abc) = aabbcc,
- *iguales:* chequea si dos secuencias son iguales elemento a elemento,
- *reversa:* dada una secuencia devuelve su reversa,
- *capicúa:* chequea si una secuencia es capicúa, por ejemplo: capicúa(aba) = true,
- *repeticiones:* dada una secuencia y un elemento, devuelve la cantidad de veces que aparece del elemento en la secuencia.

3. Una inmobiliaria necesita organizar la información sobre los inmuebles que dispone para la venta. Una forma de organizarlos sería mantener una colección de inmuebles que permita:

- agregar un nuevo inmueble,
- dado el identificador de un inmueble eliminar el inmueble correspondiente de la colección,
- seleccionar un inmueble cuyo precio sea menor o igual a un precio dado,
- seleccionar todos los inmuebles cuyo precio sea menor o igual a un precio dado,
- seleccionar todos los inmuebles de un barrio dado,
- consultar si existen inmuebles a la venta en un barrio dado,
- etc.

Se dispone de la siguiente información sobre cada inmueble: identificador, precio, barrio y descripción.

a) Especifique en *Nereus* el TDA *Inmueble* y el TDA *Colección de Inmuebles*.

b) Codifique en C++ los TDA especificados en el punto a.