TAREA #1 (Conceptos, Definición de TAD y Lenguaje C) Entregar vía Educa hasta el Jueves 22/Feb/2017 23:55 hs. Tarea grupal. Total de puntos: 30

Se entrega en un solo archivo comprimido (zip o rar).
Cada ejercicio en un subdirectorio separado. Se entrega solo los fuentes (.c y .h). En todos los ejercicios incluya el nombre de los integrantes del grupo.

1. Construya prototipos para los siguientes TAD (Tipo Abstracto de Datos) indicando la forma de implementación. Para este ejercicio diseñe la forma en que el dato será almacenado internamente e implemente, si es el caso, las operaciones indicadas *(20p)*

   - **Conjunto.** Implementar las operaciones de *pertenencia, tamaño y unión.* Los elementos deben ser del tipo cadena que a su vez debe ser diseñado como TAD. Asegurarse que no debe existir un límite al número de elementos del conjunto, solo el definido por la cantidad de memoria disponible. Recordar el concepto de Conjunto (elementos sin repetir y sin orden).

   - **Diseñar el TAD que represente Listas del lenguaje Scheme.** Escriba la estructura de datos y las posibles operaciones posibles. Las operaciones deben estar indicadas, no hace falta implementar, aunque deben estar bien documentadas.

     Entre las operaciones incluya e implemente *ConcatAll* que retorna los elementos de la lista que sean del tipo cadena (char *) como una cadena con esos elementos concatenados. (*OPCIONAL – 5 pts*)

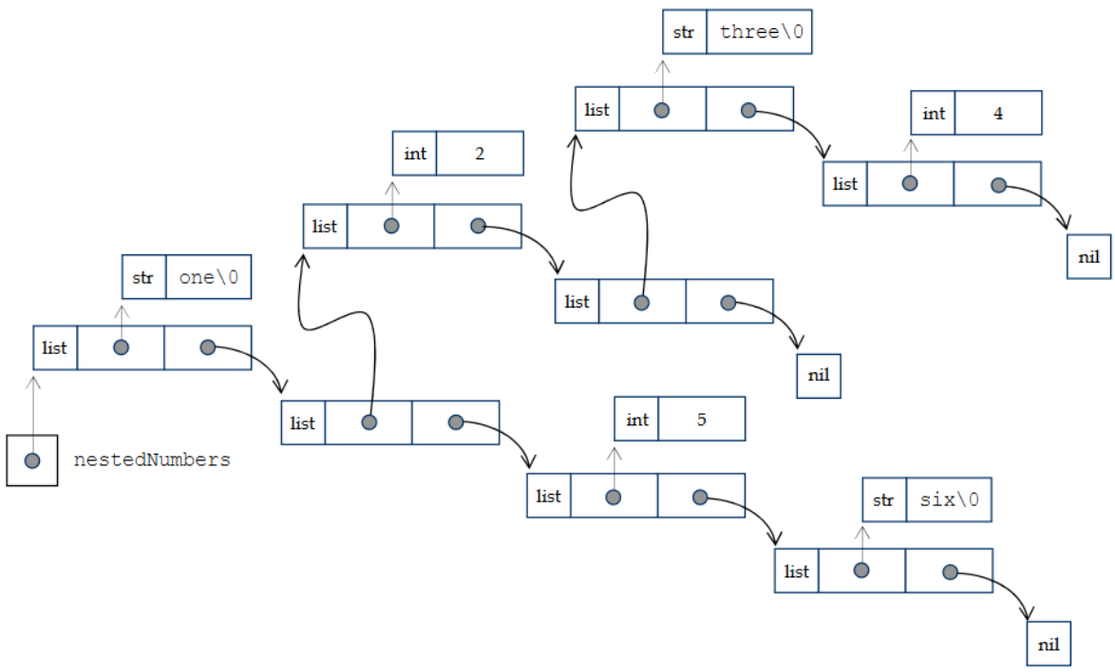
     Scheme es un lenguaje cuya estructura de dato básica es la lista enlazada. Estas listas son heterogéneas, es decir no se necesita de que sean del mismo tipo.

     Algunos ejemplos de listas en Scheme:

     ```
     ( 2 3 4 7 )
     ( "Hola" "como" "estan" )
     ( "luque" 2  "cerro" 2 )
     (("hola que") ( 1 3 (4 5) ))
     ```

     Fijarse que pueden existir listas de listas.

     Una representación de la lista `("one" ( 2 ( "three" 4 ) ) 5 "six" )` donde el nombre de la variable es *nestedNumbers* sería:



     Considerar los tipos *int*, *str* (char *) , *list*, nil.

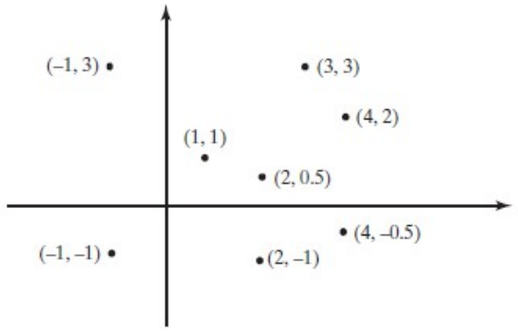
Utilice C para la implementación. Si utiliza funciones preincorporadas del lenguaje, solo use las que son estándar.

Incluya pruebas de su implementación (comentadas adecuadamente) en la función main y en un archivo fuente separado.

En su código indique la forma de compilar su programa y verificar su implementación. ***Incluir los comentarios en el propio código fuente.***

2. Resolver el siguiente problema en lenguaje C *(10p) (Entregar solo los fuentes,* ***incluir los comentarios en el propio fuente****).*

Dado un conjunto de puntos, se desea encontrar los dos puntos que son los más cercanos uno a otro, es decir el par más cercano. En la figura de abajo, el punto **(1,1)** y el punto **(2, 0.5)** son los más cercanos.



	x	y
0	–1	3
1	–1	–1
2	1	1
3	2	0.5
4	2	–1
5	3	3
6	4	2
7	4	–0.5

La distancia entre dos puntos (x_1,y_1) (x_2,y_2) es calculada utilizando la fórmula usual de la distancia Euclidiana

$$\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

Considere el siguiente algoritmo recursivo para solucionar el problema:

Paso 1: Ordenar los puntos en forma ascendente por la coordenada x. Para los puntos con la misma coordenada x, ordenar por la coordenada y. Esto genera una lista de S puntos.

Paso 2: Dividir S en dos listas. S_1 y S_2, de igual tamaño utilizando el punto medio de la lista S. Recursivamente encontrar el par más cercano en S_1 y S_2. Dado d_1 y d_2 ser las distancias de los pares más cercanos en las dos sublistas, respectivamente.

Paso 3: Encontrar el par más cercano entre S_1 y S_2, siendo esta distancia d_3. El par más cercano es uno con distancia $min(d_1,d_2,d_3)$.

Observaciones:

- La solución debe basarse en la idea de algoritmo explicada más arriba.
- La lista de puntos se lee de la entrada estándard. No debe imponer ninguna cantidad fija de puntos.
- Utilizar los TAD adecuados para implementar la solución.
- Se tendrá en cuanta el correcto uso de las estructura de datos y de la aplicación de los conceptos de TAD.
- Su programa debe se hecho en C estándar (debe poder compilarse en Linux, Windows, MacOS, etc)
- Se corregirá el adecuado uso del lenguaje y de estilo de programación