Apellido y Nombre: _____

Carrera: _____ DNI: _____

[Llenar con letra mayúscula de imprenta GRANDE]

Universidad Nacional del Litoral
Facultad de Ingeniería y Ciencias Hídricas
Departamento de Informática

Algoritmos y Estructuras de Datos

Algoritmos y Estructuras de Datos. 3er Parcial. Tema: **2B.** [8 de julio de 2003]

[Ej. 1] **[Primitivas (20 puntos)]** Escribir las funciones del `TAD COLA DE PRIORIDAD` listadas a continuación, implementado por montículos: `ANULA (C)`, `INSERTA (x,C);`, `SUPRIME_MIN (C)`. **Escribir todos los tipos, definiciones, funciones y procedimientos auxiliares necesarios.**

[Ej. 2] **[Programación (total = 60 puntos)]**

(a) **[MergeSort-Colas (30 puntos)]** Escribir un procedimiento `procedure MERGESORT (var C:cola);` que ordena los elementos de una cola `C` usando el método de clasificación por intercalación. Es decir, divide la cola en dos subcolas auxiliares `C1` y `C2` de tamaño similar, las ordena aplicando `MERGESORT` recursivamente y luego intercala las subcolas:

- dividir `C` en dos subcolas `C1`, `C2`
- clasificar `C1` y `C2` por `MERGESORT`
- intercalar `C1` y `C2` en `C`

Usar las primitivas del `TAD COLA`: `ANULA (C)`, `PONE_EN_COLA (x,C)`, `QUITA_DE_COLA (C)`, `VACIA (C)`, y `FRENTE_DE_COLA (C)` que considere necesarias.

(b) **[Burbuja para listas (30 puntos)]** Escribir un procedimiento `procedure BURBUJA(var L:lista);` que ordena los elementos de una lista `L` mediante el método de clasificación por burbujas. Como en listas simplemente enlazadas no es conveniente avanzar hacia el principio de la lista por una cuestión de eficiencia, el algoritmo de burbuja se modifica de manera de que los elementos más grandes viajan hacia el fin. Para arreglos, sería

```
for  i := n downto 2 do
   for  j := 1 to (i-1) do
      if  A[j].clave > A[j+1].clave then
         intercambia (A[j].clave, A[j+1].clave);
```

Usar las primitivas del `TAD LISTA`: `RECUPERA (p,L)`, `SIGUIENTE (p,L)`, `PRIMERO (L)`, y `FIN (L)` que considere necesarias.

[Ej. 3] **[Operativos (total = 20 puntos)]**

(a) **[Quick-sort (10 ptos)]** Dados los enteros $\{5, 9, 1, 2, 6, 7, 11, 2, 7, 11, 12, 8\}$ ordenarlos por el método de *"clasficación rápida"* (*"quick-sort"*). En cada iteración indicar el pivote y mostrar el resultado de la partición.

(b) **[Heap-sort (10 ptos)]** Dados los enteros $\{4, 8, 11, 5, 6, 3, 13, 7\}$ ordenarlos por el método de *"montículos"* (*"heap-sort"*). Mostrar el montículo (minimal) antes y después de **cada** inserción/supresión.