Departamento de Ciencias de la Computación e I.A.
Universidad de Granada
DECSAI

Fundamentos de Programación

Convocatoria de Febrero. Curso 2011/2012
14 de Febrero de 2012

1. (**2.5 puntos**) Defina una función o método **recursivo** para comprobar si una cadena de caracteres es un palíndromo, es decir, que se lee de izquierda a derecha igual que de derecha a izquierda. No se tendrán en cuenta los espacios en blanco, ni se distinguirá entre letra mayúscula y minúscula. Por ejemplo, la siguiente cadena es un palíndromo:

   ```
   (a,h, , ,B, ,c, ,C,b, ,h, ,A)
   ```

   Para resolver este problema, el alumno puede usar el tipo de dato que considere oportuno para almacenar una cadena de caracteres, y puede resolverlo tanto con una función como con un método de una clase, con las siguientes condiciones:

   - El algoritmo **debe ser recursivo**.
   - No se pueden usar bucles ni modificar la cadena de caracteres inicial.

2. (**3 puntos**) Considere un **conjunto** de enteros como una secuencia **ordenada** de números enteros **sin repetidos**. Se desea crear un programa que lea dos conjuntos de enteros y escriba el resultado de calcular la unión y la intersección de ambos conjuntos.

   a) Escriba un módulo que calcule la *unión* de dos conjuntos. Recuerde que en la unión se deben incluir los elementos que estén en cualquiera de ellos.

   b) Escriba un módulo que calcule la *intersección* de dos conjuntos. Recuerde que en la intersección se deben incluir los elementos que sean comunes a ambos conjuntos.

   c) Escriba una función `main` como ejemplo de prueba de ambos módulos. En esta prueba, deberá leer dos conjuntos desde la entrada estándar y escribir el resultado de la unión y la intersección.

   Notas:

   i) Debe implementar cuantas funciones y/o métodos requiera para la correcta resolución del problema.
   ii) No está permitido el uso de ningún algoritmo de ordenación.
   iii) Se valorará la eficiencia de los algoritmos implementados.
   iv) Ambas funciones/métodos tienen como precondición que los conjuntos son correctos, es decir, ya están ordenados y sin repetidos. El resultado debe ser igualmente correcto.
   v) Las funciones/métodos de unión e intersección no contendrán ninguna operación de E/S.

3. Diseñamos la clase **Sopa** para poder manejar *sopas de letras*. Un objeto de la clase **Sopa** almacena $F \times C$ caracteres organizados en F filas por C columnas. Por tanto, es posible recorrer secuencias de caracteres en las 8 direcciones correspondientes a la horizontal, vertical y las dos diagonales. Resuelva las siguientes cuestiones:

   a) (**1 punto**) Defina la representación de la clase. Indique brevemente cómo representa la clase y escriba el código C++ correspondiente.

   b) (**1.5 puntos**) Defina un método *"Traspuesta"* de la clase **Sopa** que nos sirva para obtener una nueva sopa de letras que contiene las mismas palabras. En concreto, tendrá un tamaño de $C \times F$ caracteres de forma que la fila (columna) i-ésima de la nueva sopa corresponderá a la columna (fila) i-ésima de la sopa original.

   c) (**2 puntos**) Defina un método *"Diagonal"* de la clase **Sopa**. Este método tiene como entrada una cadena de caracteres con la palabra a buscar, y obtiene como resultado si se encuenta en alguna de las diagonales, así como la localización de la palabra buscada. Observe que,

   - es posible que la palabra no esté en la sopa, y
   - el método busca la solución sólo en las dos diagonales, es decir, puede encontrarla en 4 direcciones: primera diagonal (abajo-derecha o arriba-izquierda), segunda diagonal (arriba-derecha o abajo-izquierda).

   Importante: Las soluciones pueden incluir llamadas a otros métodos de la clase, en cuyo caso, será **obligatorio** incluir la implementación de dichos métodos. Además, observe que no es necesario leer/escribir ningún dato en la entrada/salida estándar.