Teoría de la Computación

Práctica No. 1: **Manipulación de cadenas**

Objetivo de la práctica

Implementar algunas operaciones básicas de cadenas de la teoría de la computación y practicar la manipulación de cadenas.

Fecha límite de entrega

Martes 8 de Marzo de 2011.

Descripción del problema

El alumno deberá implementar una aplicación que tome como entrada dos cadenas de caracteres, llamadas *a* y *b*. Su aplicación deberá realizar las siguientes operaciones sobre ellas, según un menú de opciones:

- Leer una cadena (*a* ó *b*).
- Mostrar el alfabeto mínimo de una cadena (letras que conforman una cadena).
- Calcular la longitud de una cadena.
- Determinar si ambas cadenas son iguales o no.
- Calcular la concatenación de ambas cadenas (*ab* ó *ba*).
- Determinar la longitud de la concatenación de ambas cadenas.
- Calcular la *n*-ésima potencia para ambas cadenas. El valor de *n* también es ingresado por el usuario y puede comprender un valor mayor ó igual a 0.

El programa deberá ser capaz de procesar palabras vacías.

Lineamientos generales para la realización de su solución

1. **Diseño de la implementación**. Deberá implementar las siguientes funciones, las cuales podrán ser manipuladas desde la función `main`:

   a) Una función de nombre `length` que calcule la longitud de una cadena. Recibe como argumento una cadena y retorna como resultado su longitud:

   ```
   int length(char *);
   ```

   b) Una función de nombre `equals` que determine si dos cadenas son iguales. Recibe como argumento dos cadenas y retorna falso ó verdadero, según sea el resultado de la evaluación.

   ```
   bool equals(char *, char *);
   ```

   c) Una función de nombre `alpha` que muestre en pantalla cada una de las letras que forman parte de una cadena. A este conjunto de letras le llamaremos el *alfabeto mínimo* de la cadena. Esta función recibe como argumento una cadena de caracteres. No genera ningún valor de retorno.

   ```
   void alpha(char *);
   ```

   d) Una función de nombre `cat` para concatenar cadenas. Recibe como argumento dos cadenas y regresa como resultado una nueva cadena, la cual contiene el resultado de la concatenación de las dos anteriores. Esta nueva cadena deberá ser generada dinámicamente.

   ```
   char *cat(char *, char *);
   ```

e) Una función de nombre `pow` que calcula la *n*-ésima potencia de una cadena, para algún valor no negativo de *n*. Recibe como argumento una cadena de caracteres y un valor entero (el valor de *n*). Regresa como resultado la *n*-ésima potencia de la cadena proporcionada. La cadena de resultado deberá de ser generada dinámicamente.

```
char *pow(char *, int);
```

Atienda a los siguientes requerimientos:

- Su programa deberá almacenar cada una de sus cadenas en arreglos de caracteres. No use la clase `string` de C++ para este fin.
- El resultado de las funciones `cat` y `pow` deberá ser almacenado en un nuevo arreglo de caracteres que deberá ser creado dinámicamente (en tiempo de ejecución), con el fin de contar con espacio suficiente para almacenar el resultado de la operación correspondiente. No utilice funciones de la librería estándar para resolver las concatenaciones.
- El cálculo de la longitud de una cadena deberá hacerse sobre tales arreglos cada vez que se necesite, usando para ello la función `length`. No utilice funciones de la librería estándar para tal fin.
- El cálculo de la igualdad de cadenas también deberá realizarse sobre tales arreglos, usando para ello la función `equals`. No utilice funciones de la librería estándar para tal fin.

Se exhorta a hacer uso de la implementación dinámica de sus arreglos (use las funciones `malloc` y `free` del lenguaje `C` ó los operadores `new` y `delete` de C++) para evitar restricciones impuestas por el tamaño de un arreglo al momento de almacenar sus resultados. Procure atender la convención de C/C++ de señalar el final de una cadena mediante el carácter nulo (`'\0'`). El uso de esta marca facilitará la implementación de sus operaciones.

2. **Lenguaje de implementación**. Se exhorta al uso del lenguaje ANSI C/C++ como lenguaje de desarrollo. Atienda además los siguientes puntos:

   a) El uso de interfaces gráficas y/o entornos de desarrollo será opcional, y su uso se considerará responsabilidad del alumno. Cualquier inconveniente originado por el uso de ellos que impida la revisión será causa de la anulación de la misma.
   b) No podrá hacer uso de herramientas auxiliares ó librerías especiales que suplanten las técnicas requeridas para el desarrollo de su implementación. Cualquier solución alterna deberá ser señalada y justificada, y no deberá alterar en ningún caso los aspectos a evaluar. El no apego a este punto causará la anulación de su proyecto.

3. **Asesoría para su elaboración**. El alumno podrá en todo momento asesorarse con el profesor durante el desarrollo de su proyecto, empleando para ello los horarios de laboratorio destinados a este fin.

Requerimientos para la evaluación de la práctica

Los siguientes requerimientos serán solicitados para la evaluación de esta práctica:

1. **Verificación de la aplicación**. Se hará una corrida de prueba, durante la cual se proporcionarán dos cadenas arbitrarias al programa. Se hará la verificación de los resultados de acuerdo a lo expuesto en la descripción del problema. Además se verificará que la implementación de sus funciones así como el almacenamiento de sus cadenas ocurra de acuerdo a los lineamientos presentados.

2. **Comprensión del problema**. El alumno deberá tener una clara comprensión teórica y técnica del problema, la cual podrá ejercitar durante el desarrollo de esta práctica. Este punto será evaluado mediante una breve indagación que el profesor hará durante la verificación de la aplicación así como la revisión del código fuente.

3. **Documentación del proyecto**. Deberá elaborarse un reporte de práctica de acuerdo a lo que se expo-

ne en el *Manual para la elaboración de los Reportes de Práctica*. Haga énfasis en los algoritmos usados para la implementación de sus funciones, así como en la definición teórica de sus operaciones. La entrega de la documentación impresa será un requisito necesario para dar inicio a la revisión de su proyecto.

Ejemplo

Suponga que se proporciona una entrada similar a la siguiente:

```
Cadena A: hola
Cadena B: mundo
Potencia: 3
```

Entonces su programa debería generar una salida similar a la siguiente:

```
Alfabeto de la cadena A: [a,h,l,o]
Longitud de la cadena A: 4

Alfabeto de la cadena B: [d,m,n,o,u]
Longitud de la cadena B: 5

Las cadenas A y B son diferentes

La concatenacion de A y B: holamundo
La longitud de la concatenacion: 9

La potencia 3 de la cadena A: holaholahola
La potencia 3 de la cadena B: mundomundomundo

Fin de la aplicación
```

Suponga ahora que alguna de las dos cadenas es vacía (ambas cadenas pueden ser vacías), o bien que se solicita calcular la potencia 0:

```
Cadena A:
Cadena B: mandala
Potencia: 0
```

Entonces debería generarse una salida similar a la siguiente:

```
Alfabeto de la cadena A: []
Longitud de la cadena A: 0

Alfabeto de la cadena B: [a,d,l,m,n]
Longitud de la cadena B: 7

Las cadenas A y B son diferentes

La concatenacion de A y B: mandala
La longitud de la concatenacion: 7

La potencia 0 de la cadena A:
La potencia 0 de la cadena B:

Fin de la aplicación
```