****

***Laboratorio 6***

***Manejo de cadenas de caracteres***

**Objetivos:**

* Conocer el funcionamiento de las cadenas.
* Conocer las distintas operaciones permitidas con las cadenas.

**Introducción**

**Definición de String**

Una cadena de caracteres o string es una serie de caracteres cuya longitud (número de caracteres que contiene) puede variar de 1 hasta 255 caracteres. Una variable de cadena está declarada para la palabra string seguida de la longitud máxima de la cadena encerrada entre corchetes.

**Declaración de una variable tipo string:**

Las variables de cadena se pueden declarar las de cualquier otro tipo, ya sea en la sección var o en type.

Si se declara como var, se hace de la siguiente manera:

**Var**

Nombre\_completo : string [80];

Ejemplo: string [40];

Si la cadena es declarada como type, haríamos la declaración así:

**Type**

Cadena80 = string [80];

Cadena40 = string [40];

**Var**

Mensaje : Cadena80;

Nombre : Cadena40;

Es importante recordar que si se declara como Type inmediatamente después, dentro de las variables, se debe declarar una variable que haga referencia al Type. Una vez declaradas las variables de cadena se pueden realizar asignaciones o bien operaciones de lectura / escritura en los programas.

Por ejemplo:

Program Ejemplo;

Uses crt;

Var

Mensaje: string[40];

Begin

Mensaje:= 'Todo esfuerzo trae su recompensa.';

Write('Mi mensaje de hoy para ustedes es: ', Mensaje);

Readkey;

End.

**Longitud de una cadena**

Una cadena físicamente es una estructura de una secuencia de 0 hasta 255 caracteres de longitud. Sin embargo, la ocupación en memoria de una cadena es un número de bytes igual al de caracteres de la cadena más uno.

Por ejemplo, la cadena: `Programador´ tiene una longitud de 11 caracteres, entonces ocupará en memoria 12 bytes. Esto se debe a que el primer byte en una cadena, contiene la longitud de la cadena actualmente almacenada en memoria.

Supongamos que esta palabra corresponde a un dato almacenado en la variable PUESTO que fue definida de la siguiente manera:

**Var**

Puesto: String[20];

Sucede que, entonces, la longitud lógica de la variable será de 11 bytes pero la longitud física será de 20 bytes pues así fue definida.

Dependiendo del tamaño (longitud de la cadena) de las cadenas fuente y destino se pueden presentar tres casos diferentes que analizaremos a partir de la siguiente declaración de variables de cadena.

Var

Cad1 : String [8];

Cad2 : String [11];

Cad3 : String [15];

Supongamos que se ejecuta la sentencia Cad2 := ‘informatica’;

**Caso 1 :** Este caso no presenta problemas, pues cada posición de cad2 se rellena con un carácter.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **I** | **N** | **F** | **O** | **R** | **M** | **A** | **T** | **I** | **C** | **A** |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |

**Caso 2 :** Supongamos que ahora se ejecuta la sentencia Cad3 := Cad2;

Resultará que Cad3 tiene 15 caracteres y sólo hay 11 caracteres en cad2. La cadena Cad3 seguirá teniendo una longitud máxima de 15 caracteres, pero sólo se almacenan en ella ahora 11 caracteres y su longitud actual será de 11.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **I** | **N** | **F** | | **O** | **R** | **M** | **A** | **T** | **I** | **C** | **A** |  |  |  |  |
| 1 | 2 | | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

**Caso 3**: Supongamos que se trata de ejecutar ahora Cad1 := Cad2;

En este caso, como Cad1 tiene una longitud de ocho caracteres, la cadena destino no se puede almacenar totalmente. En este caso se trunca la cadena destino y se almacenan en Cad1 los primeros ocho caracteres (de izquierda a derecha) de Cad2.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **I** | **N** | **F** | **O** | **R** | **M** | **A** | **T** |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

**Comparación de cadenas**

La comparación de cadenas es una operación muy común; estas comparaciones se realizan con base en el orden del código ASCII, por ejemplo la cadena 'Prueba' es menor a la cadena 'prueba' ya que el valor del código ASCII de la “P” es 80 y el de la “p” es 112.

**Cadena nula o vacía**

Existe un caso especial de una cadena, que es cuando no tiene ningún carácter en ella, en ese momento decimos que es una cadena nula o vacía. Se recomienda ampliamente inicializar todos los valores de las cadenas a cadenas nulas al inicio del programa, ya que podrían contener datos extraños e indeseables.

**Manejo de los elementos de la cadena**

Es posible leer cada elemento de una cadena por separado, por ejemplo, si en la variable Calle almacenamos la cadena 'Hidalgo' es posible asignar a otra cadena el valor de Calle[x] donde x es el carácter de la cadena que se quiere leer, así también es posible visualizar el carácter en pantalla usando la instrucción WriteLn. Ejemplo:

**Program** caracter;

Var

Calle : string;

Letra : string;

Begin

Calle := 'hidalgo';

Writeln(calle[2]); {visualiza el segundo carácter del contenido de la variable calle}

Letra := calle[1]; {guarda en letra el primer carácter de la variable calle}

Writeln(Letra)

end.

**Longitud de una cadena**

Se utiliza la función Length, la longitud es la cantidad de caracteres que contiene la cadena en un momento determinado. La función regresa un valor entero. En este caso el programa imprime en la pantalla un mensaje con la longitud de la cadena almacenada en memoria, en este caso específico 18.

**Program** longitud;

Var

Cadena : string;

Begin

Cadena := 'prueba de longitud';

Writeln ('longitud de la cadena: ', length (cadena));

End.

**Operador +**

Es una de las formas más sencillas de unir dos cadenas y se maneja exactamente como una suma, la única limitante es que en caso de ser mayor la longitud resultante de la suma que la longitud que pueda manejar la variable en que se almacenará se truncarán los caracteres sobrantes. Ejemplo:

Program union;

uses crt;

Var

cadena1,cadena2 , cadena3: string;

Begin

Cadena1:= 'Buenos';

Cadena2:= 'dias';

Cadena3:= Cadena1 + Cadena2;

Writeln (‘La unión es : ', cadena3);

**Concat**

La función concat produce los mismos resultados que la concatenación con el “+”, es posible incluir cualquier número de cadenas que se necesiten concatenar. La suma de las cadenas no deberá sobrepasar la longitud que la variable, a la cual se asignará dicha suma, puede manejar. Ejemplo:

**Cadena := Concat (Cad1, cad2, cad3);**

**Pos**

La función Pos nos sirve para localizar una determinada cadena dentro de otra, en otras palabras para verificar si una cadena es subcadena de otra segunda.

Los parámetros que requiere son: la cadena que se buscará y la cadena donde se buscará la primera: Pos (Cadena1, Cadena2); Cuando encuentra la cadena la función devuelve su posición inicial, en caso de que no la encuentre devuelve el valor de 0.

Ejemplo:

**Program** find;

uses crt;

**Var**

cadena: string;

Begin

Cadena := 'Domingo Lunes Martes Miércoles Jueves Viernes Sábado';

**WriteLn**(Cadena);

**WriteLn**(Pos('Lunes', Cadena)); {Muestra 9}

**WriteLn**(Pos('Jueves', Cadena)); {Muestra 32}

**WriteLn**(Pos('Ayer', Cadena)); {Muestra 0} readkey;

End.

**Copy**

La función copy regresa una subcadena de una variable o constante dada como parámetro. Su sintaxis es: **Copy (Cadena, Inicio, Cantidad);** Donde **Cadena** es la fuente de la subcadena, **Inicio** es la posición desde la cual se obtendrá la subcadena y **Cantidad** es el número de caracteres que se extraerán. Los valores de Inicio y Cantidad deben ser de tipo entero. **Ejemplo:**

**Program** copiar;

uses crt;

**Var**

cadena: string;

Begin

Cadena := 'Nuevos horizontes';

WriteLn(Copy(Cadena, 8, 10)); {Visualiza: horizontes}

readkey;

End.

**Insert**: Sirve para insertar una cadena dentro de otra en alguna posición determinada. Donde **Cadena1** es la que se insertará, **Cadena2** es donde se insertará y **Posición** es el lugar donde se insertará. El parámetro Posición debe ser de tipo entero. **Insert (Cadena1, Cadena2, Posición).**

**Program** insertar;

**uses** crt;

**Var**

cadena1,cadena2: string;

**Begin**

Cadena2:= 'Hola chicos queda un mes ';

Cadena1:='de clases';

insert(cadena1, cadena2, 26); {Visualiza: Hola chicos queda un mes de clases }

writeln(cadena2);

readkey;

**End.**

**Delete:** Este procedimiento elimina un determinado número de caracteres de una cadena. Donde **Cadena** es la variable en la cual se eliminarán los caracteres, **Inicio** es la posición del primer carácter a eliminar y **Número** es la cantidad de caracteres que se borrarán.

**Delete (Cadena, Inicio, Número)**

Program borrar;

uses crt;

Var

cadena:string;

caracter:char;

longitud,i:integer;

Begin

writeln('Introduzca una frase');

readln(cadena);

writeln('Introduzca un caracter');

readln(caracter);

longitud:=length(cadena);

for i:=1 to longitud do

begin

if cadena[i]=caracter then

delete(cadena,i,length(caracter));

end;

writeln('La cadena nueva es ', cadena);

readkey;

End.

**UpCase:** La función UpCase regresa la letra mayúscula correspondiente al carácter dado como parámetro. Es muy común trabajando con cadenas que se quiera convertir una que contiene un número a su valor numérico, para poder utilizarlo en operaciones matemáticas, así como convertir un número a su equivalente en una cadena.

**Laboratorio**

1. Escriba un programa que, dado una cadena de caracteres, indique la cantidad de veces que se repite cada vocal y lo represente en un histograma. Ejemplo :
   1. a 15 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
   2. e 10 \*\*\*\*\*\*\*\*\*\*

UTILICE SUBPROGRAMAS.

1. Escribir un programa en Pascal que genere la inversa de una cadena de caracteres.
2. Escribir un programa en Pascal que elimine los blancos de una cadena de caracteres.
3. Escribir un programa en Pascal que encripte una cadena de caracteres sumando 2 al código ASCII de cada uno de sus caracteres.

**Post Laboratorio**

**Valor de cada ejercicio ( 5 ptos c/u)**

1. Diseñar un editor de texto que pueda ejecutar sobre un texto leído las siguientes operaciones :

* Determinar la longitud del texto.
* Borrar una subcadena
* Localizar una cadena especificada
* Insertar una subcadena en una posición especificada
* Sustituir una cadena por otra.
* Convertir a mayúsculas o minúsculas

**2.** Escriba un programa que reciba una cadena de caracteres y determine si esta cadena es palíndrome. Una cadena es palíndrome si se lee igual de derecha a izquierda y de izquierda a derecha. Ejemplo : "Isaac no ronca así"

**3.** Hacer un programa que lea un párrafo terminado en punto y encuentre todas las palabras que tiene tres o más vocales distintas.

4.-Escribir un programa que lea una de cadenas de caracteres de longitud arbitraria e imprima la longitud de la cadena, cuente el número de ocurrencias de palabras de cuatro letras, sustituya cada palabra de cuatro letras por una cadena de cuatro asteriscos e imprima la nueva cadena.

**Instrucciones para el envío de la tarea :**

Crear una carpeta comprimida que incluya:

* Un archivo .doc con los algoritmos de los problemas propuesto en el Postlaboratorio y con las capturas de pantalla de las corridas de los programas respectivamente.
* Los archivos .pas de los problemas resuelto (un archivo .pas por cada programa)

Además debe enviar la HOJA DE CONTROL DE PRACTICAS con la autoevaluación de la práctica de la semana.

Adjuntar archivos a la tarea correspondiente en Módulo 7