

Strings

Problem 1: (20%) Escriba un programa que realice las siguientes operaciones sobre cadenas de caracteres (strings); no usar memoria dinámica:

- a) **int longitud_str(str):** Función que regrese la longitud de un string **str** (sin contar el caracter nulo);
- b) **char *copia_str(dst, src):** copia el string **src** a **dst**, incluyendo el caracter nulo `\0`; regresa un apuntador a **dst**. **Nota:** La función debe verificar que los strings no estén superpuestos y si hay suficiente espacio en **dst** para copiar **src**; en caso que no, incluir solo hasta donde sea posible almacenar en **dst**;
- c) **int compara_str(str1, str2):** Función que concatene tres strings (en el orden dado) separados por un espacio y los almacene en **str1**; debe regresar la nueva longitud de **str1**;
- d) **int concatena_str(str1, str2, str3):** Función que concatene tres strings (en el orden dado) separados por un espacio y almacene en **str1**; debe regresar la nueva longitud de **str1**; **Nota:** La función debe verificar que los strings no estén superpuestos y si hay suficiente espacio en **str1** para almacenar todos los strings; en caso que no, incluir solo hasta donde sea posible almacenar en **str1**;
- e) **int encuentra_str(str1, str2):** Función que busque un sub-string **str1** en **str2** y regrese el número de veces que lo encuentra.
- f) **int **frecuencia(str):** Encuentre la frecuencia de cada elemento de **str**, y regrese un arreglo bidimensional donde se almacene la letra (1er columna) y frecuencia (2da columna). **Salida:** w — 1, r — 5, s — 4, etc... **Nota:** En este ejercicio puede usar memoria estática o dinámica.
- g) **char *sin_repetir(str):** Función que encuentre las palabras en **str** que no tengan letras repetidas y las imprima (dentro de la misma función por simplicidad).

Solution.



Arreglos Bidimensionales

Problem 2: (10%) Escribir un programa que genere una caminata aleatoria en una matriz de 10x10. El arreglo debe contener inicialmente puntos '.', y debe recorrerse basado en el residuo de un número aleatoria (usar **srand()** y **rand()**) cuyos resultado puede ser 0 (arriba), 1 (abajo), 2 (izq), 3 (der), que indican la dirección a moverse. A) Verificar que el movimiento no se salga del arreglo de la matriz, y B) No se puede visitar el mismo lugar más de una vez. Si alguna de estas condiciones se presenta, intentar moverse hacia otra dirección definida; si todas las posiciones están ocupadas, finalizar el programa e imprimir el resultado.

Solution.



Problem 3: (10%)

- a) Dado un arreglo bidimensional de enteros $M \times N$, encontrar el máximo valor para cada columna y cada renglón:

	9	6	8	1
8	3	6	8	1
4	2	4	3	1
9	9	5	2	1

Nota: Recorre el arreglo (columnas y renglones) en forma eficiente.

- b) Cuente el número de bytes del arreglo bidimensional con valor 0 (recuerde que cada entero está representado por 4 bytes).

Solution.



Memoria Dinámica

Problem 4: (20%) Dado una lista de nombres (string) de N personas (apellido_paterno, apellido_materno, nombre(s)), escribir una función que ordene los nombres alfabéticamente usando un arreglo de punteros:

```
char **crea_arreglo(char **arr, ...)  
char **ordena(char **arr, ...)
```

Los nombres pueden tener distinta longitud, pero la memoria que ocupan debe ser la justa (sin desperdicio); cuando un nombre sea prefijo de otro, considerar al nombre más corto como menor. El ordenamiento debe ser a través de una función que reciba el arreglo de punteros.

Solution.



Problem 5: (20%) Separe un string en tokens de acuerdo a un caracter especial dado como entrada (puede ser espacio, /, %, etc.) y que regrese un arreglo que apunte a cada uno de los tokens esperados:

```
char **tokens(char *str, char ch)
```

o NULL en caso de no encontrar algún token.

Solution.



Problem 6: Escriba una función que reciba N arreglos de enteros ordenados de menor a mayor, y mezcle los arreglos en un solo arreglo ordenado de igual forma.

Prototipo de la función: `int *merge(int **arr, int N, int *dim)` donde **arr** tiene la siguiente estructura:

N es el número total de arreglos y **dim** es un arreglo de enteros con la dimensión de cada uno de los arreglos de entrada. La función regresa un apuntador hacia el arreglo mezclado y generado dinámicamente dentro de la función **merge()**.

Nota: Generar dinámicamente todos los arreglos necesarios, e inicialice cada arreglo con valores aleatorios mediante la función **rand()**.

Solution.

