Pregunta 1

Programe la función: *int sumarVector(unsigned int x)*

El parámetro *x* almacena un vector de 8 números de 4 bits cada uno. La función debe entregar la suma de los 8 números. Por ejemplo *sumarVector(0x479)* debe entregar 20 porque 4+7+9 es 20.

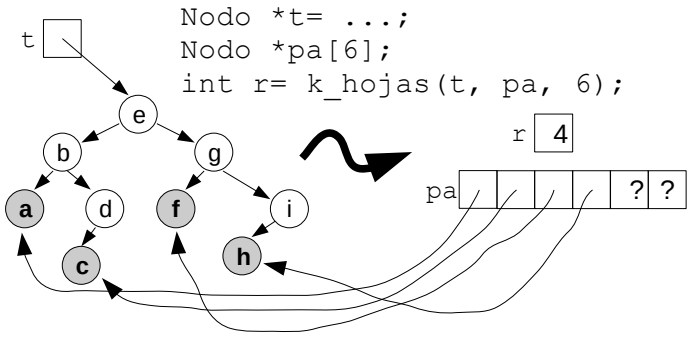
Restricciones: No use los operadores de multiplicación, división o módulo (* / %). Use eficientemente los operadores de bits, sumas y restas.

Pregunta 2

Se define una hoja en un árbol binario de búsqueda como un nodo que no tiene ningún hijo. Escriba la función *k_hojas* con el siguiente encabezado:

```
typedef struct nodo {
  char *s;
  struct nodo *izq, *der;
} Nodo;
int k_hojas(Nodo *t, Nodo **pa, int k);
```

Esta función debe recorrer en orden el árbol *t* y guardar en el arreglo *pa* las direcciones de las *k* primeras hojas de *t*. Debe retornar el número efectivo de hojas que se depositaron en *pa*, que no debe exceder *k*. En el siguiente ejemplo de uso el árbol tiene 4 hojas (a, c, f y h) y por lo tanto se guardan sus direcciones en el arreglo *pa* y se retorna 4:



Metodología obligatoria: Si *t* es el árbol nulo, termine retornando 0. Si no, si *t* es una hoja, guárdela en *pa[0]* y termine retornando 1. Si no, obtenga recursivamente hasta *k* hojas del subárbol izquierdo guardándolas en el arreglo *pa*. El valor retornado le dirá cuantas hojas se encontraron en el subárbol izquierdo. Quedaron guardadas al principio del arreglo *pa*. Si logró obtener las *k* hojas, termine retornando *k*. Si no, obtenga recursivamente las hojas restantes del subárbol derecho, para llegar hasta el máximo de *k* hojas. Ud. debe discurrir cuáles serán los parámetros *pa* y *k* que debe pasar en esta segunda invocación recursiva y qué debe retornar la función. Note que los elementos del arreglo *pa* son punteros a nodos.

Pregunta 3

Programe la función: *void* ***porciento****(char *s)*. Esta función reemplaza en *s* todas las secuencias de caracteres *o/o* por *%*. Ejemplo de uso:

```
char s[] = "el 10o/o del 10o/o es 1o/o o/";   // de largo 29
porciento(s); // s es "el 10% del 10% es 1% o/" de largo 23
```

Restricciones: No puede usar las funciones de manejo de strings como *strcmp*, *strncmp*, *strcpy*, *strlen*, etc. No use el operador de subindicación de arreglos [ ] ni su equivalente **(p+i)*, use aritmética de punteros. No puede pedir memoria adicional con *malloc* ni declarar arreglos. Sí deberá declarar punteros adicionales.

Pregunta 4

Programe la función: *int main(int argc, char *argv[ ])* para el comando *./ultima-linea*. Este comando recibe como parámetro el nombre de un archivo y despliega en la salida estándar la última línea de ese archivo. Por simplicidad considere que todas las líneas tienen a lo más 80 caracteres y terminan con el caracter adicional *'\n'*.

En el ejemplo de la derecha, el comando *cat vida.txt* muestra las 3 líneas del archivo *vida.txt*, luego se invoca *./ultima-línea*. La salida estándar de su programa debe ser solo la línea *de transmisión sexual*. Recuerde que luego de invocar *fseek(file, 0, SEEK_END)*, la función *ftell(file)* entregará el tamaño del archivo *file*.

```
$ cat vida.txt
vida:
enfermedad mortal
de transmisión sexual
$ ./ultima-linea vida.txt
de transmisión sexual
$
```

Restricciones: No puede leer más de 81 caracteres del archivo. Debe usar las funciones *fopen*, *fclose*, *ftell*, *fseek* y *fread* para leer los últimos caracteres del archivo. No se preocupe por errores como la ausencia del parámetro, inexistencia del archivo, etc.