Pregunta 1

Parte a.- Programe la función *dupMasLargo* con el siguiente encabezado:

```
typedef unsigned int ulong;
int dupMasLargo(ulong x, ulong *psec);
```

Esta función entrega el tamaño en bits de la secuencia de bits *s* más larga que aparace duplicada al final de *x* (es decir hacia los bits menos significativos). Es decir la secuencia *ss* es un sufijo de *x*. O también los bits de *x* son idénticos a *zss* para algún *z*. La secuencia *s* debe almacenarse en **psec*. En el siguiente ejemplo de uso la notación *0b...* expresa números en base 2, aunque no es parte del estándar de C.

```
ulong sec;
int tam=dupMasLargo(0b101100100100100100, &sec);
// tam==9, sec==0b100100100 (porque aparece 2 veces
// consecutivas al final de x)
```

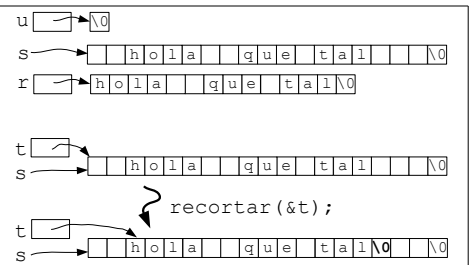
Restricción: Ud. no puede usar los operadores de multiplicación, división o módulo (* / %). Use eficientemente los operadores de bits.

Parte a.- Programe las siguientes funciones:

```
char *recorte(char *str);
void recortar(char **pstr);
```

Ambas funciones eliminan los espacios en blanco superfluos al principio y al final de un string. Para ello la función *recorte* crea un nuevo string sin alterar el string que recibe como parámetro, mientras que la función *recortar* reutiliza la memoria ocupada por el string recibido como parámetro. Ejemplos de uso:

```
char *u= recorte("  "); // u es ""
char s[]= "  hola  que tal   ";
char *r= recorte(s); // s no cambia
// string r es "hola  que tal"
free(r); // libera el string r
char *t= s;   // t==s
recortar(&t);
// string t es "hola  que tal"
// string s ¡cambia!
```



Restricciones: No use el operador de subindicación de arreglos [ ] ni su equivalente **(p+i)*, use aritmética de punteros.

Ayuda para recorte: Pida memoria para el nuevo string con *malloc*. No es posible usar *recortar* en la función *recorte* porque esto haría imposible usar *free* para liberar más tarde la memoria pedida con *malloc*.

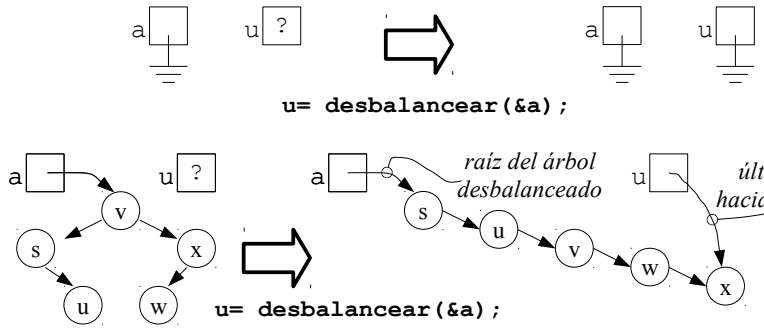
Ayuda para recortar: Avance **pstr* hasta encontrar el primer caracter que no sea un espacio en blanco. Busque donde colocar una nueva terminación para el string.

Pregunta 2

Programe la función:

```
typedef struct nodo {
  char *s;
  struct nodo *izq, *der;
} Nodo;
Nodo *desbalancear(Nodo **pa);
```

Esta función recibe en **pa* un árbol de búsqueda binaria (ABB) y entrega en el mismo **pa* un ABB *equivalente* pero desbalanceado al extremo. Un ABB está desbalanceado al extremo si (i) es el árbol vacío, o (ii) su subárbol izquierdo está vacío y su subárbol derecho está desbalanceado al extremo. Es decir todos sus nodos tienen su subárbol izquierdo vacío. Además la función retorna la dirección del último nodo yendo hacia la derecha. Las 2 figuras de más abajo muestran 2 ejemplos de uso. Las variables *a* y *u* son de tipo *Nodo **.



Restricciones: Debe ser recursivo. Su solución debe tomar tiempo O(*n*), en donde *n* es el número de nodos del árbol. No puede pedir memoria adicional con *malloc*. Reutilice los mismos nodos, reasignando los campos *izq* y *der*. Recuerde: el resultado debe seguir siendo ABB.

Ayuda: Si el subárbol izquierdo no está vacío llame recursivamente para desbalancearlo. Para rearmar el ABB le servirá el valor retornado por *desbalancear* que corresponde al último nodo del subárbol desbalanceado. Proceda de igual manera con el subárbol derecho.