Pregunta 1

Parte a.- Programe la siguiente función:

```
typedef unsigned int uint;
uint insertar_bits(uint x, int pos, uint y, int len);
```

Considere que *x* está formado por los bits x_{31} ... x_1 x_0 e *y* por y_{len-1} ... y_0. Esta función debe entregar el resultado de insertar *y* después del bit x_{pos} en *x*. Es decir que el resultado debe ser x_{31-len} ... x_{pos} $\mathbf{y_{len-1} \ldots y_0}$ x_{pos-1} ... x_0. En los siguientes ejemplos de uso la notación *0b...* expresa números en base 2. Esta notación no es parte del estándar de C, pero se emplea acá para facilitar la comprensión de los ejemplos.

```
   insertar_bits(0b101101, 3, 0b00, 2) es 0b10100101
  insertar_bits(0b110111, 0, 0b011, 3) es 0b110111011
insertar_bits(0b10111, 2, 0b01110, 5) es 0b10101110111
```

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

Parte b.- Programe la función: `void alinear_der(char *str);`

Esta función modifica el string *str* de modo que quede alineado a la derecha. Es decir mueve todos los espacios en blanco al final de *str* al comienzo (su tamaño se preserva). Ejemplo:

```
char s1[]= " hola   "; // 8 caracteres
alinear_der(s1); // s1 es "    hola" (8 caracteres)
char s2[]= "  ";
alinear_der(s2); // s2 es "  " (no se modifica)
```

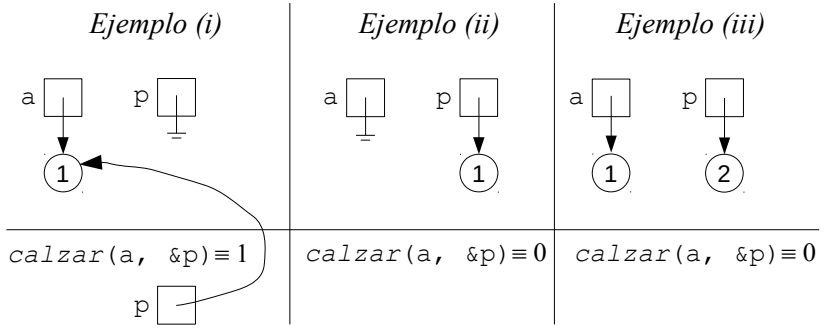
Restricciones: Ud. no puede usar el operador de subindicación [ ], ni su equivalente **(p+i)*. Para recorrer el string use aritmética de punteros. Use múltiples punteros para direccionar distintas partes del string.

Pregunta 2

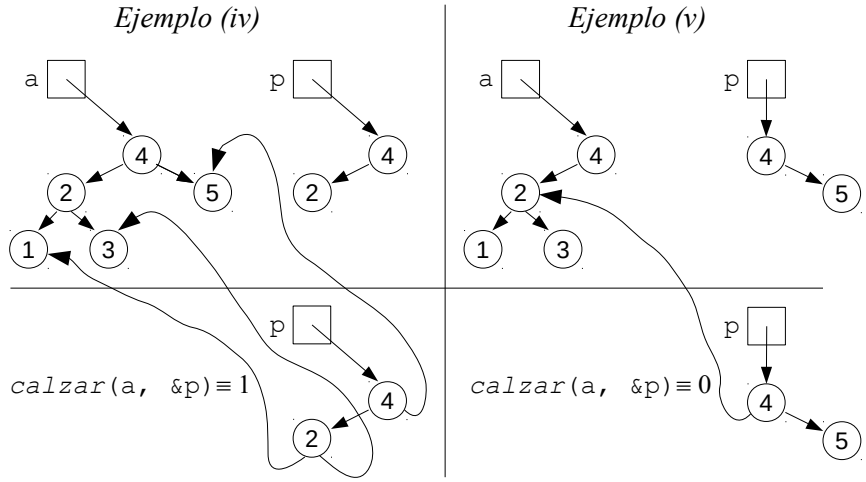
Considere la definición de la derecha para el tipo *Nodo*.

```
typedef struct nodo {
  int x;
  struct nodo *izq, *der;
} Nodo;
```

Programe la función: `int calzar(Nodo *a, Nodo **ppat);`
Esta función trata de calzar el patrón almacenado en **ppat* con el árbol binario *a*. El patrón también es un árbol binario. Calzar el patrón significa modificar los punteros nulos en el patrón de manera que apunten a nodos del árbol *a* hasta que el patrón sea igual a *a* (vea el ejemplo iv). En tal caso la función entrega 1 (verdadero). Si no se puede calzar el patrón la función retorna 0 (falso). En los siguientes ejemplos la celda superior señala la situación inicial mientras que la celda inferior indica la llamada realizada, el valor retornado y cómo se modifica el patrón.



El ejemplo (i) muestra que un patrón vacío calza con cualquier árbol, en cuyo caso el patrón se modifica de modo que sea el árbol *a* (que puede ser vacío también). El ejemplo (ii) indica que un árbol vacío no calza con un patrón no vacío y el patrón no se modifica. El ejemplo (iii) señala que un patrón no calza con un árbol cuando sus raíces difieren (el valor del campo *x* difiere). El patrón no se modifica.



El ejemplo (iv) muestra que un patrón no vacío calza con un árbol no vacío con igual raíz cuando ambos subpatrones calzan con sus respectivos subárboles. En tal caso se modifican los subpatrones. Por último el ejemplo (v) señala que el patrón no calza cuando uno de sus subpatrones no calza, pero de todos modos se modifica el otro subpatrón si éste calza con su respectivo subárbol.