***Parte a.-*** Programe eficientemente la siguiente función:

```
typedef unsigned int uint;
uint borrar_bits(uint x, uint pat, int len);
```

Esta función debe retornar el resultado de reemplazar en *x* todas las apariciones del patrón *pat* de *len* bits por ceros. En los siguientes ejemplos de uso la notación *0b...* expresa números en base 2, aunque no es parte del estándar de C.

```
borrar_bits(0b0001000100101, 0b101, 1) es 0
borrar_bits(0b1110110001, 0b110, 2) es 0b1100100001
borrar_bits(0b1110111001, 0b111, 3) es 0b1100001001
```

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

***Parte b.-*** Programe eficientemente la función:

```
char *reemplazo(char *s, char c, char *pal);
```

Esta función entrega un nuevo string resultante de reemplazar en *s* todas las ocurrencias del caracter *c* por la palabra *pal*. Ud. debe usar *malloc* para pedir memoria para el nuevo string. Ejemplo:

```
char *res= reemplazo("hola que tal", 'a', "xyz");
// res es "holxyz que txyzl"
```

*Metodología obligatoria*: Haga un primer recorrido del string *s* para calcular el largo del string resultante. Luego recorra los caracteres de *s* desde el principio de *s* hacia el final. Cuando el caracter visitado sea *c* copie *pal* en el string resultante, si no copie el caracter visitado. No olvide colocar la terminación del string. Use un puntero para recorrer *s* y otro para recorrer el string resultante.

*Restricciones*: Ud. no puede usar el operador de subindicación [ ], ni su equivalente **(p+i)*. Use ++ o *p+i*. No puede usar *strlen*, *strcpy*, etc. Por razones de eficiencia, Ud. no puede copiar *s* en el nuevo string y usar la parte *c* para resolver esta parte, ya que estaría haciendo una copia de más.

***Parte c.-*** Programe eficientemente la función:

```
void reemplazar(char *s, char c, char *pal);
```

Esta función reemplaza en *s* todas las ocurrencias del carácter *c* por la palabra *pal* dejando el resultado en el mismo *s*. La memoria destinada a *s* es suficiente para almacenar el string resultante. Ejemplo:

```
char r[17]="hola que tal";
reemplazar(r, 'a', "opa");
// r es "holopa que topal"
reemplazar(r, 'o', "");
// r es "hlpa que tpal"
```

*Metodología obligatoria*: Haga un primer recorrido del string *s* para calcular el largo del string resultante. Si el largo de *pal* es a lo más 1, recorra los caracteres de *s* desde el principio de *s* hacia el final. De lo contrario recorra *s desde el final de s hacia el principio* (en orden inverso). Para ambos casos cuando el caracter visitado sea *c* copie *pal* en la posición que le corresponde en el string resultante, si no copie el caracter visitado. Use un puntero para recorrer *s* y otro para recorrer el string resultante. No olvide colocar la terminación del string. El orden del recorrido es importante para no modificar los caracteres de *s* que aún no ha visitado.

*Restricciones*: Ud. no puede usar el operador de subindicación [ ], ni su equivalente **(p+i)*. Use ++ -- *p+i* o *p-i*. Por razones de eficiencia, Ud. no puede usar *malloc* o declarar un arreglo para pedir memoria adicional.

## *Recursos*

Baje *t1.zip* en U-cursos y descomprímalo. El directorio *T1* contiene los archivos *test-t1.c* que prueba si su tarea funciona, *t1.h* que incluye los encabezados de las funciones pedidas y *Makefile*. Ud. debe programar las 2 funciones pedidas en el archivo *t1.c*. El archivo *t1.c.plantilla* muestra los includes que Ud. debe agregar en *t1.c*. Lea las instrucciones que aparecen al comienzo de *Makefile*. Ud. debe usar el comando *make* para compilar su tarea. El programa de prueba lo felicitará si su tarea aprueba todos los tests o le indicará cuál test falla.

## *Entrega*

Ud. solo debe entregar el archivo *t1.c* por medio de U-cursos. Entregue su tarea solo si compila sin arrojar warnings en la máquina *anakena.dcc.uchile.cl* y aprueba todos los tests. Se crearán cuentas en *anakena.dcc.uchile.cl* para los alumnos que no son del DCC. Los alumnos del DCC ya tienen cuenta en ese computador. Se descontará medio punto por día de atraso. No se consideran los días sábado, domingo o festivos.