***Parte a.-*** 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*. Por razones de eficiencia, Ud. no puede copiar *s* en el nuevo string y usar la parte *b* para resolver esta parte, ya que estaría haciendo una copia demás.

***Parte b.-*** 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.

## *Instrucciones*

Baje *t2.zip* de U-cursos y descomprímalo. El directorio *T2* contiene los archivos (a) *test-reemplazar.c* que prueba si su tarea funciona y compara su eficiencia con la solución del profesor, (b) *prof.ref-x86_64 y prof.ref-aarch64* con los binarios ejecutables de la solución del profesor, (c) *reemplazar.h* que incluye los encabezados de las funcion pedidas, y (d) *Makefile* que le servirá para compilar y ejecutar su tarea. **Ejecute en un terminal el comando *make*** para recibir instrucciones adicionales. Estos son los requerimientos para aprobar su tarea.

- *make run* debe felicitarlo por aprobar este modo de ejecución. Su solución no debe ser 60% más lenta que la solución del profesor.
- *make run-g* debe felicitarlo.
- *make run-san* debe felicitarlo y no reportar ningún problema como por ejemplo desplazamientos indefinidos.

Cuando pruebe su tarea con *make run* asegúrese que su computador esté configurado en modo alto rendimiento y que no estén corriendo otros procesos intensivos en uso de CPU al mismo tiempo. De otro modo podría no lograr la eficiencia solicitada.

## *Entrega*

Ud. solo debe entregar por medio de U-cursos el archivo *reemplazar.zip* generado por el comando *make zip*. **A continuación es muy importante que descargue de U-cursos el mismo archivo que subió, luego descargue nuevamente los archivos adjuntos y vuelva a probar la tarea tal cual como la entregó**. Esto es para evitar que Ud. reciba un 1.0 en su tarea porque entregó los archivos equivocados. Créame, sucede a menudo por ahorrarse esta verificación. Se descontará medio punto por día de atraso. No se consideran los días de receso, sábados, domingos o festivos.