

Parte a.- Programe la función:

```
void eliminar(char *str, char *pat);
```

Esta función elimina del string *str* todas las apariciones del patrón *pat*.

Ejemplos:

```
char s1[] = "las palas van"; // 13 caracteres
eliminar(s1, "las"); // s1 es " pa van" (7 caracteres)
char s2[] = "111011001";
eliminar(s2, "10"); // s2 es "11101"
```

Restricciones: Ud. no puede usar el operador de subíndice [], 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. Use múltiples punteros para direccionar distintas partes del string.

Parte b.- Programe la función:

```
char *eliminados(char *str, char *pat);
```

Esta función entrega un nuevo string en donde se han eliminado del string *str* todas las apariciones del patrón *pat*. Ejemplos:

```
char *s1 = eliminados("las palas van", "las");
// s1 es " pa van" (7 caracteres)
char *s2 = eliminados("111011001", "10");
// s2 es "11101"
```

Restricciones: Ud. no puede usar el operador de subíndice [], ni su equivalente $*(p+i)$. Use $++$ o $p+i$. Para recorrer el string use aritmética de punteros. Use *malloc* para pedir memoria para el string resultante. Debe pedir exactamente la cantidad de bytes que necesita el resultado, no más. Para el primer ejemplo debe pedir 8 bytes, para el segundo 6 bytes. Si pide más memoria que la que necesita, el test de uso de memoria podría agotar la memoria de su computador haciendo que se ponga muy lento antes de que el programa falle.

Instrucciones

Baje *t2.zip* de U-cursos y descomprímalo. El directorio *T2* contiene los archivos (a) *test-elim.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) *elim.h* que incluye los encabezados de las funciones 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 80% 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 goteras de memoria.

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 *elim.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.