La función *palindrome*, cuyo encabezado se muestra más abajo, entrega 1 si la secuencia de 32 bits del entero sin signo *x* es palíndrome en hexadecimal, es decir que se lee igual de izquierda a derecha como de derecha a izquierda. Entrega 0 si *x* no es palíndrome.

```
typedef unsigned int uint32_t;  // uint32_t es equivalente a unsigned int
int palindrome(uint32_t x);
```

Ejemplos de uso:

```
int rc1= palindrome(0x3a0ff0a3);  // rc1 es 1, es palíndrome
int rc2= palindrome(0x3a0ff0a4);  // rc2 es 0, no es palíndrome
int rc3= palindrome(0x3a0fe0b3); // rc3 es 0
int rc4= palindrome(0x11); // rc4 es 0, 0x11 ≡ 0x00000011
```

Restricciones:

- Ud. no puede usar los operadores de multiplicación, división o módulo (* / %). Use los operadores de bits eficientemente.
- Debe usar un ciclo *for* con 4 iteraciones para verificar si *x* es palíndrome. Está prohibido programar código para comparar la primera cifra con la última cifra, otro código para comparar la segunda cifra con la penúltima, otro código para comparar la tercera cifra, etc.
- Se descontará medio punto por no usar el estilo de indentación de Kernighan como se explica en esta sección de los apuntes.
- El estándar de C no especifica el resultado para desplazamientos mayores o iguales al tamaño del operando. *Sanitize* rechaza el desplazamiento *x<<nbits* cuando *nbits* es mayor o superior a la cantidad de bits de *x*.

Instrucciones

Baje *t1.zip* de U-cursos y descomprímalo. El directorio *T1* contiene entre otros archivos (a) *test-palin.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) *palin.h* que incluye el encabezado de la función pedida, y (d) *Makefile* que le servirá para compilar y ejecutar su tarea. Ud. debe programar la función *palindrome* en el archivo *palin.c*.

Pruebe su tarea bajo Debian 12 nativo o virtualizado con VirtualBox, Vmware, QEmu o WSL 2. **Ejecute el comando *make* sin parámetros**. Le mostrará las opciones que tiene para compilar su tarea. 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 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 *palin.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.