15- ¿Cuál es la condición del siguiente ciclo? ¿Cuándo finaliza el ciclo? (Pila1, Pila2, y Descarte son

pilas):

```
while(!pilavacia(&Pila1))
{
        apilar(&Pila2, desapilar(&Descarte));
}
```

El ciclo se ejecuta mientras la pila Pila1 no esté vacía, lo que significa que el ciclo continuará hasta que Pila1 esté vacía, como dentro de ese while no hay nada que cambie el estado de Pila1, básicamente es un ciclo infinito, no va a finalizar nunca si no se cambiar el estado de Pila1.

16 - ¿Que realiza el siguiente código escrito en lenguaje C (Pila1, Aux y Result son pilas):

```
While(!pilavacia(&Pila1))
{
        if(tope(&Pila1 == 5))
        {
                apilar(&Aux, desapilar(&Pila1));
                apilar(&Result, desapilar(&Aux));
        }
}
```

En resumen, el código mueve todos los elementos de la pila Pila1 que tienen el valor 5 a la pila Result en el orden inverso en el que se encontraban en la pila Pila1.

Además, después de que el ciclo haya terminado, las pilas Pila1 y Aux estarán vacías y la pila Result contendrá todos los elementos de la pila Pila1 que tenían el valor 5, en orden inverso.

17- **El programa no resuelve completamente el problema planteado, ya que solo mueve los elementos de la pila Origen a la pila Distinto que están antes del primer valor 5 que se encuentra en la pila. Es posible que haya más valores 5 en la pila Origen y que no se tengan en cuenta.**

El programa tiene algunos errores:

1. **Después de mover los elementos que preceden al primer valor 5 a la pila Distinto, el programa no hace nada más. Debería seguir moviendo los elementos que quedan en la pila Origen después del primer valor 5 a la pila Distinto.**
2. **El programa no tiene en cuenta el caso en que la pila Origen esté vacía, por lo que debería agregar una comprobación de pila vacía antes de comenzar el ciclo while.**
3. **En la condición del ciclo while, debería incluir una comprobación de pila vacía para evitar que el programa intente acceder al tope de una pila vacía**.

d. • Declaración **de variables: se definen dos pilas, Origen y Distinto.**

• Inicialización de pilas: se utiliza la función inicpila() para asignar memoria y establecer los valores iniciales de las pilas.
• Lectura de datos: se utiliza la función leer() para cargar datos en la pila Origen.
• Bucle while: se utiliza un bucle while para recorrer la pila Origen mientras el tope no sea un 5.
• Condición if: se verifica si el tope de la pila es diferente a 5.
• Apilado y desapilado: si la condición se cumple, se utiliza la función apilar() para mover el elemento a la pila Distinto y la función desapilar() para eliminarlo de la pila Origen.
• Interrupción del bucle: si se encuentra un 5, se utiliza el flag como variable booleana encontrado para salir del bucle.

• Finalización del programa: si la pila Origen quedó vacía o solo tiene un 5, la pila Distinto quedará vacía

18 – **La condición es que mientras la pila Pila 1 y la pila Pila2 no estén completamente vacías, va a seguir ejecutándose las sentencias dentro del while. Al finalizar el ciclo deberían estar ambas pilas vacias, hay un tema con la memoria ahí, ya que, si una pila se vacía primero, al querer desapilar elementos de dicha pila va a haber un problema para acceder a ese espacio de memoria que ya fue liberado al desapilar antes, asi que habría que hacer una comprobación dentro del bucle por separado que se va a desapilar solo si no esta vacía esa pila en cuestión. Si se hace ese arreglo ambas pilas deberían salir vacías sin problemas, pero asi como está escrito el código no podría garantizar que eso pase, seguramente va a tirar algún error de memoria al ejecutar.**