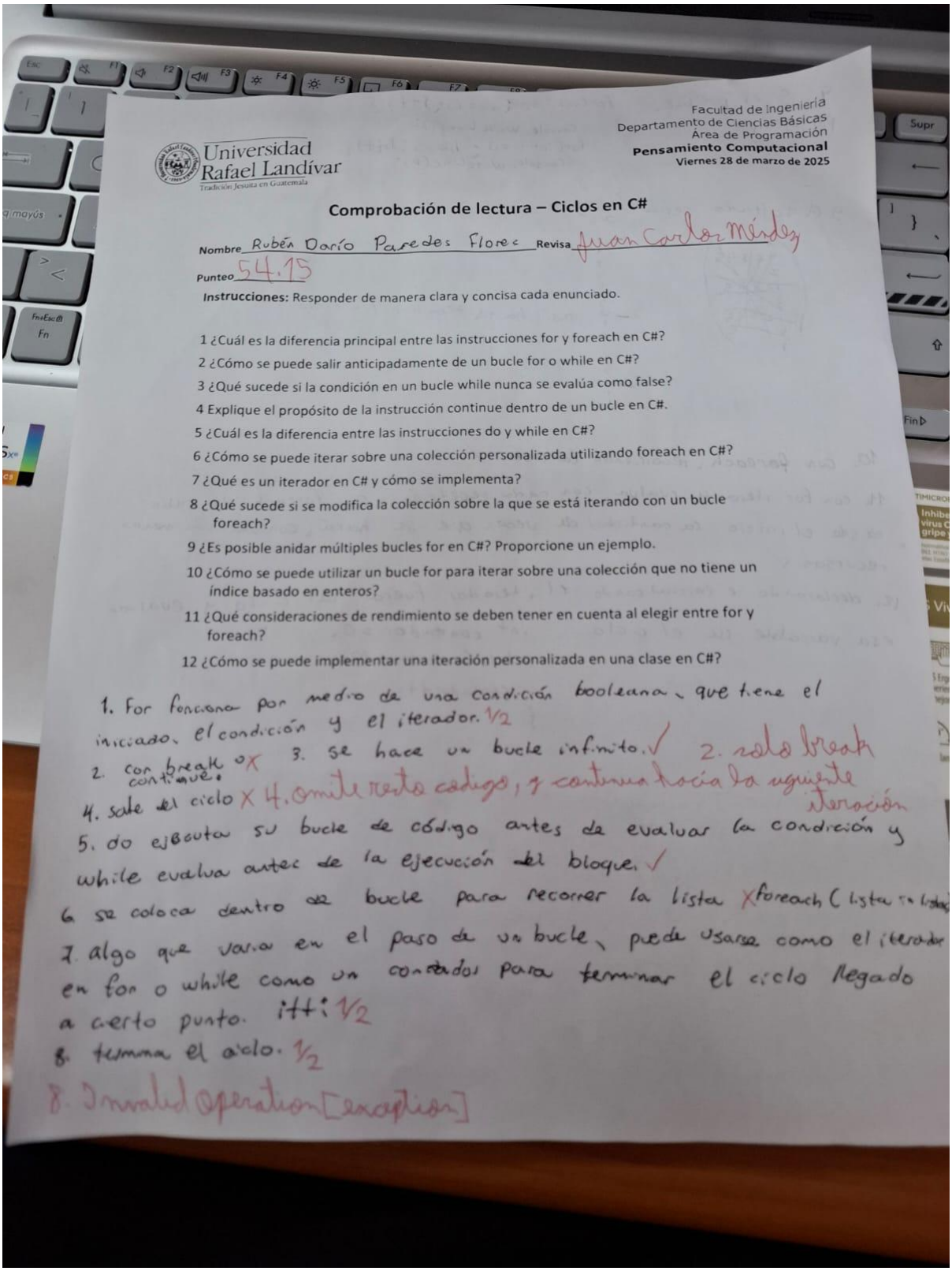
Rubén Darío Paredes Flores – 1152225



Universidad Rafael Landívar
Tradición Jesuita en Guatemala

Facultad de Ingeniería
Departamento de Ciencias Básicas
Área de Programación
**Pensamiento Computacional**
Viernes 28 de marzo de 2025

## Comprobación de lectura – Ciclos en C#

**Nombre** Rubén Darío Paredes Flores **Revisa** Juan Carlos Méndez

**Punteo** 54.15

**Instrucciones:** Responder de manera clara y concisa cada enunciado.

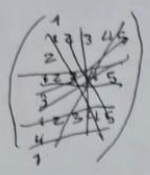
1 ¿Cuál es la diferencia principal entre las instrucciones for y foreach en C#?
2 ¿Cómo se puede salir anticipadamente de un bucle for o while en C#?
3 ¿Qué sucede si la condición en un bucle while nunca se evalúa como false?
4 Explique el propósito de la instrucción continue dentro de un bucle en C#.
5 ¿Cuál es la diferencia entre las instrucciones do y while en C#?
6 ¿Cómo se puede iterar sobre una colección personalizada utilizando foreach en C#?
7 ¿Qué es un iterador en C# y cómo se implementa?
8 ¿Qué sucede si se modifica la colección sobre la que se está iterando con un bucle foreach?
9 ¿Es posible anidar múltiples bucles for en C#? Proporcione un ejemplo.
10 ¿Cómo se puede utilizar un bucle for para iterar sobre una colección que no tiene un índice basado en enteros?
11 ¿Qué consideraciones de rendimiento se deben tener en cuenta al elegir entre for y foreach?
12 ¿Cómo se puede implementar una iteración personalizada en una clase en C#?

1. For funciona por medio de una condición booleana, que tiene el iniciado, el condición y el iterador. 1/2
2. con break o continue. X 3. se hace un bucle infinito. ✓ 2. solo break
4. sale del ciclo X 4. omite resto código, y continua hacia la siguiente iteración
5. do ejecuta su bucle de código antes de evaluar la condición y while evalua antes de la ejecución del bloque. ✓
6 se coloca dentro de bucle para recorrer la lista X foreach ( lista : lista)
7. algo que varia en el paso de un bucle, puede usarse como el iterador en for o while como un contador para terminar el ciclo llegado a cierto punto. itt: 1/2
8. termina el ciclo. 1/2
8. Invalid operation [exception]

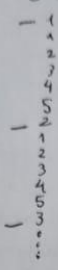
9. Sí es posible. Por ejemplo.

```
for(int i=0; i<=10; i++) {
    Console.WriteLine(i);
    for(int b=0; b<=5; b++) {
        Console.WriteLine(b);
    }
}
```

y el resultado sería así



así hasta terminar los ciclos



10. con foreach, recorriendo la lista. ✓

11. con for itera y evalua con cada repetición, con foreach ya sabe desde el inicio la cantidad de veces que se hará, consumiendo menos recursos. ✓

12. declarando e inicializando el iterador fuera del ciclo y evaluar esa variable en el ciclo.

```
int contador = 0;
while (contador < 10) {
    contador++;
}
```

✗

6. No genérica: IEnumerable
Especifica tipo elemento: IEnumerable<T>
método GetEnumerator

12. Repuesta en Teams