Métodos Recursivos en C#

Nombre del estudiante: _____

Fecha: _____

Instrucciones:

A continuación, se presentan fragmentos de código de métodos recursivos en C#. Debes identificar (copiar la o las líneas dentro de la condicional correspondiente) o completar la parte recursiva y el caso base de cada método. Escribe tu respuesta en los espacios proporcionados.

Ejercicio 1: Factorial de un número

```
int Factorial(int n)
{
  if (n=1){
     return 1;
  }else{
     return n*Factorial(n-1);
  }
}
```

Caso base identificado: _____

Parte recursiva identificada: _____

Ejercicio 2: Suma de los primeros N números

```
int Suma(int n)
{
  // Caso base
  if (_____) {// completa aquí
     return 0;
  }else{
  // Parte recursiva
  return _____; // completa aquí
  }
}
```

Ejercicio 3: N-ésimo número de Fibonacci

```
int Fibonacci(int n)
{

   if(                              )
   {


   }
   else
   {
       return 1;
   }
}
```


LIVEWORKSHEETS

Ejercicio 4:

Relaciona correctamente los conceptos de la columna A con sus respectivas definiciones o ejemplos de la columna B. Escribe la letra correspondiente en el espacio provisto.

Columna A (Conceptos)

1. Recursión
2. Caso base
3. Llamada recursiva
4. Pila de llamadas
5. Desbordamiento de pila (Stack Overflow)

Columna B (Definiciones)

A. Ocurre cuando una función se llama a sí misma sin llegar a un caso base.
B. Técnica de programación donde una función se llama a sí misma para resolver un problema.
C. Estructura de datos utilizada para controlar las llamadas de funciones en tiempo de ejecución.
D. Parte de la función que detiene la recursión.
E. Instrucción dentro de una función que ejecuta otra instancia de sí misma.

Respuestas:

1. _____ 2. _____ 3. _____ 4. _____ 5. _____

Ejercicio 5: Selecciona la respuesta correcta

1. **¿Qué es la recursividad?**

   A. Una técnica para ordenar datos

   B. Una función que se llama a sí misma

   C. Una forma de compilar funciones

   D. Una estructura de datos

2. **¿Cuál de las siguientes opciones describe correctamente el *caso base* en una función recursiva?**

   E. La parte donde la función llama a sí misma

   F. La condición que inicia la recursión

   G. La condición que termina la recursión

   H. La definición de parámetros

LIVEWORKSHEETS

3. **¿Qué pasaría si una función recursiva no tiene un caso base bien definido?**

   I. El programa se ejecutará más rápido

   J. El compilador optimizará la función

   K. Se producirá un desbordamiento de pila

   L. No hay consecuencias

4. **¿Cuál es la salida del siguiente código si se llama Contar(3)?**

```
void Contar(int n)
{
    if (n == 0){ return;}
    else{
        Console.WriteLine(n);
        Contar(n - 1);
    }
}
```

   M. 0 1 2 3

   N. 3 2 1

   O. 3 2 1 0

   P. 1 2 3

Pregunta 6: Preguntas de Falso y Verdadero

Escribe V y el enunciado es verdadero y F si no lo es

1. La recursividad siempre requiere una condición de salida o caso base.

2. Una función que llama a otra, pero no a sí misma, es una función recursiva.

3. Una función puede tener múltiples llamadas recursivas en su cuerpo.

4. Toda función recursiva puede reescribirse como una función iterativa.

5. El siguiente código es un ejemplo correcto de función recursiva:

```
int Suma(int n)
{
    if (n <= 0){ return 0;}
    return n + Suma(n - 1);
}
```


LIVEWORKSHEETS

LIVEWORKSHEETS