Trabajo de Investigación RECURSIVIDAD

Alumno: Leandro Panozzo

Año: 2023

Materia: Laboratorio de Programacion 1

Profesora: Carolina Vargas



(Gottfried, 2005)

# Bibliografía

Gottfried, B. (2005). *PROGRAMACION EN C Schawn.* Aravaca (Madrid): McGraw-Hill/lnteramericana.

**Responder los siguientes interrogantes**

1. ¿Qué es la Recursividad?

2. ¿Qué ventajas tiene la Recursividad?

3. ¿Cómo compara la Recursividad con la Iteración?

4. Si se programa recursivamente un proceso iterativo, ¿será necesariamente más eficiente el programa que el correspondiente a una versión no recursiva?

5. Cuando se ejecuta un programa que contiene llamadas recursivas, ¿cómo se interpretan las variables locales dentro de una función recursiva?

6. Realizar un programa que ejemplifique la Recursividad. Explicar cómo funciona la recursividad

1. Se llama recursividad a un proceso mediante el que una función se llama a sí misma de forma repetida, hasta que se satisface alguna condición determinada. El proceso se utiliza para cálculos repetitivos en los que cada acción se determina en función de un resultado anterior. Se pueden escribir en esta forma muchos problemas repetitivos.
2. destacan varias ventajas de la recursividad en la programación, algunas de ellas son:
3. Simplificación de problemas complejos: La recursividad puede utilizarse para simplificar problemas complejos en tareas más pequeñas y manejables. Un problema grande puede dividirse en problemas más pequeños, que se resuelven recursivamente y luego se combinan en una solución.
4. Ahorro de espacio: A veces, la recursividad puede ser más eficiente en términos de espacio que una solución iterativa. Debido a que las llamadas recursivas pueden compartir la misma pila de memoria.
5. Legibilidad del código: En algunos casos, la recursividad puede hacer que el código sea más fácil de leer y entender.
6. Facilidad de implementación: En algunos casos, la recursividad puede ser más fácil de implementar que una solución iterativa.
7. La recursividad utiliza la estructura de control de llamada de función para repetir una tarea, mientras que la iteración utiliza la estructura de control de bucle (por ejemplo, while, for) para repetir una tarea. La recursividad requiere más memoria que la iteración, ya que cada llamada recursiva agrega una nueva pila de memoria en la pila de llamadas. En contraste, la iteración utiliza la misma cantidad de memoria en cada iteración y La recursividad puede ser utilizada para resolver problemas que son difíciles o imposibles de resolver con iteración, como la manipulación de estructuras de datos recursivas, mientras que la iteración es más adecuada para problemas que se pueden resolver de manera iterativa.
8. No necesariamente. En algunos casos, un proceso iterativo puede ser más eficiente que la versión recursiva correspondiente, y en otros casos puede ser al revés.

La recursión puede tener ciertas desventajas en términos de rendimiento y uso de memoria. Cada vez que se llama a una función recursiva, se agrega una nueva pila de memoria en la pila de llamadas, lo que puede consumir mucha memoria y causar problemas de desbordamiento de pila si la recursión es muy profunda.

Un proceso iterativo no tiene el mismo costo de llamada de función que la recursión, lo que lo hace más eficiente en algunos casos. Además, pueden optimizarse para aprovechar mejores recursos del sistema, lo que puede mejorar aún más su rendimiento.

Resumiendo, la elección entre una solución recursiva y una solución iterativa dependerá de las características específicas del problema y de las necesidades de rendimiento y uso de memoria del programa dependiendo de esto una solución u otra puede ser más eficiente y práctica.

1. las variables locales dentro de una función recursiva se comportan de la misma manera que las variables locales en una función no recursiva, con la diferencia de que se crean nuevas instancias de esas variables cada vez que se llama a la función recursiva. Cada instancia de la función tiene su propia copia de las variables locales, que solo existen mientras se está ejecutando esa instancia. Cada vez que se realiza una llamada recursiva, se agrega una nueva pila a la pila de llamadas. Cada instancia de la función se ejecuta en su propia pila, y cuando se completa la ejecución de una instancia, se elimina su pila de la pila de llamadas.
2. el programa utiliza una función recursiva llamada factorial para calcular la factorial de un número.

Cuando se llama a la función factorial(n), la función comprueba si n es 0 o 1. Si es así, devuelve 1 como resultado. Si n es mayor que 1, la función se llama a sí misma con el argumento n-1, y el resultado se multiplica por n para obtener la factorial de n.

La recursividad funciona llamando a una función dentro de sí misma, creando una pila de llamadas para cada instancia de la función. Cada vez que se llama a la función recursiva, se agrega una nueva pila de llamadas a la memoria, y cuando se completa la ejecución de la función, se elimina su pila de la memoria.

# Bibliografía

Gottfried, B. (2005). *PROGRAMACION EN C Schawn.* Aravaca (Madrid): McGraw-Hill/lnteramericana.

