Instituto Tecnológico Superior de Jerez – ITSJ

Estudiante: Miguel Angel Bazán garduño.

mabg211299@hotmail.com

3er Semestre Carrera: Ingeniería en sistemas computacionales (ISC).

Materia: Estructura de Datos.

Actividad: Mapa Conceptual.

Docente: I.S.C. Salvador Acevedo Sandoval

Jerez de García Salinas Zacatecas

1. ¿Qué es la RECURSIVIDAD?

Es una técnica de programación que se utiliza para realizar una llamada a una función desde ella misma, de allí su nombre

2. ¿Para qué se utiliza la recursividad?

Se utiliza para operaciones repetidas en las que cada acción se determina mediante un resultado anterior

3. ¿Qué ventajas y desventajas existen al utilizar recursividad?

Ventajas:

- No es necesario definir la secuencia de pasos exactos para resolver el problema
- Soluciones elegantes
- Soluciones a problemas complejos.

Desventajas:

- Podría ser menos eficiente
- Sobrecarga asociada con las llamadas a subalgoritmos
- El valor de la recursividad reside en el hecho de que se puede usar para resolver problemas sin fácil solución iterativa.
- La ineficiencia inherente de algunos algoritmos recursivos.

4. ¿Qué es un procedimiento/función recursivo?

es aquel que se llama así mismo, solo que no regresa valor

5. ¿Cuáles son las partes que conforman un función/procedimiento recursivo?

Debe existir criterio base para que este se llame a sí mismo.

Cada vez que el procedimiento se llame a si mismo debe estar más cerca del criterio base.

6. ¿Por qué son importantes estas partes que lo conforman?

Para que no ocurra ningún Tipo de error al momento de llamarlo.

7. ¿Cuáles son los pasos para implementar un procedimiento recurso?

1. Primero obtener una función exacta del problema a resolver.
2. A continuación, determinar el tamaño del problema completo que hay que resolver, este tamaño determina los valores de los parámetros en la llamada inicial al procedimiento o función.
3. Resolver el caso base en el que el problema puede expresarse no recursivamente, esto asegura una respuesta afirmativa a la pregunta base.

8. Tipos de recursividad

Recursión directa. Cuando el código F tiene una sentencia que involucra a F.

Recursión indirecta o cruzada.- Cuando la función F involucra una función G que invoca a la ves una función H, y así sucesivamente, hasta que se involucra la función F. Por ejemplo el algoritmo de Par o impar.

Recursión simple.- Aquella en cuya función solo aparece una llamada recursiva. Se puede transformar con facilidad en algoritmos iterativos.

Recursión múltiple.- Se da cuando hay más de una llamada a sí misma dentro del cuerpo de la función, resultando más difícil de transformar a iterativa. Por ejemplo el algoritmo de Fibonacci.

Recursión anidada.- En algunos de los argumentos de la llamada hay una nueva llamada a sí misma.

Recursión infinita

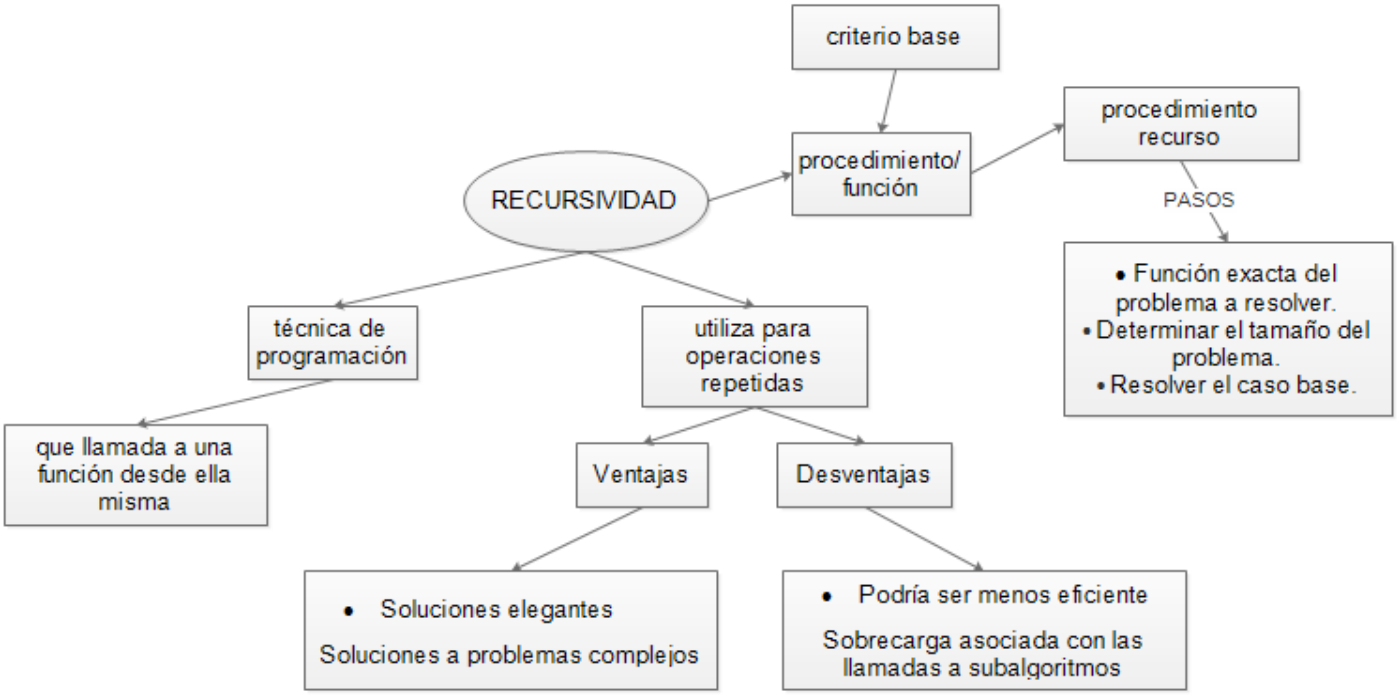
La iteración y la recursión pueden producirse infinitamente. Un bucle infinito ocurre si la prueba o test de continuación del bucle nunca se vuelve falsa.

En realidad, la recursión infinita significa que cada llamada recursiva produce otra llamada recursiva y esta a su vez otra llamada recursiva, y así para siempre. En la práctica, dicha función se ejecutará hasta que la computadora agote la memoria disponible y se produzca una terminación anormal del programa.

9. ¿Qué consideraciones tomaría en cuenta para resolver un problema mediante un algoritmo recursivo?

Que la solución depende de las soluciones de pequeñas instancias del mismo problema.

10. ¿Definitivamente cuándo no aplicaría un algoritmo recursivo?


criterio base
procedimiento/ función
procedimiento recurso
PASOS
• Función exacta del problema a resolver.
• Determinar el tamaño del problema.
• Resolver el caso base.
RECURSIVIDAD
técnica de programación
que llamada a una función desde ella misma
utiliza para operaciones repetidas
Ventajas
Desventajas
• Soluciones elegantes
Soluciones a problemas complejos
• Podría ser menos eficiente
Sobrecarga asociada con las llamadas a subalgoritmos