Resumen de programacion

August 17, 2020

1 Recursion

definición inductiva: obtener conceptos nuevos empleando el mismo concepto que intenta describir.

Una **definición recursiva** dice cómo obtener conceptos nuevos empleando el mismo concepto que intenta definir

Un razonamiento recursivo tiene **dos partes:**

- La base, que no es recursiva y es el punto tanto de partida como de terminación de la definición. (son los elementos del conjunto se especifican explícitamente y aseguren el final o corte)
- regla recursiva de construcción. Los elementos del conjunto que se definen en términos de los elementos ya definidos

Recursividad de cola: Cuando una llamada recursiva es la última posición ejecutada del procedimiento

Recursion directa: Cuando un procedimiento incluye una llamada a si mismo

Recursion indirecta: Cuando un procedimiento llama a otro procedimiento y este causa que el procedimiento original sea invocado

Es menos peligrosa una recursión infinita que un ciclo infinito, ya que una recursividad infinita pronto **se queda sin espacio** (para cada llamada se crean copias independientes de las variables declaradas en el procedimiento) y termina el programa, mientras que la iteración infinita puede continuar mientras no se termine en forma manual.
el cálculo recursivo es mucho más caro que el iterativo.

Pasos de la recursion:

1. El procedimiento se llama a si mismo
2. El problema se resuelve, resolviendo el mismo problema pero de tamaño menor
3. La manera en la cual el tamaño del problema disminuye asegura que el caso base eventualmente se alcanzará

2 Principios de programacion

Nombres de variables y funciones

- Se deben elegir nombres significativos que sugieran claramente la finalidad de la funcion o variable en cuestion
- Las variables que se usan poco deben tener nombres mas sencillos (una letra basta).
- No cometer errores de ortografia.

Documentacion y formato

- Es conveniente documentar el programa **Antes** de empezar a codearlo.
- Se introduce cada seccion importante del programa con un comentario breve que explique su finalidad
- No se hacen comentarios que expliquen lo obvio ni se comenta cada linea.

Refinamiento y Modularidad

- Se deben elegir nombres significativos que sugieran claramente la finalidad de la funcion o variable en cuestion
- Las variables que se usan poco deben tener nombres mas sencillos (una letra basta).
- No cometer errores de ortografia.

Codificacion, prueba y refinamiento ulterior

- **Refinamiento descendente** Primero pensar en el programa entero y desde alli ramificarlo en subprogramas mas pequeños.
- Siempre revisar nuestro codigo para ver que se le puede mejorar. Para ello, es muy util probar el codigo, y existen 3 maneras conocidas para hacerlo:

1. **caja negra** Se prueba el codigo con valores tipicos, realistas, genericos, etc.
2. **caja de cristal** Se prueba el codigo con todos los valores de entrada posibles.
3. **Caja de pandora** Dejamos que el cliente haga las pruebas y nos comunique los resultados.

Algunas otras recomendaciones:

- Procurar que los algoritmos sean lo mas simple y pequeños posibles.
- Analice las exigencias del tiempo y espacio al seleccionar un algoritmo.
- Comenzar de nuevo suele ser mas fácil que parchar un programa

3 Estructura de Datos

La pc solo entiende numeros binaros. Para mostrarnos todo lo que podemos ver en pantalla, todo se convierte a binario de distintas formas:

Enteros binarios 00100110 $\rightarrow$ 38// El primer dígito representa positivo o negativo, en caso de ser negativo se invierten todos los valores de los otros BITS 11011001 = -38

Notación de Punto Flotante El número real se representa por un número llamado MANTISA multiplicado por una base de elevada a una potencia entera, llamada EXPONENTE.
Ej. El número 387,53 se representa como $38753x10^{-2}$

Cadena de Caracteres Se pueden representar caracteres también.
Si 8 bits representan un carácter, se pueden representar hasta 256 caracteres diferentes. bit $\rightarrow$ 0 / 1