Programación modular

1. Introducción

- Programación estructurada: todo el código dentro de un mismo bloque
- Definición de módulos programados de forma independiente y compilados por separado
- Se usa programación modular y estructurada al mismo tiempo
- Prog. Modular descompone problema en subproblemas (refinamiento) llamados módulos
- Módulos se comunican entre ellos mediante paso de parámetros

2. Programación modular

- Objetivos módulo:
    - Cada uno realiza una tarea, un identificador diferente. Se pasan parámetros
    - Módulos tienen un único punto de entrada y de salida
    - Unos módulos pueden llamar a otros

Ventajas

- Menor coste de desarrollo. Trabajo simultáneo en paralelo
- Reutilización de código. Facilidad de probar y mantener
- No es necesario conocer el funcionamiento interior del módulo
- Módulos se pueden agrupar en librerías o bibliotecas para poder ser usados en otros

3. Diseño descendente TOP-DOWN

- Descomposición de problemas en sucesivos niveles
    - Cada módulo tiene una función lo más específica posible
    - Reducción del tamaño de los subproblemas. Minimizar duplicidad de código. Módulos reutilizables
    - Se agrupan módulos comunes según funcionalidad o temática

4. Clasificación de módulos

Según el módulo que lo invoca

- Interno: están en el mismo fichero que el módulo que lo invoca
- Externo: está en un fichero diferente

Según retorno de valores

- Funciones: deben devolver un valor
- Definición, Declaración, Llamada
    - Procedimientos:
        - No devuelven ningún valor. Parámetros de entrada, salida, o E/S.
        - Permiten devolver más de un valor
        - Definición, declaración, llamada

Según cuando ha sido desarrollado

```
-  De programa: se ha desarrollado en el programa actual
-  De librería: ha sido desarrollado rpeviamente y está contenido en ficheros de librerías
```

Según el número de módulos que realizan la llamada

- Subprograma: es invocado por un solo módulo
- Rutina o subrutina: es invocado por diversos módulos

5. Estructura modular

- Se puede representar mediante diagrama basado en jerarquía, descomposición e independencia de módulos
    - Estructura secuencial
    - Estructura selectiva
    - Estructura repetitiva

6. Diseño de funciones

Parámetros

```
-  Entrada, salida, entrada y salida
-  Paso por copia, por valor, por referencia
```

Ámbito de identificadores

```
-  Identificador puede ser variable, función, procedimiento, constante, etv.
-  Identificadores globales y locales
-  Recomendaciones
```

- Independencia funcional

7. Recursividad

- En el cuerpo de un subprorgama hay una llamada al propio subprograma.
- Condiciones: necesita una condición de parada. Cada iteración debe utilizar cada vez una versión del problema más reducida.
- En algoritmos o en datos
- Simple y múltiple
- Directa e indirecta
- Ventajas: solución simple y elegante a problemas complejos. No es necesario definir secuencia de pasos

- Inconvenientes: sobrecarga puede producir desbordamiento de la pila. Ineficiencia de algunos algoritmos

8. Librerías

- Conjunto compilado,documentado y probado de procedimientos y funciones
- Se puede invocar desde otro programa
- Necesario hacer referencia a la librería

Tipos de librerías

- Código fuente (necesario compilar)
- Compiladas (se enlazan)
- Enlace dinámico: compartidas por varias aplicaciones, reduce tamaño ejecutable, aprovechameitno de memoria

Características

Ejemplos (Java, C)

Ejemplos de librerías:

- C: stdio.h, stdlib.h
- Java: java.io, java.net