Funciones y Procedimientos

Programación digital I
Escuela de Sistemas
Facultad de Ingeniería
Gilberto Diaz

Programación Modular

- Un software monolítico no puede ser entendido fácilmente por un solo lector. El número de caminos de control, ámbito de referencia, número de variables y la complejidad global harían su comprensión casi imposible.

Programación Modular

- Es más fácil resolver un problema complejo cuando se rompe en piezas manejables.

¡Divide y Vencerás!

Modularidad: Acoplamiento

- Un módulo debe ofrecer un grupo de servicios diseñados para que el resto del programa pueda interactuar con él

- Por ejemplo, en el procesador de texto se debe contar con rutinas para:
  - Cambiar la letra utilizada: cambiarEstilo()
  - Cambiar el color: cambiarColorLetra()
  - etc.

Programación Modular

- Es un método de resolución de problemas que consiste en resolver de forma independiente los sub-problemas que se obtienen de una descomposición del problema general

Programación Modular

- La base fundamental de este paradigma de programación es el **módulo**
- Un módulo en un conjunto de **rutinas** que prestan un servicio específico.
- Una **rutina, subrutina o subprograma,** como idea general, se presenta como un algoritmo separado del algoritmo principal, el cual permite resolver una tarea específica.

Programación Modular

- Decimos que algo es modular si es construido de manera tal que se facilite su ensamblaje, acomodamiento flexible y reparación de sus componentes.

Programación Modular

- La modularidad da una mejor compresión del problema y reduce el tamaño del código
- Generalmente se hace la asociación de un módulo como una caja negra. Se sabe que entra y que sale pero no como se procesa

Funciones y Procedimientos

- En la programación modular se debe asegurar los siguientes preceptos
  - máxima cohesión
  - mínimo acoplamiento entre módulos

Modularidad: Cohesión

- Un módulo debe ofrecer un grupo de servicios que sin lugar a dudas deben ir juntos, por ejemplo:
  - math.h
  - stdio.h
- Por ejemplo en un procesador de texto, todos los módulos tienen como objetivo cooperar para producir documentos con estilo y formato definido (módulo de colores, módulo de tamaño, etc)

Ventajas de los Módulos

- Facilitan la escritura y depuración de un programa
- Localización rápida de errores
- La modificación de un módulo no afecta a los demás
- Un grupo de instrucciones que se repite en varias partes de un programa puede incluirse en un módulo y llamarlo en el programa.

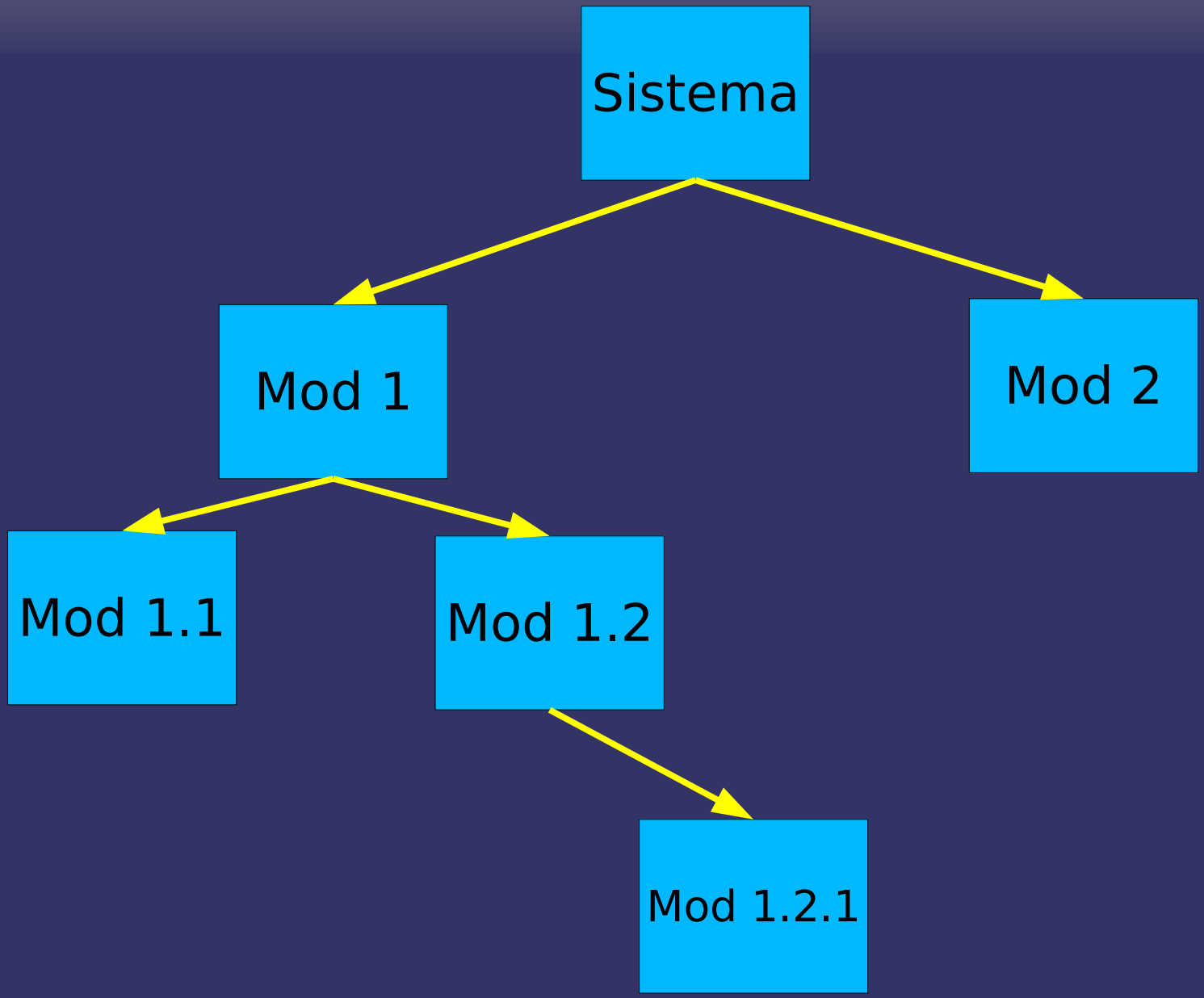
Diseño Descendente

- En us proceso de refinamiento por pasos, etapas o capas. Se comienza desde lo más general hasta lo más específico.
- En la solución de problemas grandes es conveniente dividirlo en problemas más pequeños, los cuales a su vez pueden dividirse en sub-problemas más pequeños.

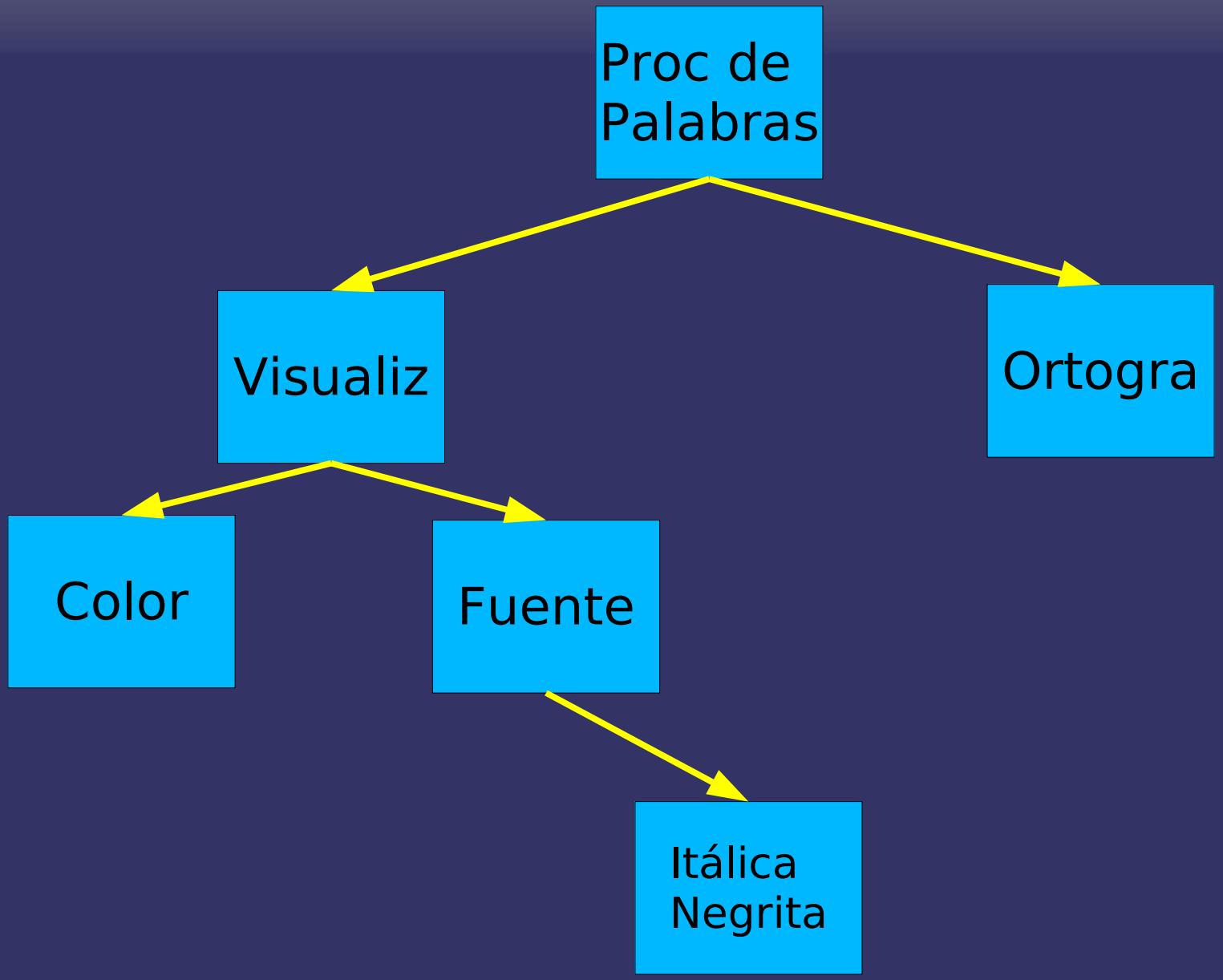
Diseño Descendente

- Se inicia desde lo más general, luego dividir y vencer
- Asegurarse de las capacidades y especificaciones del lenguaje. Esto permite cambiar el lenguaje en medio del diseño sin ningún trauma.
- Postergar lo más posible el trabajo en los detalles.
- Verificar cuidadosamente cada nivel.

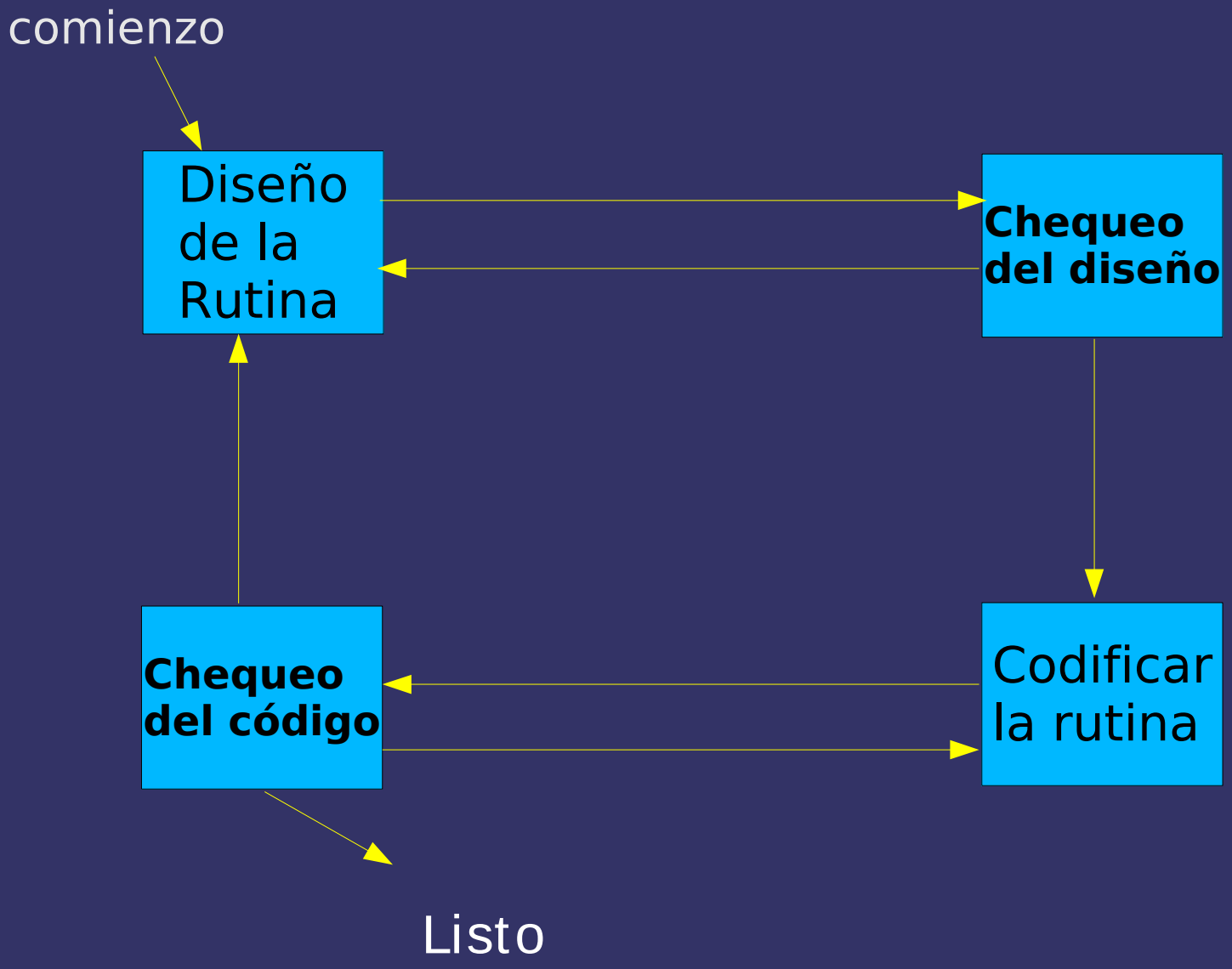
Diseño Descendente



Diseño Descendente



Diseño de Rutinas



Diseño de Rutinas

- Chequeo de prerrequisitos. Verificar si en trabajo de la rutina esta bien definida
- Definir el problema de la rutina en términos de los datos de entrada, salida y gestión de errores.
- Nombre de la rutina. Debe ser conciso y representativo de lo que hace la rutina.

Diseño de Rutinas

- Establecer mecanismos de prueba para la rutina. Revisar cualquier caso que genere un error.
- Pruebe con diferentes formas para codificar la rutina. La construcción de programas es iterativa, es decir, se prueba, se falla y se realiza un nuevo esfuerzo hasta conseguir la versión definitiva.

Codificación de Rutinas

- El diseño de una rutina equivale al plano de una casa.
- La construcción de la casa equivale a la codificación de la rutina
- Escriba la declaración de la rutina. Esto se conoce como interfaz
- Escriba el algoritmo con frases de muy alto nivel

Codificación de Rutinas

- Complete cada frase de alto nivel con una o más líneas de código.
- Cheque el código informalmente.
- Comente y haga una prueba de los trozos de código nuevo

Chequeo Formal de Código

- Haga una inspección mental de código. Repase los algoritmos que se han inventado para la solución del problema.
- Compile la rutina. Esto mostrará todos los errores de sintaxis.
- Elimine todas las causas que generen advertencias

Reutilización de un Módulo

- Generalmente los algoritmos de cada modulo sólo se escriben una sola vez
- La reutilización de un módulo por otros programas implica ahorro de tiempo. Si el módulo ha sido probado y verificado previamente, se reduce la posibilidad de errores.
- Fácil compresión del programa completo.

Función

- En C los módulos se llaman funciones. (unidad básica de los programas)
- Una función se define una sola vez pero puede ser utilizada tantas veces como sea necesario a través de una llamada.

Función

```
Int  funcion1 ( arg1, arg2 ...){
   sentencias
}

int funcion2 (arg1, arg2 ...){
   sentencias
   funcion1(a,b...)
}

int main(){
   funcion1(arg1, arg2...);
   funcion2(arg1, arg2...);
```

Tipo de Funciones en C

- Funciones de biblioteca (módulos): C tiene su propio conjunto de bibliotecas de funciones básicas que permiten realizar las operaciones de entrada salida, operaciones lógicas, aritméticas.

- Funciones definidas (diseñadas y codificadas) por el programador para realizar sus propias tareas.

Biblioteca Estándar de C

- Contiene una amplia colección de funciones para llevar a cabo:
  - cálculos matemáticos comunes,
  - manipulaciones con cadenas de caracteres,
  - operaciones de entrada salida, etc

- Esta biblioteca de funciones comunes construida una vez, puede ser reutilizada por diferentes programas.

Biblioteca Estándar de C

- Entrada salida
  - Operaciones sobre archivos
  - Entrada – Salida

- Cadenas de caracteres: string.h
- Funciones matemáticas: math.h
- Funciones varias: stdlib.h (atoi, atof, calloc, malloc, etc)
- Fecha y hora: time.h

Ejercicios

- Escriba una función que devuelva el mayor de dos números que se le pasan como entrada.
- Escriba una función que devuelva el factorial de un número.