Ejercicio enlaces a una posición de una página – Anclas 2.

Realizar las siguientes tareas.

- Crea la carpeta anclas.
- Crea una página web con el nombre de anclas.html y guárdala en la carpeta anclas.
- Crea una página web con el nombre ancla_externa.html y guárdala en la carpeta anclas.
- En la página anclas.html pega el texto que viene a continuación y para que mantenga el mismo formato usa por ejemplo la etiqueta <pre>.
- En la página ancla_externa.html crea un enlace para ir al final de la página anclas.html.
- Al principio de la página anclas.html crea las siguientes anclas:
    - Una para desplazarse al apartado "Programación orientada a objetos (POO)"
    - Otra para desplazarse al apartado "Modularización".
    - Una tercera para ir al final de la página.
    - Al final de la página anclas.html, crea un ancla para volver al principio de la página y otra para subir al apartado "Programación Estructurada".

Texto a copiar:

INTRODUCCIÓN

En el presente trabajo de investigación se darán a conocer alunas metodologías de programación utilizadas en la creación de softwares. Una metodología es un conjunto de principios generales que un programador ha de seguir para desarrollar un programa. Cada lenguaje de programación permite al programador seguir una metodología determinada, por tanto, los lenguajes de programación se pueden también clasificar según la metodología que sigan. Las metodologías de programación definen cómo se organizan los diferentes elementos de un programa, como esos elementos interactúan entre ellos, etc. Actualmente, las metodologías de programación más comúnmente usadas son los siguientes:

- PROGRAMACIÓN ESTRUCTURADA
- PROGRAMACIÓN ORIENTADA A OBJETOS
- MODULARIZACIÓN

A continuación, se detallan cada uno de estos puntos.

PROGRAMACIÓN ESTRUCTURADA

A finales de los años 1970 y principios de los 1980, se creó una nueva filosofía de diseño de programas llamada *programación estructurada*. Esta idea eliminó ese tipo

de código largo, monolítico y lineal que saltaba de un lugar a otro con sentencias tipo GOTO. En su lugar, los programadores seccionaban el código en secciones claramente definidas mediante párrafos cortos los cuales eran llamados por un módulo principal. En lugar de la utilización del lenguaje Ensamblador los lenguajes de tercera generación, tales como COBOL, PASCAL y BASIC se hicieron muy populares. Además, un *hardware* de ordenador más sofisticado y más rápido hizo posible las bases de datos relacionales, permitiendo el almacenamiento de la información en diskettes y en discos duros. A finales de los 1970 y comienzos de los 1980 era muy caro hacer esto, pero los beneficios eran tremendos. Por no más tiempo del que necesitabas para cerrar un sobre podías almacenar 80 caracteres de información. Ahora podías almacenar y recuperar decenas de miles de registros y tenerlos disponibles en un terminal de ordenador.

La programación estructurada postula el principio de división de un programa en unidades algorítmicas más pequeñas llamadas funciones o procedimientos. Una vez definidos, el programador puede invocar a un procedimiento desde otro. Típicamente un programa estructurado se realiza diseñando primeramente un conjunto de estructuras de datos, y posteriormente un conjunto de algoritmos (procedimientos) que actúan sobre esas estructuras de datos. Por ejemplo, si queremos desarrollar un programa para calcular la media de diversos conjuntos de variables, el programador desarrolla un procedimiento al que llama media donde implementa el algoritmo mediante el cual se calcula la media para cualquier conjunto de variables. Posteriormente invoca a ese procedimiento por cada conjunto de variables de los cuales quiera calcular la media. Algunos lenguajes que siguen metodología de la programación estructurada son el C, Pascal, Basic, Fortran, Cobol, etc.

Aunque la programación estructurada es una metodología que se ha usado durante mucho tiempo y aún se usa en la actualidad, existe un límite en la complejidad de los programas que se pueden realizar manteniendo los costes de mantenimiento y modificación de los programas razonablemente bajos. Es otras palabras, modificar un programa complejo en Pascal o COBOL puede ser una tarea altamente costosa en términos de tiempo. Para sobrepasar estos problemas se creó la metodología de la programación orientada a objetos, que postula la división de un programa en unidades autocontenidas llamadas objetos. A diferencia de los procedimientos que contienen sólo algoritmos y los datos sobre los que actúan están aparte, un objeto contiene un conjunto de algoritmos junto con los datos sobre los que actúan. De esta manera cuando un programador invoca algoritmos en un objeto, no necesita saber qué tipo de datos el objeto maneja de forma interna. Además, los objetos pueden ser organizados en jerarquías, de forma unos objetos pueden heredar datos y algoritmos de otros objetos. Con esto lo que se consigue es que la organización de un programa orientado a objetos sea más modular y rica que la organización de un programa estructurado, con lo que la arquitectura de los programas complejos puede ser acomodada a cambios más fácilmente.

PROGRAMACION ORIENTADA A OBJETOS (POO*)*

Con la introducción de sistemas operativos gráficos tales como Windows, ha surgido un nuevo concepto de programación. Los programadores ahora diseñan aplicaciones a base de unir diferentes piezas de código ya escrito y probado con anterioridad, cada una de estas piezas se llama "objeto". Los objetos pueden tener propiedades, tales como forma, tamaño, color, y tipo de datos. Un ejemplo podría ser un cuadro en la pantalla conteniendo una cuenta de dinero. Usted podría cambiar el tamaño de este cuadro, cambiar el color, cambiar la forma en la que se muestra la figura de la cuenta de dinero. Entonces usted podría realizar operaciones en esa cuenta, controlando los eventos que se hicieran sobre ella, tales como una pulsación del ratón que podría disparar una determinada operación, o mover la caja sobre la pantalla. Gracias al entorno del sistema operativo subyacente tras ello, usted puede hacer todo esto colocando propiedades en lugar de escribir todo un código. Y no sólo esto, usted puede crear una aplicación con una base de datos subyacente tras ella, utilizando todas las plantillas preprogramadas sobre la estructura de la base de datos. En los días del antiguo lenguaje ensamblador, tal sistema hubiera sido extraordinario, si no imposible, de llevarlo a la práctica, además el tamaño y la complejidad serían horrendos. Pero mucha de esa complejidad está ya realizada en el sistema operativo de los ordenadores que ejecutan estos programas "orientados a objeto". Lo bueno de esto es que muchas de las funciones que antes llevaban meses para programarlas, hoy se pueden hacer fácilmente en unos días, incluso en cuestión de horas. Además, el coste económico de los sistemas operativos de ordenador se ha revolucionado, debido a que los costes del hardware caen en picado. Hoy día es normal que un ordenador personal medio tenga 64 megas de memoria RAM y 8 gigas de disco en una máquina que cuesta 1/40 de lo que costaba nuestro pobre sistema Univac en 1978. Esta tendencia no muestra tendencias de ir a cambiar La Orientación a Objetos (OO) se configura como la metodología de desarrollo de software predominante para los próximos años. Según algunos estudios recientes, más del 50% de las corporaciones que desarrollan aplicaciones informáticas, a nivel mundial, ya han abordado de alguna manera estas técnicas, empezando por proyectos piloto con el fin de explorar las ventajas conseguidas con ellas. El soporte que todas las empresas informáticas están dando a la OO (desde fabricantes de ordenadores, bases de datos, hasta lenguajes tanto de tercera como de cuarta generación) hace que la dificultad de migrar a esta tecnología sea cada vez menor.

La Orientación a Objetos (OO), que inicialmente fue un conjunto de técnicas de programación soportadas en el uso de lenguajes especiales (orientados a objetos), ha ido poco a poco más allá de la propia programación hasta convertirse en una metodología genérica y de gran potencia para construir modelos de sistemas, que puede ser aplicada en todas las fases del desarrollo de aplicaciones: análisis, diseño, programación y mantenimiento.

Frente a otras metodologías tiene la ventaja de ser más natural (más próxima a la forma de pensar y hablar de las personas) e integrar los principios generales de la ingeniería del software en un paradigma coherente (el concepto de "objeto").

Los conceptos fundamentales de Objetos están en la actualidad bien asentados. A continuación, se citan los más importantes, desde nuestro punto de vista. Un Modelo

de Objetos es un conjunto de entidades (denominadas objetos) que colaboran entre ellos para desempeñar una serie de servicios. Esos servicios se solicitan por medio del intercambio de mensajes. Todos los objetos del modelo pertenecen a algún tipo (Clase).

En cuanto a los Objetos en sí, el principio fundamental es que un Objeto es la representación de un concepto. Como tal, tendrá unas características (atributos) y un comportamiento, plasmado en una serie de operaciones.

Pero, si bien estos conceptos son aceptados generalmente, en cambio en las metodologías para construir modelos de objetos hay grandes diferencias entre las distintas escuelas, lo cual está perjudicando en cierta medida la implantación de las técnicas OO, ya que se les achaca aún una cierta inmadurez.

Aunque la programación estructurada es una metodología que se ha usado durante mucho tiempo y aún se usa en la actualidad, existe un límite en la complejidad de los programas que se pueden realizar manteniendo los costes de mantenimiento y modificación de los programas razonablemente bajos. Es otras palabras, modificar un programa complejo en Pascal o COBOL puede ser una tarea altamente costosa en términos de tiempo. Para sobrepasar estos problemas se creó la metodología de la programación orientada a objetos, que postula la división de un programa en unidades autocontenidas llamadas objetos. A diferencia de los procedimientos que contienen sólo algoritmos y los datos sobre los que actúan están aparte, un objeto contiene un conjunto de algoritmos junto con los datos sobre los que actúan. De esta manera cuando un programador invoca algoritmos en un objeto, no necesita saber qué tipo de datos el objeto maneja de forma interna. Además, los objetos pueden ser organizados en jerarquías, de forma unos objetos pueden heredar datos y algoritmos de otros objetos. Con esto lo que se consigue es que la organización de un programa orientado a objetos sea más modular y rica que la organización de un programa estructurado, con lo que la arquitectura de los programas complejos puede ser acomodada a cambios más fácilmente.

Conceptos fundamentales de POO

La POO es el método de implementación en el que los programas se organizan como colecciones cooperativas de objetos, cada uno de los cuales representa un ejemplar de una clase y cuyas clases son miembros de una jerarquía de clases unidas mediante relaciones de herencia.

Objeto:

- Entidad que contiene los atributos que describen el estado de un objeto del mundo real y las acciones que se asocian con el objeto del mundo real.
- Se designa con un nombre o un identificador.
- Objeto = Datos + Operaciones

- Los datos deberían estar ocultos en el objeto, y las operaciones serían la interface del objeto con el exterior, pero estas operaciones están encapsuladas en "cajas negras".

Conclusiones:

- La unidad lógica de programación es el objeto.
- Los objetos tienen relaciones.
- Existen clases que agrupan conceptualmente los objetos.
- Las clases pertenecen a una jerarquía (son las clases las que heredan y no los objetos).

MODULARIZACIÓN

Consiste en descomponer un programa en un número pequeño de abstracciones coherentes que pertenecen al dominio del problema y enmascaran su complejidad interna.

Existen muchas acepciones del concepto de modularidad, que van desde una subrutina hasta la asignación de trabajo para un programador.

En general, consiste en subdividir el software en partes denominadas módulos, cada uno con un propósito específico, que se integran para satisfacer los requerimientos de un problema. Es el atributo más sencillo del software, que permite a un programa ser más manejable.

Asimismo, es un enfoque comúnmente aceptado tanto para análisis como para diseño.

La modularidad proporciona:

- Calidad de diseño
- Facilidad de instrumentación
- Facilidad de depuración
- Facilidad de pruebas
- Facilidad de documentación
- Facilidad de mantenimiento