Algoritmos y estructuras de datos 1

Breve introducción a Java

* Java es un lenguaje de la programación orientado a objetos
* Se rige por la filosofía de “*Write once, run everywere*”, lo que quiere decir que es un lenguaje portable
* Al compilar el código, no se genera un código intermedio conocido como Byte Code
* La única máquina capaz de ejecutar byte code es la “Java Virtual Machine” (JVM)
* La JVM es especificada por Oracle, pero distribuida por varias empresas, o sea, existen varias “marcas” de la JVM (Oracle, IBM…)
* En una computadora pueden correr varias instancias de la JVM
* Por cada proceso de Java, se ejecuta una JVM aparte

Java+JVM+Utilidades para programar, conforman una plataforma de software para el desarrollo de aplicaciones de usuario final.

Existen varias plataformas de Java:

* SE: (Standard Edition) Para aplicaciones de escritorio
* ME: (Mobile Edition) Para móviles
* EE: (Enterprise Edition) Aplicación de servidor

Java se distribuye de varias formas:

* JRG: Mínimo para ejecutarse
* JDK: Java Development Kit JRG+SE
* Servidor de aplicaciones: Implementación de EE

. Java

JDK Compilador

.class

Hardware

JVM

Paradigmas de la programación

* En general, es una forma de ver/entender/moldear el mundo
* En el ámbito del software, es un estilo fundamental de programación, determina como el programa “ve” el mundo
* Determina como debe ser usado el lenguaje por el programador
* Algunos lenguajes son multiparadigmas

Existen 4 paradigmas principales:

* Paradigma Lógico
  + El mundo se modela mediante predicciones lógicas
  + Se aplican principios de matemática discreta
  + Poco usado en aplicaciones comerciales
  + Prolog es el principal lenguaje
* Paradigma Funcional
  + El mundo se modela por funciones matemáticas
  + Lenguaje 100% funcionales: Lisp Scheme
  + Permite declarar funciones, pasar funciones por parámetro, retornar funciones, etc.
  + Muchos lenguajes modernos han incorporado este paradigma por conveniencia
* Paradigma Imperativo
  + El mundo se modela por instrucciones
  + Ampliamente usado en aplicaciones comerciales
  + Tienden a ser muy eficientes, generalmente compilan en lenguaje máquina
  + Gramáticamente simples
  + Se consideran como niveles de abstracción bajo
  + Ejemplo: C, C++
* Paradigma Orientado a Objetos
  + El mundo se modela por objetos del mundo real y su interacción
  + Muy naturales para el ser humano
  + Facilitan la reutilización de código
  + Mayoritariamente no generan lenguaje maquina
  + Poseen muchas estructuras sintácticas
  + Ejemplo: Java, C++, Python, JavaScript

UML y patrones de diseño:

SDLC: Software Development Life Cycle (Ciclo de Desarrollo de Software) es un proceso de creación de software que se ordena en pasos para tener éxito en la creación de sistemas, los pasos son:

* Recopilación de requerimientos: Se pregunta a los clientes sus deseos para su futura plataforma
* Análisis de requerimientos: Se observan los deseos del cliente
* Diseño de Software: Se crea un prototipo con el código necesario en papel
* Implementación: Se programa el código diseñado
* Pruebas: Se le realizan pruebas especializadas al prototipo de código
* Instalación y mantenimiento: Una vez pasadas las pruebas realizadas, se puede instalar el programa y darle mantenimiento periódicamente

El objetivo del UML es poder realizar un software de calidad y esto se logra si este sirve, es mantenible, reusable y ajustable al tiempo y presupuesto

Existen varios métodos de realizar el SDLC:

**Modelo de Cascada:**

* + Se piensa un paso a la vez
  + Cada cierto tiempo se entrega un documento con el avance del proyecto
  + Fracasó porque tomaba mucho tiempo y dinero

**Modelo de Espiral:**

* El proyecto está compuesto por mini proyectos
* Se puede ir modificando el proyecto conforme se avanza en el
* Cada cierto tiempo se entrega una propuesta de software con el avance de este