Paradigmas de Programación

Metodología de la Programación

Ing. José Zapana

Equipo docente de la cátedra

		
Zapana, José Vidal	Tolaba, Ana Carolina	Ramos, Héctor Orlando
jose_zapana@fi.unju.edu.ar	atolaba@fi.unju.edu.ar	hramos@fi.unju.edu.ar

Avisos generales

- Formulario de matriculación para quienes no están registrados en el aula virtual: https://forms.gle/dmDnYsoPUxFS1pxz6
- La foto de perfil del aula virtual es obligatoria
- Clases de trabajos prácticos inician la semana del 19/08
- Inscripción a comisiones de trabajos prácticos: desde el 15-08-2024 10:00 hs hasta el 24-08-2024 a las 23:55 hs
- Es obligatorio leer el reglamento:
  - Feriados
  - Evaluaciones, etc.
- Revisar el panel “Inicio” del aula virtual

¿Qué es un paradigma de programación?

Los paradigmas de programación son enfoques o estilos de programación que proporcionan un marco conceptual para diseñar, estructurar y escribir programas de software.

Cada paradigma ofrece un conjunto de principios, reglas y técnicas para resolver problemas y expresar soluciones de manera efectiva en un lenguaje de programación específico.

Los lenguajes de programación están basados en uno o más paradigmas, por ejemplo:

- **Java** en POO
- **Haskell** en programación funcional
- **Pascal** en programación imperativa
- **Python** soporta múltiples paradigmas (orientado a objetos, imperativo, funcional)

Paradigma Imperativo

- Es un enfoque de programación donde el código describe una secuencia de instrucciones que cambian el estado del programa a través de asignaciones, bucles y control de flujo.
- En este paradigma, el énfasis está en **cómo** realizar tareas, especificando pasos concretos que la computadora debe seguir.

Paradigma Imperativo - Ejemplo

```python
def maximo(lista):
    return max(lista)

# Leer lista de números
numeros = list(map(int, input("Ingrese números separados por espacios: ").split()))

# Calcular y mostrar el máximo
print(f"El valor máximo es: {maximo(numeros)}")
```

Características fundamentales

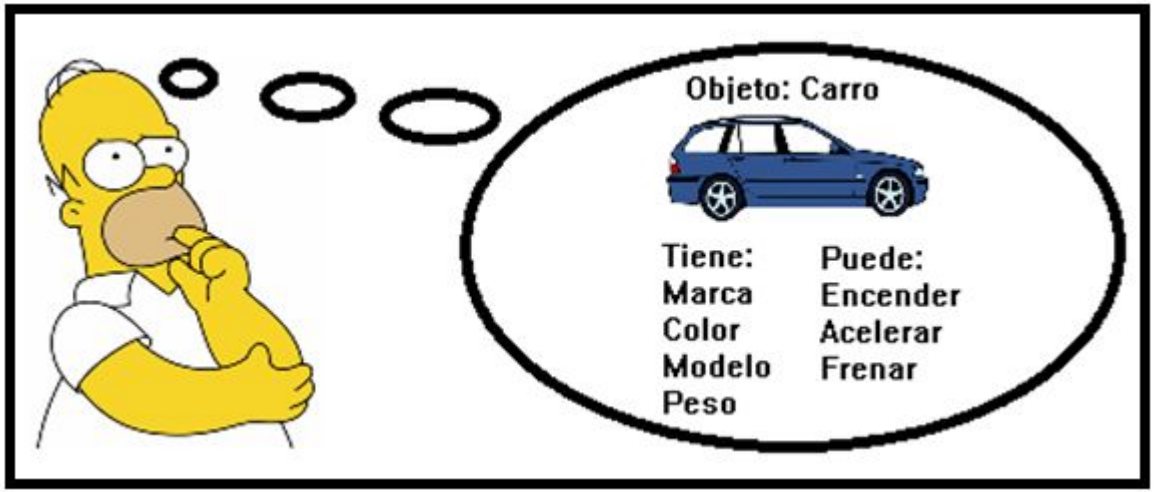
- Concepto de celda de memoria para almacenar valores. El componente principal de la arquitectura es la memoria, compuesto por un gran número de celdas.
- Operaciones de asignación. Estrechamente ligado a la arquitectura de memoria, cada valor calculado debe ser almacenado en una celda.
- Repetición. Un programa imperativo normalmente realiza su tarea ejecutando repetidamente una secuencia de pasos fundamentales.

Lenguajes del paradigma imperativo

- Los primeros fueron denominados de bajo nivel (Assembler)
- Los primeros de alto nivel fueron Fortran y Cobol.
- Posteriormente aparecieron Pascal y Algol-68
- La aparición del lenguaje C marcó un hito importante en los lenguajes imperativos.
- El paradigma fue dominante hasta mediados de los 90.
- Otros lenguajes: Pascal, C, Fortran, Algol, Ada, Clipper, Fox, PL/1, etc.

Paradigma Orientado a Objetos

- Está basado en la idea de encapsular el **estado** y **comportamiento** en objetos.
- Considera al mundo como un conjunto de entidades u objetos que se comunican entre sí a través de mensajes
- Objetos y clases son los conceptos fundamentales
- Sus pilares son la herencia, polimorfismo, abstracción y encapsulamiento



Paradigma Orientado a Objetos - Ejemplo

```java
public class Vehiculo {
    String marca;
    String color;
    String modelo;
    double peso;

    public void encender() {
        System.out.println("El vehículo está encendido.");
    }

    public void acelerar() {
        System.out.println("El vehículo está acelerando.");
    }

    public void frenar() {
        System.out.println("El vehículo está frenando.");
    }
}
```

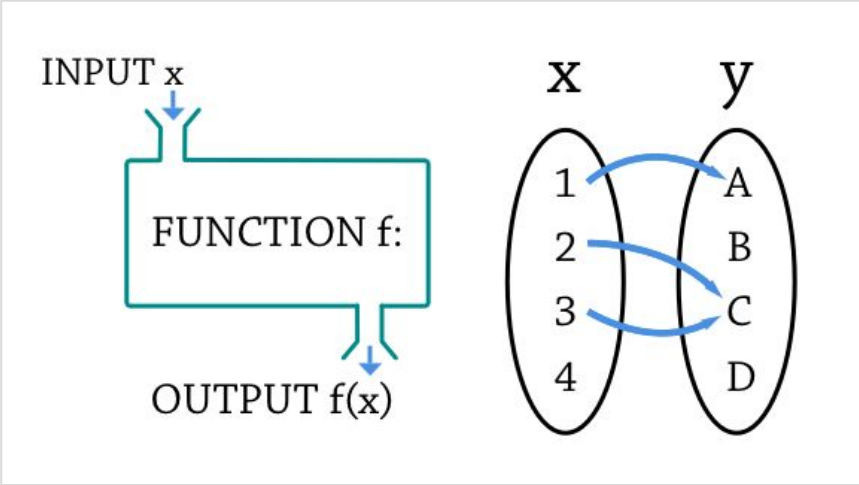
```java
public class Main {
    public static void main(String[] args) {
        Vehiculo miVehiculo = new Vehiculo("Toyota", "Rojo",
                                           "Corolla", 1500.0);

        miVehiculo.encender();
        miVehiculo.acelerar();
        miVehiculo.frenar();
    }
}
```

Paradigma Orientada a Objetos - Historia

- Surge en base al paradigma imperativo, provocando un giro importante en sus principios y consideraciones básicas.
- Smalltalk es considerado el primero de los lenguajes orientados a objetos, publicado en la década del 80 pero tuvo poca aceptación
- En 1995 se lanzó Java que logró gran aceptación y masividad aún en la actualidad.

Paradigma Funcional

- Está basada en el modelo matemático de composición funcional. Es decir, el resultado de un cálculo es la entrada del siguiente, y así sucesivamente hasta que una composición produce el valor deseado.
- No existe el concepto de memoria que es asignada o modificada. Más bien existen valores intermedios que son el resultado de cálculos anteriores y las entradas a cálculos subsiguientes.
- Tampoco existen sentencias imperativas y todas las funciones tienen transparencia referencial.



Paradigma funcional ejemplo

```
(defun factorial (n)
  (if (<= n 1)
      1
      (* n (factorial (- n 1)))))

(write-line "Ingresa un número:")
(setq num (read))

(write-line "El factorial es:")
(write (factorial num))
```

Paradigma funcional - Ámbitos de uso

- **Procesamiento de datos masivos:** Gracias a la capacidad de manejar funciones puras y la inmutabilidad
- **Desarrollo de software concurrente y paralelo:** La inmutabilidad y la ausencia de efectos secundarios facilitan la creación de aplicaciones que pueden ejecutarse en paralelo sin riesgos de condiciones de carrera.
- **Desarrollo de sistemas financieros:** Se utiliza para crear aplicaciones seguras y robustas, donde la inmutabilidad ayuda a evitar errores críticos.
- **Desarrollo de aplicaciones web y servidores:** Utilizado en backends escalables y altamente concurrentes, por ejemplo, con frameworks funcionales en Scala (Play) o Elixir (Phoenix)

Aplicaciones desarrolladas con paradigma funcional

- **Twitter:** Utiliza Scala para manejar grandes volúmenes de datos y procesamiento en tiempo real.
- **LinkedIn:** Utiliza Kafka (escrito en Scala) para la gestión de flujos de datos en tiempo real.
- **Facebook:** Utiliza Haskell en algunos de sus sistemas de anti-spam y prevención de fraude.
- **WhatsApp:** Su backend está escrito en Erlang, un lenguaje funcional, lo que le permite manejar millones de conexiones simultáneas de manera eficiente.

Paradigma Lógico

- Se basa en la declaración de **hechos** y **reglas**, utilizando la lógica para resolver problemas.
- En lugar de especificar cómo se debe ejecutar un algoritmo, el programador define **qué es verdad** y **qué debe cumplirse**. Un motor de inferencia busca soluciones que satisfagan esas condiciones.
- Es utilizado principalmente en inteligencia artificial, sistemas expertos y bases de datos.
- Esta forma de tratamiento de la información permite pensar la existencia de “programas inteligentes” que puedan responder, no por tener en la base de datos todos los conocimientos, sino por poder inferirlos a través de la deducción.

Paradigma lógico - Funcionamiento

- Un programa lógico contiene una base de conocimiento sobre la que se hacen consultas.
- La base de conocimiento está formada por hechos, que representan la información del sistema expresada como relaciones entre datos, y reglas lógicas que permiten deducir consecuencias a partir de combinaciones entre los hechos y, en general, otras reglas.
- Una clave de la programación lógica es poder expresar apropiadamente todos los hechos y reglas necesarios que definen el dominio de un problema.

Prolog - Ejemplo

```
% Hechos
es_padre(juan, maria).
es_padre(juan, jose).

% Regla
es_hermano(X, Y) :- es_padre(P, X), es_padre(P, Y), X \= Y.

% Consulta
% ¿Quién es hermano de María?
% es_hermano(maria, Quien).
```

es_hermano(X, Y) SI:
- X e Y tienen el mismo padre P, y
- X no es igual a Y.

Hechos: Se declara que Juan es el padre de María y de José.
Regla: Se define que dos personas son hermanos si tienen el mismo padre y no son la misma persona.
Consulta: Se puede preguntar al sistema quién es hermano de María.

Prolog - Usos en la actualidad

- **Sistemas de asistencia médica:** se utiliza en aplicaciones médicas para el diagnóstico y tratamiento de enfermedades
- **Sistemas de planificación y control de robots:** donde se pueden definir reglas lógicas para la toma de decisiones y la coordinación de acciones del robot.
- **Análisis de lenguaje natural:** es utilizado en el análisis de lenguaje natural para tareas como el análisis gramatical, el procesamiento de texto y la generación de lenguaje natural.
- **Sistemas de lógica difusa:** La programación difusa se utiliza para modelar sistemas y tomar decisiones en situaciones donde la información es incompleta, incierta o vaga. Permite realizar razonamientos aproximados y tomar decisiones basadas en la incertidumbre y la imprecisión de los datos disponibles. Se pueden definir reglas lógicas difusas para representar y procesar información incierta.

Otros Paradigmas

- Existen otras clasificaciones tales como: programación orientada a eventos, orientada a aspectos, basada en restricciones, basada en reglas, etc.
- Para algunos autores se tratan sólo de subparadigmas

Conclusiones

- Ningún paradigma es capáz de resolver todos los problemas de forma sencilla y eficiente, por lo tanto es útil poder elegir entre distintos estilos de programación dependiendo del tipo de problema
- Por otra parte, hay lenguajes que permiten mezclar distintos paradigmas. Por ejemplo el lenguaje Oz emplea programación lógica, funcional y orientada a objetos.
- Otros lenguajes como Delphi, C y Visual Basic combinan el paradigma imperativo y orientado a objetos
- Por ejemplo Facebook utiliza: Hack, Java, C++, Erlang, Python, Haskell, Dlang, PHP, JavaScript y el framework React.