Programación Declarativa, 2023-2
Nota de clase 01: **Introducción al curso**

Manuel Soto Romero

Facultad de Ciencias UNAM

Seguramente en cursos previos de la licenciatura, has trabajado con lenguajes que pertenecen al estilo de programación declarativo. Por ejemplo en Estructuras Discretas, Lógica Computacional o Lenguajes de Programación e incluso en otras materias. Sin embargo, en este curso profundizaremos en los conceptos y tecnicismos de este tipo de lenguajes de programación en lugar de usarlos únicamente como una herramienta para resolver problemas particulares como la implementación de intérpretes o de algunas funciones de los principales sistemas lógicos.

En esta nota revisaremos brevemente el concepto de estilo de programación haciendo especial énfasis en el estilo declarativo con el fin de exponer las principales ventajas de trabajar con este tipo de lenguajes de programación y que estudiaremos con mucho más detalle a lo largo del curso.

Estilos de programación

Recordemos que un estilo de programación, también llamado paradigma de programación, representa la visión que tiene el programador sobre la forma de atacar un problema particular. Esto nos permite, entre otras cosas, resolver un mismo problema de distintas maneras y obtener un mismo resultado.

Ejercicio

Supón que estás vuelves a aquellos tiempos en los que estudiabas la secundaria o el bachillerato y tu profesor de matemáticas te pide resolver la siguiente ecuación de segundo grado:

$$3x^2 + 2x - 3 = 0$$

¿Puedes dar solución a esta ecuación? ¿Qué método usarías?

De acuerdo con la actividad anterior, es probable que hayas utilizado uno de muchos métodos que existen para resolver la ecuación. Por ejemplo, el famoso método de *la chicharronera* o aplicando una simple factorización. Sin embargo, sin importar el método elegido para resolver la ecuación, el resultado es el mismo. Esto es justo lo que ocurre con los estilos de programación, la forma en que atacamos un problema como programadores cambia dependiendo del estilo de programación que use el lenguaje.

En general, los estilos de programación se dividen en dos categorías:

1. Programación imperativa
2. Programación declarativa

A continuación se revisan las principales diferencias entre éstos.

Programación imperativa

En el estilo de programación imperativo, los programas son representados como órdenes dadas a la computadora. Es decir, lxs programadorxs especifican paso a paso cómo resolver el problema.

Ejercicio

Imagina que eres el empleado de una cafetería y te piden preparar un capuchino. ¿Cómo prepararías esta bebida usando el estilo imperativo? Escribe tu solución.

Dentro de este estilo de programación se encuentran dos representantes muy importantes que son la *Programación Estructurada* y la *Programación Orientada a Objetos*.

Programación estructurada

Este estilo de programación se basa en el *Teorema de Böhm–Jacopini*, también llamado *Teorema del Programa Estructurado* que es un resultado de la Teoría de Lenguajes de Programación que dice que todo problema computable puede resolverse con únicamente tres estructuras:

- Secuencias
- Estructuras de decisión
- Estructuras de repetición

Si un lenguaje incluye al menos estas tres estructuras, podemos decir entonces que es Turing-Completo. El principal representante de este estilo es el lenguaje de programación C que incluye las tres estructuras. Por ejemplo:

```c
#include <stdio.h>

int main(void)
{
    int n;
    int s = 0;

    printf("Introduce un número: ");
    scanf("%d", &n);

    if (n > 0)
    {
        for (int i = 0; i < n; i++)
            s += i;
        printf("\nLa suma de los primeros %d
            naturales es: %d", n, s);
    } else
        printf("\nEl número debe ser positivo
            .\n");

    return 0;
}
```

Notamos en este ejemplo otra característica de este tipo de lenguajes y es el uso de la operación de asignación que es de gran importancia en este tipo de programas pues de esto depende el flujo de ejecución de la mayoría de estructuras de repetición.

Programación orientada a objetos

En este estilo la programación se basa en la definición y uso de distintos entes del mundo real conocidos como *objetos*. Un objeto tiene un conjunto de características particulares que lo hacen diferente de los demás (atributos) y un comportamiento asociado, es decir, las acciones que puede realizar dicho objeto (métodos).

Para construir objetos y agruparlos se usan clases. Algunas de sus principales características son la posibilidad de heredar comportamiento entre clases mediante el mecanismo de herencia y el de polimorfismo lo cual permite la reutilización de código, entre muchas otras cosas. Por ejemplo:

```
public class Persona {

   private String nombre;
   private int edad;

   public void setNombre(String nombre) {
      this.nombre = nombre;
   }

   public void setEdad(int edad) {
      this.edad = edad;
   }

   public String getNombre() {
      return this.nombre;
   }

   public String getEdad() {
      return this.edad;
   }
}
```

Programación declarativa

En el estilo de programación declarativo, en lugar de indicarle cómo resolver un problema a la computadora, se dan una serie de definiciones (declaraciones) y se usan en conjunto para llegar a la solución, es decir, nos enfocamos en el qué. Algo similar a lo que ocurre en el desarrollo de distintas teorías matemáticas. Usamos definiciones en la construcción y prueba de teoremas y después usamos estos resultados para resolver problemas particulares.

Ejercicio

Volviendo al problema de cómo preparar un capuchino. ¿Cómo prepararías esta bebida usando el estilo declarativo? Escribe tu solución. Recuerda que en este estilo, no debes indicar paso a paso la solución sino usar definiciones.

Dentro de este estilo de programación se encuentran dos representantes muy importantes que son la *Programación Lógica* y la *Programación Funcional* que estudiaremos con mucho detalle en este curso.

Programación lógica

La programación lógica se basa en la lógica de primer orden, principalmente en las Cláusulas de Horn para representar el conocimiento. En este sentido, la programación se basa en la definición de hechos y reglas y en lugar de ejecutar programas se realizan consultas mediante metas que son resueltas a través de resolución binaria y retroceso (*backtracking*).

El principal representante de este estilo de programación es PROLOG. Por ejemplo:

```
nat(0).
nat(s(x)) :- nat(x).
```

El código anterior muestra la definición de los números naturales. Una posible consulta podría ser `?- nat(1)`, con lo cual PROLOG respondería `true`. Podemos apreciar el uso de definiciones para llegar a una solución y en este caso también el uso de la recursión como método de repetición.

Programación funcional

La programación funcional se basa en el Cálculo λ y la Teoría de Categorías. En este estilo la programación se compone de la definición de funciones que actúan como *miembros de primera clase*, es decir, que se comportan como cualquier otro valor en el lenguaje.

Los cálculos se dan aplicando reducciones y combinando funciones ya sea pasándose entre ellas como parámetro, devolviéndolas como resultando o incluso componiéndolas tal y como se hace en matemáticas. El principal representante de este estilo es HASKELL. Por ejemplo:

```
suma :: Int → Int
suma 0 = 0
suma n = n + suma (n-1)
```

El código anterior muestra la definición de una función que suma los primeros n naturales. Una forma de ejecutar la función podría ser `suma 5`. Al igual que con el código de PROLOG se aprecia el uso de definiciones y de la recursión.

Es importante notar que ambos códigos son significativamente más cortos que los dos imperativos.

Beneficios de la programación declarativa

Una vez revisadas estas diferencias, resulta natural preguntarnos acerca de los beneficios que trae consigo la programación *declarativa*. A continuación se mencionan algunos: [1]

1. La programación declarativa no depende del lenguaje en particular.
2. Los programas imperativos son rápidos y especializados. Sin embargo, un programa declarativo es usualmente, más general, corto y legible.
3. Los programas declarativos son elegantes matemáticamente hablando. Lo cual implica que es más fácil verificar si el programa cumple su especificación.
4. Aprender programación declarativa permite al programador desarrollar un estilo de programación riguroso y disciplinado que puede ser usado ventajosamente sin importar el lenguaje de programación elegido.
5. Este estilo genera programas con una mejor ingeniería, más fáciles de depurar, mantener y modificar.

Por supuesto, revisaremos más ventajas a lo largo del curso y haremos más notables algunas de éstas.

Referencias

[1] Notas de Clase para el Curso de Programación Funcional y Lógica, Favio E. Miranda, et. al. Facultad de Ciencias UNAM, Revisión 2012-1. (Nota 1: `https://sites.google.com/site/pfyl121unam/recursos/pfyl121n1.pdf`)