APLICANDO LO APRENDIDO 0

Alumno: Joaquín Elías Medero
Objetivos: Analizar lenguajes de programación en base a la lógica paradigmática.

Ejercicio 1

Considera un lenguaje que conozcas bien y analízalo en términos de los 4 componentes de un paradigma mencionados por Kuhn.

1. Generalización simbólica: ¿Cuáles son las reglas escritas del lenguaje?
2. Creencias de los profesionales: ¿Qué características particulares del lenguaje se cree que sean "mejores" que en otros lenguajes?
3. Valores: ¿Qué pensamiento o estilo de programación consideraron mejor los creadores
4. Ejemplares: ¿Qué clase de problemas pueden resolverse más fácilmente en el lenguaje?

   Si conoces más de un lenguaje, repite el ejercicio, comparando este segundo lenguaje con el primero.

Respuesta

Lenguaje elegido: C

1) Generalización simbólica:

- El programa que se cree en lenguaje C se empieza ejecutando desde la apertura de la función main (), el compilador busca esta sentencia y empieza a ejecutar el programa a partir de esa línea, y para indicar que el programa termina exitosamente debe retornar cero.
- Las variables se deben declarar antes de ser utilizadas, y tienen un ámbito de vida local, global, o dentro de bloques de procesos.
- Utilización de estructuras de control condicionales simples (if y else), condicionales múltiples (switch), repetitivas definidas (for) e indefinidas (while, do-while)
- Las sentencias terminan con un punto y coma a menos que se necesite abrir una llave al final de la sentencia, como es en el caso de un if. Ej

  ```
  if(variable_entera==1)

  {

  }
  ```

- Los programas deben incluir bibliotecas externas de C para trabajar con distintas funciones como scanf y printf, o para manipulación de cadenas de caracteres como la función que compara entre dos strings: strcmp.

2) Creencias de los profesionales:

El lenguaje C es mejor corriendo programas rápidamente a diferencia de otros lenguajes por su nivel cercano al ensamblado, que le permite manejar menos recursos que otros que son de alto nivel. También ofrece un manejo de memoria directa a través de punteros. Además ofrece una alta portabilidad, se pueden ejecutar programas en C en muchas plataformas debido a su estandarización. Y por último se destaca su sintaxis sencilla de aprender y útil para trabajar en distintos ámbitos.

3) Valores:

El lenguaje C está orientado hacia una programación estructurada, debido a su manejo del flujo de ejecución y el uso de bloques para modularizar el programa en miniprogramas, usando el refinamiento sucesivo, permitiendo resolver un problema grande en distintas partes que ejecuten pequeñas acciones que sean independientes del resto del código y faciliten el mantenimiento del mismo.

4) Ejemplares:

C es ideal para problemas que requieran un control cercano al hardware, como el desarrollo de sistemas operativos, controladores de dispositivos, o cualquier aplicación que necesite manipular directamente la memoria y los recursos del sistema.

Análisis con Java:

1) Generalización simbólica:

Java tiene una sintaxis rigurosa basada en la orientación a objetos. Cada clase debe pertenecer a un archivo y el archivo debe tener el mismo nombre que la clase pública. los programas hechos en java se ejecutan desde un archivo ejecutable que contenga la función public static void main (String args[]). Además es fuertemente tipado: Las variables y objetos deben ser declarados con un tipo específico.

2) Creencias de los profesionales:

Una de las creencias clave sobre Java es que su código es altamente portable debido a la JVM (Java Virtual Machine), lo que permite que el mismo código corra en diferentes plataformas sin modificaciones. Este lema se conoce como "Write once, run anywhere".

Los profesionales consideran que el paradigma orientación a objetos en Java está mejor implementado y es más estricto en comparación con lenguajes como C + +, lo que fomenta una mejor organización del código.

3) Valores:

Los creadores de Java querían un lenguaje que promoviera el código claro, fácil de leer y mantenible. Por eso introdujeron un sistema de tipos estrictos y una sintaxis uniforme. El diseño de Java está fuertemente orientado a evitar errores comunes en lenguajes como C, como el acceso ilegal a memoria o la gestión manual de memoria. Se enfatiza en la robustez a través del manejo automático de memoria y el manejo de excepciones obligatorio.

4) Ejemplares:

Java es excelente para desarrollar aplicaciones empresariales y web, donde la modularidad y la portabilidad son clave. Los frameworks como Spring facilitan el desarrollo de aplicaciones de gran escala, y su uso en Android lo hace un lenguaje muy viable para el desarrollo móvil.

Ejercicio 2

Considera un lenguaje que conozcas bien y analízalo en términos de los ejes propuestos para la elección de un lenguaje de programación y responde:

1. ¿Tiene una sintaxis y una semántica bien definida? ¿Existe documentación oficial?
2. ¿Es posible comprobar el código producido en ese lenguaje?
3. ¿Es confiable?
4. ¿Es ortogonal?
5. ¿Cuáles son sus características de consistencia y uniformidad?
6. ¿Es extensible? ¿Hay subconjuntos de ese lenguaje?
7. El código producido, ¿es transportable?

Respuesta

Lenguaje elegido: Java

1) Si, Java tiene una sintaxis y semántica bien definida. Sigue reglas estrictas que controlan los códigos en los programas para ver si cumplen con la sintaxis. Hay una documentación oficial extensa, proporcionada por Oracle, que cubre desde la sintaxis básica hasta las bibliotecas avanzadas (Java API).

2) Sí, Java permite la comprobación del código en varios niveles. Los errores de sintaxis se detectan durante la compilación, mientras que los errores semánticos se detectan en tiempo de ejecución mediante el uso de excepciones. Además, existen herramientas externas como **JUnit** para realizar pruebas unitarias, lo que permite verificar el comportamiento del código de forma automatizada.

3) Si, de hecho Java es altamente confiable. Es conocido por su fuerte manejo de excepciones, recolección automática de basura y administración de memoria, lo que

previene errores comunes como fugas de memoria. Su arquitectura también se diseñó para evitar peligros de seguridad.

4) Java es parcialmente ortogonal porque muchos conceptos (como el manejo de control de flujo, tipos de datos, y estructuras) se pueden combinar sin causar confusión. Por ejemplo, se puede combinar bucles con manejo de excepciones sin problemas. A pesar de eso, no es completamente ortogonal en ciertos aspectos. Un ejemplo es la distinción entre tipos primitivos (como `int o char`) y objetos (como `Integer o String`). Los primitivos no se comportan como objetos y, en algunos casos, se requiere conversión manual entre ellos, lo que rompe la consistencia de las características del lenguaje al ser combinadas.

5) Java tiene las siguientes características de uniformidad y consistencia:
    - Los objetos(instancias de clases) y las clases son los bloques de construcción fundamentales, creando un estándar de elementos para programar.
    - Todas las clases en Java heredan directa o indirectamente de la clase base `Object`, lo que establece una jerarquía uniforme. Esto significa que cada clase tiene métodos básicos comunes como `toString()`, `equals()`, y `hashCode()`, lo que crea consistencia al trabajar con cualquier tipo de objeto.
    - Sistema unificado de manejo de excepciones basado en `try-catch-finally`. Todas las excepciones, sin importar su origen (errores de archivo, división por cero, etc.), se manejan con el mismo patrón.
    - Una tipificación estricta y estática. Esto significa que todas las variables deben declararse con un tipo específico antes de ser usadas, y este tipo no puede cambiar durante la ejecución.

6) Extensibilidad significa que puedes ampliar el lenguaje mediante la creación de nuevas bibliotecas, frameworks, y clases sin tener que cambiar el propio lenguaje. Java es muy extensible, ya que te permite crear tus propias bibliotecas de clases que otros pueden usar, y utilizar frameworks como Spring, Hibernate y Junit.

Hay subconjuntos del lenguaje Java, como:

- Java ME (Micro Edition) para dispositivos móviles y embebidos con menos recursos.
- Java EE (Enterprise Edition), una extensión para desarrollar aplicaciones empresariales a gran escala.

7) Sí, los códigos producidos son transportables en Java. Gracias a la Máquina Virtual de Java (JVM), el código Java compilado puede ejecutarse en cualquier plataforma que tenga una JVM, lo que lo convierte en un lenguaje muy portátil.