Practica Nro 1

Objetivo: Conocer la evolución de los lenguajes de programación y sus características.

Ejercicio 1: Los lenguajes de programación más representativos son:

1951 - 1955: Lenguajes tipo assembly 1956 - 1960: FORTRAN, ALGOL 58, ALGOL 60, LISP 1961 - 1965: COBOL, ALGOL 60, SNOBOL, JOVIAL 1966 - 1970: APL, FORTRAN 66, BASIC, PL/I, SIMULA 67, ALGOL-W 1971 - 1975: Pascal, C, Scheme, Prolog 1976 - 1980: Smalltalk, Ada, FORTRAN 77, ML 1981 - 1985: Smalltalk 80, Turbo Pascal, Postscript 1986 - 1990: FORTRAN 90, C++, SML 1991 - 1995: TCL, PERL, HTML 1996 - 2000: Java, Javascript, XML

Indique para cada uno de los períodos presentados cuales son las características nuevas que se incorporan y cual de ellos la incorpora.

Ejercicio 2: Escriba brevemente la historia del lenguaje de programación que eligió en la encuesta u otro.

Ejercicio 3: ¿Qué atributos debería tener un buen lenguaje de programación? Por ejemplo, ortogonalidad, expresividad, legibilidad, simplicidad, etc. De al menos un ejemplo de un lenguaje que cumpla con estos atributos.

Ejercicio 4: Tome uno o dos lenguajes de los que ud. Conozca y describa los tipos de expresiones que se pueden escribir en él/ellos; Describa las facilidades provistas para la organización del programa; Indique cuáles de los atributos del ejercicio anterior posee el/los lenguaje/s elegidos y cuáles no posee, justifique en cada caso.

Lenguajes - ADA

Ejercicio 5: Describa las características más relevantes de Ada, referida a:

Tipos de datos; Tipos abstractos de datos – paquetes; Estructuras de datos; Manejo de excepciones; Manejo de concurrencia

Lenguajes - JAVA

Ejercicio 6: Diga para qué fue, básicamente, creado Java.¿Qué cambios le introdujo a la Web?

¿Java es un lenguaje dependiente de la plataforma en dónde se ejecuta? ¿Por qué?

Ejercicio 7: ¿Sobre qué lenguajes está basado?

Ejercicio 8: ¿Qué son los applets? ¿Qué son los servlets?

Lenguajes - C

Ejercicio 9: ¿Cómo es la estructura de un programa escrito en C? ¿Existe anidamiento de funciones?

Ejercicio 10: Describa el manejo de expresiones que brinda el lenguaje.

Lenguajes - Python - RUBY - PHP

Ejercicio 11: ¿Qué tipo de programas se pueden escribir con cada uno de estos lenguajes? ¿A qué paradigma responde cada uno? ¿Qué características determinan la pertenencia a cada paradigma?

Ejercicio 12: Cite otras características importantes de Python, Ruby, PHP, Gobstone y Processing. Por ejemplo: tipado de datos, cómo se organizan los programas, etc.

Lenguaje Javascript

Ejercicio 13: ¿A qué tipo de paradigma corresponde este lenguajes? ¿A qué tipo de Lenguaje pertenece?

Ejercicio 14: Cite otras características importantes de javascript. Tipado de datos, excepciones, variables, etc.

Ejercicio 1:

1951 - 1955: Lenguajes tipo Assembly

Característica nueva: Uso de mnemónicos en lugar de código máquina puro.

1956 - 1960: FORTRAN, ALGOL 58, ALGOL 60, LISP

Característica nueva:

Primeros lenguajes de alto nivel. (FORTRAN)

Estructuras de control como bucles y condicionales. (ALGOL 58)

Definición clara de estructuras de bloque y alcance de variables. (ALGOL 60)

Introducción de la programación funcional y listas enlazadas. (LISP)

1961 - 1965: COBOL, ALGOL 60, SNOBOL, JOVIAL

Característica nueva:

Enfoque en la programación orientada a negocios con sintaxis cercana al inglés. (COBOL)

Manejo de cadenas y patrones de texto. (SNOBOL)

Expansión del uso de estructuras de control y tipos de datos estructurados. (JOVIAL)

1966 - 1970: APL, FORTRAN 66, BASIC, PL/I, SIMULA 67, ALGOL-W

Característica nueva:

Introducción de operadores vectoriales y matriciales. (APL)

Ampliación de estándares en programación científica. (FORTRAN 66)

Accesibilidad a la programación para principiantes. (BASIC)

Lenguaje multipropósito con concurrencia. (PL/I)

Primer lenguaje con clases y objetos, base de la POO. (SIMULA 67)

Estructuras de datos mejoradas y control de flujo más expresivo. (ALGOL-W)

1971 - 1975: Pascal, C, Scheme, Prolog

Característica nueva:

Tipado fuerte y estructurado, promoviendo buenas prácticas. (Pascal)

Uso de punteros y acceso a memoria de bajo nivel. (C)

Programación funcional con soporte para recursión avanzada. (Scheme)

Primer lenguaje de programación lógica basado en inferencia. (Prolog)

1976 - 1980: Smalltalk, Ada, FORTRAN 77, ML

Característica nueva:

Primera implementación real de la Programación Orientada a Objetos (POO). (Smalltalk)

Lenguaje altamente estructurado con fuerte tipado y concurrencia. (Ada)

Ampliación de FORTRAN con mejoras en manipulación de caracteres y estructuras de datos. (FORTRAN 77)

Inferencia de tipos y programación funcional avanzada. (ML)

1981 - 1985: Smalltalk 80, Turbo Pascal, Postscript

Característica nueva:

Refinamiento de la POO con un entorno completamente orientado a objetos. (Smalltalk 80)

Introducción de compilación rápida e integración con entornos de desarrollo. (Turbo Pascal)

Lenguaje de descripción de páginas con enfoque en gráficos. (Postscript)

1986 - 1990: FORTRAN 90, C++, SML

Característica nueva:

Introducción de arreglos dinámicos y control estructurado. (FORTRAN 90)

Programación orientada a objetos con herencia y polimorfismo. (C++)

Refinamiento de la inferencia de tipos en programación funcional. (SML)

1991 - 1995: TCL, PERL, HTML

Característica nueva:

Creación de lenguajes de scripting para automatización y extensibilidad. (TCL)

Manejo eficiente de texto y expresiones regulares. (PERL)

Estándar de mercado para estructurar información en la web. (HTML)

1996 - 2000: Java, JavaScript, XML

Característica nueva:

Seguridad y portabilidad con ejecución en máquinas virtuales. (Java)

Interactividad en navegadores web con ejecución del lado del cliente. (JavaScript)

Estandarización del intercambio de datos estructurados. (XML)

Ejercicio 2:

JAVA:

Java fue desarrollado por James Gosling, Mike Sheridan y Patrick Naughton en Sun Microsystems en 1991 bajo el proyecto llamado Oak. Inicialmente, el objetivo era crear un lenguaje para dispositivos electrónicos y electrodomésticos inteligentes, pero el proyecto evolucionó hacia un lenguaje de programación más general.

En 1995, Sun Microsystems lanzó Java 1.0, con la promesa de "Escribir una vez, ejecutar en cualquier lugar" (Write Once, Run Anywhere - WORA), gracias a la Java Virtual Machine (JVM), que permitía ejecutar programas en cualquier sistema operativo sin necesidad de compilarlos nuevamente.

Durante la segunda mitad de los años 90, Java se convirtió en uno de los lenguajes más utilizados en la web debido a la popularidad de los **applets de Java** para navegadores. En esta época, se lanzaron diversas versiones: 1.1, 2 (Introdujo swing, primera interfaz grafica) y 5 (introdujo genericos y autoboxing).

En 2009, Oracle Corporation adquirió Sun Microsystems, lo que generó cambios importantes en Java, entre ellas, se aceleró el desarrollo del lenguaje, se implementaron versiones más frecuentes con soporte a largo plazo (LTS), se introdujeron mejoras en el rendimiento y la seguridad, las versiones más relevantes incluyen:

Java 8 (2014): Expresiones lambda, API de Streams y mejoras en la concurrencia.

Java 9 (2017): Sistema de módulos Jigsaw.

Java 11 (2018): Primera versión LTS bajo Oracle con mejoras en el recolector de basura y HTTP Client.

Java 17 (2021): Segunda versión LTS tras Java 11, con mejor rendimiento y seguridad.

Java 21 (2023): Última versión LTS, con características como patrones de coincidencia y mejoras en la concurrencia.

Ejercicio 3:

Ortogonalidad: Un conjunto pequeño de conceptos combinables de muchas formas sin interacciones inesperadas.

Ejemplo: Scheme, un lenguaje funcional que permite combinar estructuras simples de manera poderosa.

Expresividad: Capacidad de escribir código de manera concisa y clara, minimizando la cantidad de código repetitivo.

Ejemplo: Python, que permite escribir código legible y directo con menos líneas en comparación con otros lenguajes.

Legibilidad: Facilidad para que otros programadores comprendan el código sin esfuerzo adicional.

Ejemplo: Python, gracias a su sintaxis clara y cercana al lenguaje natural.

Simplicidad: Un diseño que evita características innecesarias o redundantes, reduciendo la complejidad.

Ejemplo: Go, con una sintaxis minimalista y enfoque en la facilidad de uso.

Eficiencia: Capacidad de ejecutar programas con buen rendimiento en términos de tiempo y uso de recursos.

Ejemplo: C, que permite optimizar el uso de memoria y CPU.

Portabilidad: La capacidad de ejecutarse en diferentes sistemas sin cambios significativos en el código.

Ejemplo: Java, gracias a la Java Virtual Machine (JVM) que permite ejecutar programas en cualquier plataforma.

Modularidad: Soporte para dividir el código en módulos reutilizables, mejorando la mantenibilidad.

Ejemplo: Ada, diseñado con un fuerte soporte para módulos y encapsulación.

Seguridad: Prevención de errores y vulnerabilidades mediante mecanismos como tipado fuerte y manejo seguro de memoria.

Ejemplo: Rust, que evita errores de acceso a memoria y proporciona seguridad en concurrencia.

Facilidad de Depuración y Mantenimiento: Herramientas que ayuden a detectar errores y mejorar el código a lo largo del tiempo.

**Ejemplo**: **Java**, con su sistema de excepciones bien definido y herramientas como el depurador de Eclipse.

Entre todos estos lenguajes uno que cumple con muchas de las anteriores caracteristicas es Python. El cual es legible y simple gracias a su sintaxis clara, posee expresividad permitiendo realizar tareas complejas con poco codigo, portabilidad al ser interpretado en multiples plataformas y modularidad, gracias al amplio ecosistema de bibliotecas.

ADA

Ejercicio 5: Ada es un lenguaje fuertemente tipado, lo que evita errores en el manejo de datos y mejora la seguridad del codigo. Admite tipos escalares (enteros,char,boolean), datos compuestos (arreglos), derivados (permite crear tipos basados en otros) y subtipos (para restringir valores dentro de un rango especifico).

Soporta el encapsulamiento mediante paquetes, que permiten definir tipos abstractos de datos con alta modularidad.

Ofrece estructuras de datos robustas y seguras como arreglosm registros, listas y colas y punteros.

Posee un sistema avanzado de manejo de excepciones que permite capturar y manejar errores en tiempo de ejecucion.

Tiene soporte nativo para la concurrencia a traves de tareas, lo que facilita la programacion en paralelo.

JAVA

Ejercicio 6: Java fue desarrollado por Sun Microsystems en 1991 bajo el nombre Oak, con el objetivo de programar dispositivos electrónicos y sistemas embebidos. Sin embargo, con la expansión de Internet, se adaptó como un lenguaje de propósito general y se lanzó oficialmente en 1995. Java revoluciono el desarrollo web en los 90 al introducir applets de JAVA, Java Servlet y JSP y JEE (Plataforma para el desarrollo de aplicaciones empresariales escalables). No, Java es independiente de la plataforma, gracias al concepto "Write Once, Run Anywhere" (WORA). Esto se debe a que: Java se compila en bytecode (no en código máquina específico). La JVM (Java Virtual Machine) interpreta el bytecode y lo ejecuta en cualquier sistema operativo. Es por ello que existen implementaciones de JVM para Windows, macOS, Linux y dispositivos móviles.

Ejercicio 7: Java se inspiró en varios lenguajes anteriores, incluyendo:

C: Heredó su sintaxis básica y estructuras de control.

C++: Adoptó la Programación Orientada a Objetos (POO) y algunas características de modularidad.

Smalltalk: Influenció su modelo de POO con encapsulación y manejo de memoria automático.

Objective-C: Contribuyó con conceptos de mensajes entre objetos.

Java simplificó aspectos de C y C++ al eliminar la manipulación directa de memoria (pointers) y mejorar la gestión de errores.

Ejercicio 8: Los applets eran pequeños programas en Java que se ejecutaban dentro de un navegador web. Estos se usaban para añadir funcionalidades interactivas en páginas web, se ejecutaban en una JVM dentro del navegador. Pero eran inseguros y requerían permisos, lo que llevó a su desuso y posteriormente fueron reemplazados por tecnologías como JavaScript, HTML5 y WebAssembly.

Los servlets son programas en Java que se ejecutan en el servidor y generan contenido dinámico en páginas web. Estos reemplazaron a los CGI (Common Gateway Interface) en servidores web. Son parte de Java EE y se ejecutan en un contenedor de servlets (como Tomcat) y pueden procesar solicitudes HTTP y generar respuestas dinámicas.

C

Ejercicio 9: Un programa en C sigue una estructura básica compuesta por: Directivas de preprocesador (#include, #define, etc.). Declaraciones globales (variables y funciones). Función main(), que es el punto de entrada del programa y otras funciones definidas por el usuario. C no permite la anidación de funciones dentro de otras funciones. Las funciones deben declararse de manera separada, fuera de main() o cualquier otra función. Pero sí es posible anidar llamadas a funciones, pero no su declaración dentro de otra función.

Ejercicio 10: En C, una **expresión** es cualquier combinación de operadores y operandos que se evalúan para producir un resultado. Tipos de expresiones:

1. Aritmeticas (+, -, /, \*)

2. Relacionales (true o false)

3. Logicas (and, or, not)

4. Bit a Bit (operan directamente sobre los bits de un numero)

5. Condicionales (if, se utiliza el operador ternario ? : )

6. De asignacion (=, +=. -=, etc.)

Python – Ruby – PHP

Ejercicio 11:

Python

¿Qué tipo de programas se pueden escribir con Python?

Desarrollo web (Django, Flask), Ciencia de datos e IA (NumPy, Pandas, TensorFlow), Automatización y scripting, Aplicaciones de escritorio y videojuegos (PyQt, Pygame), Ciberseguridad y análisis forense.

¿A qué paradigma responde?

Multiparadigma: Soporta Programación Orientada a Objetos (POO), funcional e imperativa.

Características que determinan su paradigma:

Clases y objetos (POO). Funciones de primera clase y programación funcional (map, filter, lambda). Código legible y estructurado (imperativo).

Ruby

¿Qué tipo de programas se pueden escribir con Ruby?

Aplicaciones web (Ruby on Rails). Automatización y scripting. Desarrollo de juegos.

¿A qué paradigma responde?

Principalmente Orientado a Objetos, aunque soporta programación funcional e imperativa.

Características que determinan su paradigma:

Todo es un objeto (incluso números y funciones). Métodos y bloques permiten estilos funcionales.

PHP

¿Qué tipo de programas se pueden escribir con PHP?

Aplicaciones web dinámicas. Sistemas de gestión de contenidos (WordPress, Drupal). API backend.

¿A qué paradigma responde?

Multiparadigma: Soporta POO e imperativa.

Características que determinan su paradigma:

POO: Clases y objetos. Código procedural para scripts.

Ejercicio 12:

Python: Tipado dinámico y fuerte, sintaxis clara, modularidad con módulos y paquetes, alto nivel de abstracción.

Ruby: Todo es un objeto, sintaxis intuitiva, tipado dinámico, enfoque en la simplicidad y productividad.

PHP: Lenguaje orientado a la web, incrustable en HTML, tipado débil, ejecución en servidor.

Gobstone: Diseñado para enseñar programación, lenguaje gráfico basado en tableros y movimientos, imperativo.

Processing: Orientado a gráficos e interacción visual, sintaxis similar a Java, usado en arte digital y visualización.

Javascript

Ejercicio 13:

JavaScript es un lenguaje multiparadigma, lo que significa que admite diferentes estilos de programación:

Programación Orientada a Objetos (POO): Usa objetos y prototipos en lugar de clases tradicionales.

Programación Funcional: Permite funciones de orden superior, inmutabilidad y programación basada en funciones.

Programación Imperativa: Permite escribir código con estructuras de control tradicionales (bucles, condicionales).

Programación Basada en Eventos: Es clave en el desarrollo web, reaccionando a eventos del usuario (clicks, teclados, etc.).

JavaScript pertenece a:

Lenguaje de alto nivel: Su sintaxis ya que es fácil de entender y no requiere gestión manual de memoria.

Lenguaje interpretado: Ya que originalmente se ejecutaba en el navegador sin necesidad de compilación.

Lenguaje dinámico: Debio a que permite cambios en tiempo de ejecución, como la asignación de tipos de datos flexibles.

Lenguaje de scripting: Ya que fue creado inicialmente para añadir interactividad a páginas web, aunque ahora también en servidores (Node.js).

Ejercicio 14:

Tipado dinámico: No es necesario definir el tipo de una variable, ya que se asigna en tiempo de ejecución.

Declaración de variables:

var: Tiene ámbito de función y permite redeclaración.

let: Tiene ámbito de bloque y evita redeclaración.

const: Define constantes y no permite reasignación.

JavaScript usa try...catch para capturar y manejar errores.

JavaScript maneja asincronía con callbacks, promesas y async/await. Esto es util para peticiones HTTP y operaciones no bloqueantes.