ESTRUCTURA DE DATOS FUNDAMENTALES Y ALGORITMOS
GUÍA DE LABORATORIO 01 – 02
Comparación de los lenguajes de programación JAVA, PYTHON y C

Asignatura	Datos del alumno	Fecha y Firma
Algoritmos y solución de problemas	Apellidos:	
	Nombre:	

Instrucciones:

Desarrollar las actividades que indica el docente en base a la guía de trabajo que se presenta.

1. Objetivos:

- Definir grupos de trabajo de laboratorio (conformado por 2 personas)
- Comparativa de los lenguajes de programación JAVA, PYTHON Y C
- Escribir algoritmos y codificar haciendo uso de recursividad.

2. Equipos, herramientas o materiales

- Computador, Software: Python, Algoritmos

3. Fundamento teórico

3.1. Conceptos Clave

3.1.1. JAVA

Tipado: Fuertemente tipado y estático. Las variables deben ser declaradas con un tipo.

Paradigma: Principalmente orientado a objetos, lo que facilita la creación de estructuras de datos complejas.

Gestión de Memoria: Automática, a través del recolector de basura.

Rendimiento: Más lento que C, pero más rápido que Python en general. Máquina virtual Java (JVM) permite portabilidad entre plataformas.

Uso en Estructuras de Datos y Algoritmos: Buena elección para aprender conceptos OOP junto con estructuras de datos y algoritmos. Ampliamente utilizado en aplicaciones empresariales.

3.1.2. PYTHON

Tipado: Dinámico y con tipado fuerte. No es necesario declarar el tipo de una variable antes de usarla.

Paradigma: Multi-paradigma, incluyendo programación funcional y orientada a objetos.

Gestión de Memoria: Automática, con un recolector de basura.

Rendimiento: Generalmente más lento debido a ser un lenguaje interpretado. Pero su sintaxis concisa lo hace popular para la enseñanza y el desarrollo rápido.

Uso en Estructuras de Datos y Algoritmos: Ideal para la enseñanza debido a su sintaxis clara y legible. Ampliamente utilizado en ciencia de datos, aprendizaje automático y desarrollo web.

3.1.3. C

Tipado: Fuertemente tipado y estático. Requiere declaración explícita de tipos de variables.

Paradigma: Imperativo. No soporta programación orientada a objetos de forma nativa.

Gestión de Memoria: Manual, dando al programador control total sobre la asignación y liberación de memoria.

Rendimiento: Muy eficiente y rápido, ideal para sistemas de bajo nivel.

Uso en Estructuras de Datos y Algoritmos: Excelente para entender cómo funcionan las estructuras de datos a nivel de memoria. Ampliamente usado en sistemas embebidos, sistemas operativos y aplicaciones donde el rendimiento es crítico.

3.1.4. Comparación en el Contexto de Estructuras de Datos y Algoritmos

Java y Python facilitan más la implementación de algoritmos y estructuras de datos debido a su gestión automática de memoria y tipado de alto nivel. Son ideales para aprender conceptos sin preocuparse demasiado por los detalles de bajo nivel.

C, por otro lado, es excelente para entender cómo funcionan las estructuras de datos y algoritmos a nivel de hardware. El manejo manual de memoria y el tipado estático proporcionan una comprensión profunda de la eficiencia y la optimización.

3.1.5. Comparación a través de los Ejercicios

Orientación a Objetos: Java y Python utilizan clases y objetos, lo que facilita la implementación de estructuras de datos complejas. C, en cambio, utiliza estructuras y apuntadores, lo que requiere un entendimiento más profundo de la gestión de memoria.

Sintaxis y Legibilidad: Python destaca por su simplicidad y legibilidad, mientras que Java y C requieren más líneas de código y una sintaxis más estricta.

Gestión de Memoria: En C, debes gestionar la memoria manualmente, lo que implica un mayor control pero también más responsabilidad. Java y Python manejan la memoria automáticamente.

Tipado: Java y C requieren la declaración explícita de tipos, mientras que Python infiere los tipos, permitiendo un código más flexible y conciso.

3.1.6. Ejemplo de Ejercicio: Suma de Números Naturales

Java

En Java, podríamos escribir un programa simple para sumar los primeros n números naturales usando un bucle for.

```java
public class SumNumbers {
    public static void main(String[] args) {
        int n = 10; // Cambia esto para probar con diferentes números
        int sum = 0;

        for (int i = 1; i <= n; i++) {
            sum += i;
        }

        System.out.println("La suma de los primeros " + n + " números naturales es: " + sum);
    }
}
```

Python

En Python, el mismo ejercicio se puede hacer con menos líneas de código gracias a su sintaxis concisa.

```python
n = 10  # Cambia esto para probar con diferentes números
sum = sum(range(1, n+1))

print(f"La suma de los primeros {n} números naturales es: {sum}")
```

C

En C, necesitamos incluir bibliotecas para la entrada/salida y manejar la inicialización de variables y bucles de una manera más explícita.

```c
#include <stdio.h>

int main() {
    int n = 10; // Cambia esto para probar con diferentes números
    int sum = 0;

    for (int i = 1; i <= n; i++) {
        sum += i;
    }

    printf("La suma de los primeros %d números naturales es: %d\n", n, sum);
    return 0;
}
```

3.1.7. Conclusión

Para Principiantes: Python es a menudo recomendado debido a su sintaxis simple y clara.

Para Profundizar en OOP: Java es una buena opción, especialmente en entornos empresariales.

Para Entender el Funcionamiento Interno y el Rendimiento: C es inigualable.

4. Desarrollo y Actividades

- **Ejercicio parte 01:**

Operaciones Básicas:

1) Realiza la suma, resta, multiplicación y división de dos números ingresados por el usuario.

Verificación de Número Par o Impar:

2) Solicita un número al usuario y determina si es par o impar.

Área de un Triángulo:

3) Pide la base y la altura de un triángulo al usuario y calcula su área.

Calculadora de Factorial:

4) Crea una función que calcule la factorial de un número.

Número Primo:

5) Verifica si un número ingresado por el usuario es primo o no.

Inversión de Cadena:

6) Toma una cadena de texto y muestra su inversión.

Suma de Números Pares:

7) Calcula la suma de los números pares en un rango especificado por el usuario.

Lista de Cuadrados:

8) Crea una lista de los cuadrados de los primeros 10 números naturales.

Contador de Vocales:

9) Cuenta el número de vocales en una cadena de texto.

Números de la Serie Fibonacci:

10) Genera los primeros 10 números de la serie Fibonacci.

Ordenamiento de Lista:

11) Ordena una lista de números ingresados por el usuario de menor a mayor.

Palíndromo:

12) Verifica si una palabra ingresada por el usuario es un palíndromo.

Generador de Tablas de Multiplicar:

13) Crea un programa que genere la tabla de multiplicar de un número ingresado por el usuario.

Cálculo del Área de un Círculo:

14) Pide el radio de un círculo al usuario y calcula su área.

Suma de Dígitos:

15) Toma un número entero y calcula la suma de sus dígitos.