
	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación

Universidad Politécnica Salesiana

Vicerrectorado Docente

Código del Formato:	GUIA-PRL-001
Versión:	VF1.0
Elaborado por:	Directores de Área del Conocimiento Integrantes Consejo Académico
Fecha de elaboración:	2016/04/01
Revisado por:	Consejo Académico
Fecha de revisión:	2016/04/06
Aprobado por:	Lauro Fernando Pesántez Avilés Vicerrector Docente
Fecha de aprobación:	2016/14/06
Nivel de confidencialidad:	Interno

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Descripción General

Propósito


El propósito del presente documento es definir un estándar para elaborar documentación de guías de práctica de laboratorio, talleres o centros de simulación de las Carreras de la Universidad Politécnica Salesiana, con la finalidad de lograr una homogenización en la presentación de la información por parte del personal académico y técnico docente.


Alcance

El presente estándar será aplicado a toda la documentación referente a informes de prácticas de laboratorio, talleres o centros de simulación de las Carreras de la Universidad Politécnica Salesiana.

Formatos


- Formato de Guía de Práctica de Laboratorio / Talleres / Centros de Simulación – para Docentes
- Formato de Informe de Práctica de Laboratorio / Talleres / Centros de Simulación – para Estudiantes


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		


		FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES	
CARRERA: COMPUTACIÓN		ASIGNATURA: Programación Aplicada	
NRO. PRÁCTICA:	1	TÍTULO PRÁCTICA: Investigación de las versiones de Java	
OBJETIVO: Identificar los cambios importantes de Java Diseñar e Implementar las nuevas técnicas de programación Entender la cada uno de las características nuevas en Java			
INSTRUCCIONES (Detallar las instrucciones que se dará al estudiante):		1. Revisar los conceptos fundamentales de las nuevas versiones de Java	
		2. Establecer las características de Java	
		3. Implementar y diseñar los nuevos componentes de programación	
		4. Realizar el informe respectivo según los datos solicitados.	
<p align="center">ACTIVIDADES POR DESARROLLAR</p> <p align="center">(Anotar las actividades que deberá seguir el estudiante para el cumplimiento de la práctica)</p>			
1. Revisar la teoría y conceptos de Java 9 ,10, 11, 12			
2. Diseñar e implementar las características de Java,			
3. Probar su funcionamiento y rendimiento dentro de los equipos de computo			
4. Realizar práctica codificando los códigos de las nuevas características de Java.			
RESULTADO(S) OBTENIDO(S): Realizar procesos de investigación sobre los cambios importantes de Java Entender las aplicaciones de codificación de las nuevas características Entender las funcionalidades adicionales de Java.			
CONCLUSIONES: Aprenden a trabajar en grupo dentro de plazos de tiempo establecidos, manejando el lenguaje de programación de Java.			
RECOMENDACIONES: Realizar el trabajo dentro del tiempo establecido.			

Docente / Técnico Docente: _____

Firma: _____

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

		FORMATO DE INFORME DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA ESTUDIANTES	
CARRERA: Ingeniería en Computación		ASIGNATURA: Programación Aplicada	
NRO. PRÁCTICA:	1	TÍTULO PRÁCTICA: JAVA 12	
OBJETIVO ALCANZADO: Identificar los cambios importantes de Java Diseñar e Implementar las nuevas técnicas de programación Entender la cada uno de las características nuevas en Java			
ACTIVIDADES DESARROLLADAS			
<p>1. Revisar la teoría y conceptos de Java 12:</p> <p>JDK 12 o coloquialmente más conocido como Java 12. No es una versión LTS como Java 8 y Java 11. La versión 12 de Java tiene las siguientes actualizaciones:</p> <ul style="list-style-type: none"> • Expresiones Switch: Extiende a la sentencia Switch para que pueda ser utilizada tanto como una sentencia como una expresión. Las dos formas pueden tener un comportamiento de alcance y flujo de control tradicional o simplificado. <p>Estos cambios simplifican la escritura del código, además de preparar el camino para el uso de coincidencia de patrones en <<switch>>.</p> <ul style="list-style-type: none"> • Colecciones abortables listas para G1: Hace que las colecciones abortables mixtas de G1 se puedan cancelar si superan el objetivo de la pausa. • Devolución rápida de la memoria comprometida no utilizada en G1: Mejora el recolector de basura G1 para que pueda devolver automáticamente un conjunto de memoria Java al sistema operativo cuando está inactivo. • Recolector de basura de bajo tiempo de pausa Shenandoah: Añade un nuevo algoritmo de recolector de basura que reduce los tiempos de pausa del propio recolector mediante un trabajo de evacuación concurrente con los hilos de Java en ejecución. Por ahora se encuentra en fase experimental. • Microbenchmark suite: Una suite de microbenchmark es añadida al código fuente del JDK y su intención es hacer para los desarrolladores más fácil ejecutar microbenchmark existentes y ejecutar unos nuevos. • API de constantes en la JVM: Se introdujo una API para modelar descripciones nominales de archivos de clase clave y artefactos de tiempos de ejecución, en particular las constantes que se pueden cargar desde la agrupación constante. • Un único port a AArch64: Elimina todo tipo de fuentes relacionadas con el port ARM64 mientras se mantiene los ports de ARM 32-bit y aarch64-64bit, para que los desarrolladores se centren en una única forma de implementación de ARM64-bit. 			

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

- **Archivos CDS por defecto:** Mejora el proceso de compilación del JDK a la hora de generar un archivo CDS usando la lista de clases predeterminada en plataformas de 64-bit.

2. Establecer las características de Java:

- Simple: Ofrece funcionalidad potente y sin características menos usadas y más confusas. Reduce un 50% los errores más comunes de programación (C+, C++) como: Aritmética de punteros, no existencia de referencias, registros(struct), definición de tipos(typedef), macros(#define), liberación de memoria(free).
- Orientada a objetos: Java trabaja los datos como objetos e implementa interfaces a esos objetos.
- Distribuido: Java tiene extensas capacidades de interconexión TCP/IP. Tiene librerías para acceder e interactuar con protocolos como http y ftp, así hay un mejor acceso a la información a través de la red.
- Robusto: Java va en busca de problemas en tiempo de compilación y en tiempo de ejecución; La comprobación ayuda a detectar errores, comprueba los punteros, comprueba los límites del array, excepciones, verificación de byte-codes.
- Arquitectura neutral: El compilador Java compila su código a un fichero objeto con formato independiente de la arquitectura de la máquina en el que se ejecuta.
- Seguro: Características como punteros o el casting hacen que los compiladores de C y C++ se eliminan para prevenir el acceso ilegal a la memoria.
- Portable: Implementación de estándares de portabilidad para facilitar el desarrollo.
- Interpretado: El intérprete Java (sistema run-time) puede ejecutar directamente el código. No existen compiladores específicos de Java para las plataformas; Java es más lento que otros lenguajes de programación, como C++, ya que es interpretado y no ejecutado como en otro programa tradicional.
- Multithreaded: Java permite muchas actividades a la vez en un programa. Los Threads son pequeños procesos o piezas independientes de un gran proceso. Su beneficio es el mejor rendimiento interactivo y mejor comportamiento en tiempo real.

- Dinámico: Java se beneficia de la tecnología orientada a objetos, las librerías nuevas o actualizadas no paralizan las aplicaciones actuales.
Java simplifica el uso de protocolos nuevos o actualizados.
Para evitar que los módulos de byte-codes, objetos, nuevas clases se traigan de la red cada vez que sea necesario implementa opciones de persistencia para que no se eliminen cuando se limpie la caché de la máquina.

3. Probar su funcionamiento y rendimiento dentro de los equipos de cómputo:

Switch:

Permiten quitar varias sentencias “*If else*” encadenadas. Cada rama de la sentencia switch devuelve un valor, no hace falta utilizar la sentencia “*break*” si no que se pueden utilizar varios casos para cada rama.

```
Scanner leer= new Scanner(System.in);  
System.out.println("Ingrese el día:");  
String dia=leer.nextLine();
```

```
switch (dia) {  
  
    case "Lunes":  
    case "Martes":  
    case "Miercoles":  
        System.out.println(1);  
        break;  
    case "Jueves":  
        System.out.println(4);  
        break;  
    case "Viernes":  
        System.out.println(5);  
    case "Sabado":  
        System.out.println(5);  
        break;  
    case "Domingo":  
        System.out.println(7);  
        break;  
}
```

Sin break:

```
run:  
Ingrese el día:  
Lunes  
1
```

Con Break:

```
run:
Ingrese el día:
Jueves
4
BUILD SUCCESSFUL (total time: 2 seconds)
|
```

- *Declaración de tipo de dato:*

CASO 1:


```
Scanner leer= new Scanner(System.in);
System.out.println("Ingrese el día:");
String dia=leer.nextLine();
System.out.println("DÍA:"+ dia);
```

```
run:
Ingrese el día:
Lunes
DÍA:Lunes
BUILD SUCCESSFUL (total time: 3 seconds)
|
```

CASO 2:

```
Scanner leer= new Scanner(System.in);
System.out.println("Ingrese el día:");
var day=leer.nextLine();
System.out.println("DÍA:"+ day);
```

```
run:
Ingrese el día:
Lunes
DÍA:Lunes
BUILD SUCCESSFUL (total time: 3 seconds)
|
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

4. Bibliografía:

<https://www.muylinux.com/2019/03/22/oracle-jdk-12-java-12/>

<https://internetpasoapaso.com/versiones-java/>

<https://picodotdev.github.io/blog-bitix/2019/03/novedades-de-java-12/>

<http://www.itlp.edu.mx/web/java/Tutorial%20de%20Java/Intro/carac.html>

RESULTADO(S) OBTENIDO(S):

- Entendimiento del funcionamiento JDK 12 comparado con versiones anteriores.
- Java 12 tiene mayores facilidades en algunos ámbitos que ayudan a programar.
- Aplicar investigación sobre las versiones de Java.

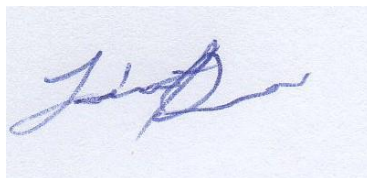
CONCLUSIONES:

Para el mejor entendimiento de la programación genérica se necesita investigación sobre el ambiente en que se trabaja, se necesita investigar y actualizarse sobre el JDK en el que se trabaja ya que gracias a la información adquirida podemos realizar una mejor programación.

RECOMENDACIONES:

Por el momento no tengo ninguna recomendación que dar, el docente fue claro al dar la tarea.

Nombre de estudiante: Lisseth Reinoso



Firma de estudiante: