Clase grabada

Recibe una cálida:

¡Bienvenida!

Te estábamos esperando 😁

English Always | alkemy

Clase grabada

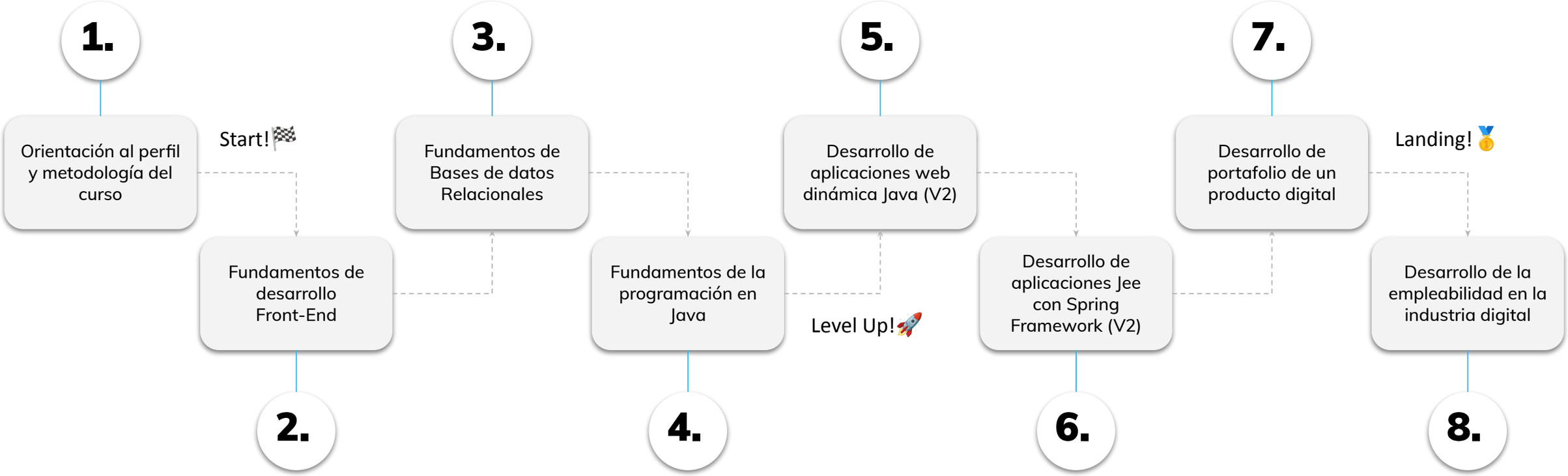
Algoritmos

Plan formativo: Desarrollo de Aplicaciones Full Stack Java Trainee V2.0

English Always | alkemy

HOJA DE RUTA

¿Cuáles **skill** conforman el programa?



English Always | alkemy

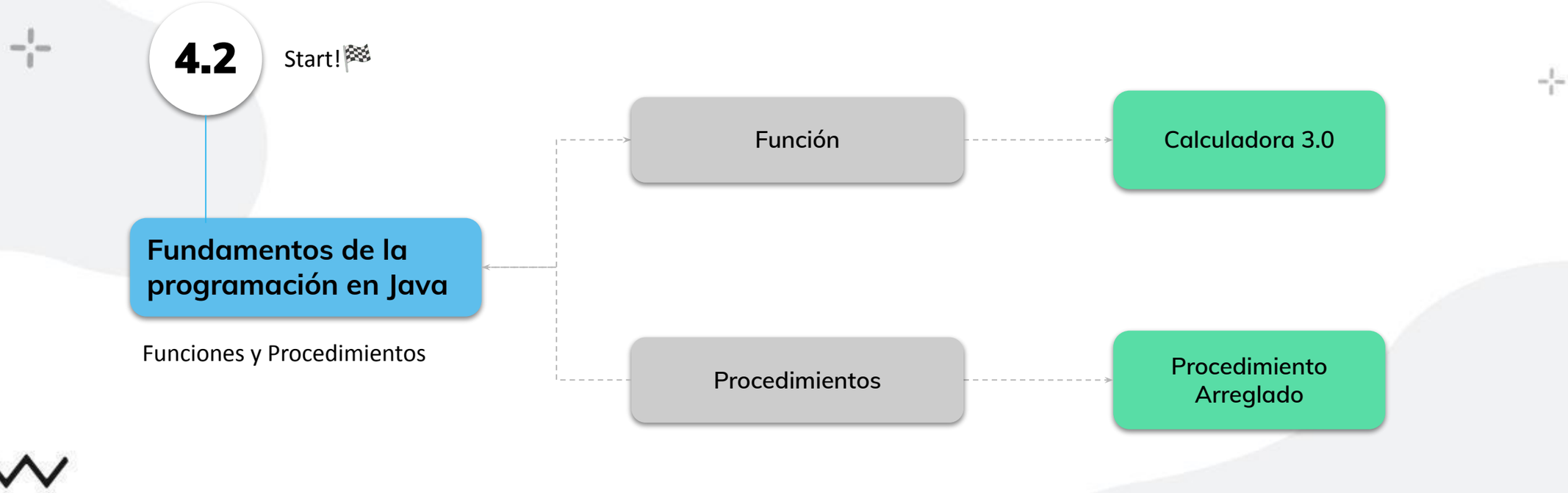
REPASO CLASE ANTERIOR

En la clase anterior trabajamos 📚:

- ✓ *Diagramas de Flujo*
- ✓ *Arreglos*

English Always | alkemy

LEARNING PATHWAY



English Always | alkemy

OBJETIVOS DE APRENDIZAJE

¿Qué aprenderemos?

- *Reconocer la importancia de los subprogramas*
- *Comprender el uso de funciones*
- *Aprender la implementación de procedimientos*

Rompehielo🥶

Chat descompuesto: 🙌

El moderador enviará la siguiente frase al chat: “Llenar una taza con agua caliente”.

A continuación, deben enviar mensajes en el chat que completen los pasos faltantes en la frase según crean conveniente.

Consigna: ✍️

Cada participante deberá reescribir el mensaje basándose en el último mensaje recibido, pero enfocándose en una instrucción lógica o paso en un algoritmo.

Analicemos entre todos: ⚙️

¿Cómo cada mensaje fue transformado en pasos específicos?

¿Cómo podríamos agrupar estos pasos en bloques de código que puedan ser reutilizados en diferentes situaciones?

Funciones

Funciones

¿Qué es una función?

Una función en programación es un **bloque de código reutilizable** que realiza una tarea específica. Funciona como una pequeña unidad de trabajo que acepta entradas (llamadas parámetros) y puede producir salidas (un valor de retorno). Las funciones permiten dividir un programa en partes más pequeñas y manejables, lo que facilita la comprensión, el mantenimiento y la reutilización del código.

Una función es un subprograma que tiene 3 componentes importantes:

- Los **parámetros** son los valores que recibe la función como entrada;
- El **bloque de código** son las operaciones que hace la función;
- El resultado (o **valor de retorno**) es el valor final que entrega la función.

Funciones

¿Cómo se declara una función?:

Al definir una función, establecemos su firma, que incluye el **tipo de datos que retorna** la función y los **tipos y nombres de los parámetros** que acepta. Los parámetros son variables locales que **reciben valores cuando se llama a la función**, y estos valores se utilizan dentro del bloque de código para realizar operaciones.

```
nombre_funcion (argumento1 , argumento2)

sumar (num1, num2)
```

Es posible tener funciones con 0 o más argumentos y pueden estar definidas por el propio usuario o, en muchas ocasiones, ser funciones definidas por el propio compilador.

Funciones

¿Cómo se invoca una función?:

Una función definida por el usuario se invoca haciendo referencia a su nombre. En pseudocódigo:

...
nombreFuncion (lista de parámetros reales o actuales)
...

La sentencia **nombreFuncion** acompañada de los **parámetros** es la que inicia la ejecución de la función. El control de ejecución lo toma la función, ejecuta secuencialmente cada una de sus sentencias, y cuando termina de ejecutarse, le devuelve el control al programa llamador

LIVE CODING

Ejemplo en vivo

Creando una función en pseudocódigo:

Vamos a crear un algoritmo que a través de una función convierta una cantidad de euros introducida por teclado a otra moneda, estas pueden ser a dólares, yenes o libras. La función tendrá como parámetros, la cantidad de euros y la moneda a convertir (cadena).

El cambio de divisas es:

* *0.86 libras es un 1 €*
* *1.28611 $ es un 1 €*
* *129.852 yenes es un 1 €*

LIVE CODING

Ejemplo en vivo

Creando una función: Paso a paso

1. *Primero debes crear el algoritmo y pedirle al usuario el monto en euros*
2. *Luego llamar a la función.*
3. *La función debe codificarse teniendo en cuenta que su parámetro será (euro)*
4. *Devolver el resultado de todas las conversiones.*

Tiempo: 20 minutos

Ejercicio N° 1

Calculadora 3.0

Calculadora 3.0

Contexto: 🙌

Vamos a mejorar la calculadora que hemos estado creando en diferentes ejercicios. En este caso, vamos a reutilizar código para que el cuerpo del algoritmo quede más simple. Para esto, haremos uso de funciones.

Consigna: ✍️

Crea un algoritmo que le pida dos números al usuario y este pueda elegir entre sumar, restar, multiplicar y dividir. La aplicación debe tener una función para cada operación matemática y deben devolver sus resultados.

Tiempo🕐: *20 minutos*

Paso a paso: ⚙️

- Definir una función para sumar que reciba dos números como parámetros y retorne la suma de ambos. Realizar lo mismo con el resto de las operaciones.
- Pedir al usuario que ingrese dos números y almacenarlos en variables num1 y num2
- Mostrar un menú con las opciones: sumar, restar, multiplicar y dividir
- Leer la opción elegida por el usuario y almacenarla en una variable opción
- Utilizar un condicional Según para ejecutar la función según la opción elegida.
- En cada caso del Según, llamar a la función correspondiente para imprimir su valor.

Procedimientos

Procedimientos

¿Qué es un procedimiento?:

Un procedimiento es un bloque de código que realiza una tarea específica, **pero no retorna ningún valor**. Es por esto que los procedimientos no especifican tipo de dato de retorno. También deben ser declarados obligatoriamente antes de que puedan ser llamados en el cuerpo del programa principal.

Todo procedimiento, al igual que un programa principal, consta de una cabecera, que proporciona su nombre y sus parámetros de comunicación; de una sección de declaraciones locales y el cuerpo de sentencias ejecutables.

Procedimientos

Las principales diferencias entre procedimientos y funciones son:

- Un procedimiento es un bloque de código que realiza una tarea específica, pero no retorna ningún valor.Una función también realiza una tarea, pero debe retornar un valor.
- Las funciones deben declararse indicando el tipo de dato del return, en cambio los procedimientos no especifican tipo de retorno.
- Las funciones se pueden llamar desde dentro de expresiones, por ejemplo para pasarlas como parámetro a otra función. Los procedimientos no se pueden usar de esta forma.

En general, si una tarea requiere devolver un valor, es mejor usar una función. Si solo se necesita ejecutar un bloque de código, un procedimiento es suficiente.

Procedimientos

¿Cómo de declara un procedimiento?

En pseudocódigo, la declaración de un procedimiento tiene la siguiente estructura:

PROCEDIMIENTO nombreProcedimiento(parámetros)

    Instrucciones

FINPROCEDIMIENTO

- **nombreProcedimiento**: es el identificador único con el que se llamará al procedimiento.
- **parámetros** (opcional): lista de parámetros o variables entre paréntesis que recibe el procedimiento.
- **Instrucciones**: son las acciones que realiza el procedimiento, pueden ser asignaciones, operaciones, lectura de datos, etc.

Paso de parámetros

Parámetros

Paso de parámetros a subprogramas

Los métodos más empleados para realizar el paso de parámetros son:

o **Paso por valor** (parámetro valor)
o **Paso por referencia o dirección** (parámetro variable)

Los parámetros formales (locales al subprograma) reciben como valores iniciales los valores de los parámetros reales, y con ellos se ejecutan las acciones descritas en el subprograma.

Parámetros

Paso por valor:

Son los parámetros **unidireccionales**, que se usan para dar información al subprograma, pero no pueden devolver valores desde el mismo. **Aunque el procedimiento les cambie su valor, este nuevo valor no se refleja en el programa** llamador. Esto se debe a que en la llamada al subprograma, se le asigna el valor del parámetro real a la variable que representa al parámetro formal correspondiente, dicho de otra forma, **se crea una copia del parámetro real.**

Paso por referencia:

*Son los parámetros que **sólo reciben valor en el subprograma**, o bien proporcionan valor al subprograma y reciben un valor nuevo en el mismo. Así, **todo cambio realizado sobre los parámetros formales se refleja en los parámetros reales correspondientes**. Se los considera como parámetros **bidireccionales** o variables, ya que son de Entrada y/o Salida.*

Parámetros

En funciones:

Una función puede tener parámetros variables además de parámetros valor en la lista de parámetros formales. Una función puede cambiar el contenido de una variable global y ejecutar instrucciones de entrada/salida. Estas operaciones se conocen como parámetros laterales, y se deben evitar.

En procedimientos

Si un procedimiento modifica una variable global (distinta de un parámetro real), éste es un efecto lateral. Por ello, excepto en contadas ocasiones, no debe aparecer en la declaración del procedimiento. Si se necesita una variable temporal en un procedimiento, se debe utilizar una variable local, no una global. Si se desea que el procedimiento modifique el valor de una variable global, utilízala como el parámetro real en una llamada al procedimiento.

LIVE CODING

Ejemplo en vivo

Creando un procedimiento en pseudocódigo:

Vamos a crear un procedimiento que permita ingresar los días de la semana por teclado y guardarlos en un arreglo. Para finalizar, imprimimos el arreglo por pantalla.

Tiempo: 20 minutos

LIVE CODING

Ejemplo en vivo

1. *Definir el arreglo "dias[7]"*
2. *Definir el procedimiento y el parámetro: "cargarDias(dias)"*
3. *Dentro del procedimiento, iterar para pedir los días:*
4. *Dentro del ciclo, leer un día con "LEER dias[i]"*
5. *Al finalizar la carga, recorrer el arreglo e imprimirlo:*
6. *Usar otro ciclo para escribir cada día con "ESCRIBIR dias[i]"*
7. *Invocar el procedimiento desde el programa principal.*

Tiempo: 20 minutos

Ejercicio N° 2

Procedimiento Arreglado

Procedimiento Arreglado

Contexto: 🙌

Es hora de realizar un ejercicio de procedimientos y de paso de parámetros. Para esto ¡vamos a manipular arreglos y bucles! Es un buen momento para poner en práctica todo lo aprendido en esta lección.

Consigna: ✍️

Crear un procedimiento que reciba un arreglo por parámetro y muestre por pantalla los valores del arreglo.

Tiempo🕛: *20 minutos*

Procedimiento Arreglado

Paso a paso: ⚙️

1. Definir un arreglo unidimensional y darle valores a sus elementos.
2. Escribir el nombre del procedimiento, por ejemplo "mostrarArreglo"
3. Entre paréntesis definir el parámetro que recibirá, en este caso un arreglo: "mostrarArreglo(arreglo)"
4. Dentro del procedimiento recorrer el arreglo con un ciclo Para
5. Dentro del ciclo, acceder a cada elemento con arreglo[i]
6. Mostrar el elemento actual en cada iteración:
7. Cerrar procedimiento:
8. Llamar al procedimiento: Después de definirlo, hay que llamarlo pasando como parámetro el arreglo. Por ejemplo: "mostrarArreglo(miArreglo)"

¿Alguna consulta?

RESUMEN

¿Qué logramos en esta clase?

- ✓ *Comprender el uso de funciones y procedimientos*
- ✓ *Entender la diferencia entre parámetros por valor y por referencia*
- ✓ *Aplicar la utilización de subprogramas a los algoritmos*

English Always | alkemy

#WorkingTime

Continuemos ejercitando

¡Antes de cerrar la clase! Te invitamos a: 👇👇👇

1. Repasar nuevamente la grabación de esta clase
2. Revisar el material compartido en la plataforma de Moodle (lo que se vio en clase y algún ejercicio adicional)
   a. Material 1 *(Foro)*
   b. *Lectura Módulo 4, Lección 2: páginas 20 - 25*
3. Traer al próximo encuentro, todas tus dudas y consultas para verlas antes de iniciar nuevo tema.

¡Muchas Gracias!

Nos vemos en la próxima clase 👋🏻

English Always | alkemy

Momento:

Time-out!

⏰5 min.