

# UNIDAD 1: Conceptos básicos de programación e IA

## Descripción del Curso

Este curso ofrece una experiencia educativa orientada a introducir conceptos básicos de programación y fundamentos de inteligencia artificial mediante proyectos prácticos y entornos simulados. La propuesta se estructura en unidades que combinan teoría, ejercicios de código y actividades de análisis, con un enfoque aplicado para entender cómo modelar tareas simples de IA y su implementación. En particular, la Unidad 8 propone un Proyecto integrador: diseño de un agente básico en un entorno simulado. El estudiante aplicará lo aprendido para diseñar un agente que perciba un entorno simple, decida acciones y documente su razonamiento, considerando el rendimiento y las limitaciones frente a enfoques de IA más complejos. Este proyecto facilita la conexión entre conceptos de programación (variables, tipos de datos, operadores) y conceptos de IA (agente y entorno), al tiempo que reconoce las limitaciones prácticas y las posibles mejoras futuras. A lo largo del curso se enfatiza el aprendizaje práctico con Python como lenguaje de implementación, el uso de estructuras de control y estructuras de datos simples para modelar percepciones, acciones y reglas de decisión, y la importancia de la documentación técnica y el razonamiento detrás de las decisiones de diseño. Además, se fomenta el desarrollo de habilidades de pensamiento computacional, resolución de problemas, comunicación de resultados y trabajo colaborativo, con una valoración continua de progreso mediante entregas y revisiones iterativas. La modalidad del curso favorece actividades de aprendizaje activo, feedback formativo y reflexión sobre el proceso de diseño de software y sistemas básicos de IA, preparando a los estudiantes para proyectos futuros y para comprender las distancias y limitaciones entre agentes simples y soluciones de IA más avanzadas.

## Competencias

- Aplicar conceptos fundamentales de programación en Python: variables, tipos de datos, operadores, estructuras de control y estructuras de datos simples.
- Comprender y modelar la relación entre un agente y su entorno en tareas de IA simples.
- Diseñar, implementar y probar un agente básico que percibe, decide y actúa en un entorno simulado.
- Documentar de forma clara el razonamiento, las decisiones de diseño y las limitaciones del agente frente a IA más compleja.
- Analizar el rendimiento del agente y proponer mejoras prácticas dentro de un marco de soluciones simples.
- Trabajar de forma colaborativa, comunicar resultados y justificar elecciones técnicas mediante presentaciones y documentación.
- Aplicar pensamiento crítico para comparar enfoques simples y complejos de IA, identificando límites éticos y de seguridad.

## Requerimientos

- Computadora personal con acceso a Internet y capacidades básicas de procesamiento.

- Python 3.x instalado y un editor de código (p. ej., VS Code, PyCharm, o IDLE).
- Conocimientos básicos de programación (recomendado) o haber completado módulos introductorios de programación.
- Entorno de desarrollo para escribir, ejecutar y depurar código Python, así como herramientas para la documentación de proyectos (opcional: Markdown o similar).
- Espacio para entregar código, documentación y reportes de progreso de las entregas del proyecto integrador.

## Unidades del Curso

### Unidad 1: UNIDAD 1: Conceptos básicos de programación e IA

#### Objetivos de Aprendizaje

- Reconocer y definir variables y tipos de datos básicos (numéricos, texto, booleanos) y operadores simples.
- Identificar qué es un agente y qué es un entorno en un problema de IA.
- Ilustrar con un ejemplo sencillo cómo se modela una tarea de IA usando código y percepciones básicas.

#### Contenidos Temáticos

1. **Tema 1: Conceptos básicos de programación y IA** - Descripción: Variables, tipos de datos y operadores; introducción a agente y entorno en IA.
2. **Tema 2: Modelar una tarea de IA con código** - Descripción: Identificación de percepciones, acciones y entorno en un problema simple.
3. **Tema 3: Primeros ejemplos prácticos** - Descripción: Pequeño ejemplo de código que ilustra una tarea de IA de forma simple.

#### Actividades

- **Actividad 1: Exploración de variables y tipos** - Identificar variables y tipos en ejemplos cotidianos, y categorizar operaciones básicas. Puntos clave: comprensión de conceptos y su relación con la programación.
- **Actividad 2: Debate guiado sobre agente y entorno** - En parejas, definir qué es un agente y qué es el entorno con ejemplos de la vida real y de IA. Aprendizajes: diferenciación de conceptos clave.
- **Actividad 3: Modelado en papel** - Representar una tarea pequeña de IA en un esquema de percepciones, acciones y entorno usando lenguaje sencillo o pseudocódigo básico.

#### Evaluación

Evaluación basada en OA1 y OA2: participación en actividades (30%), cuestionario corto de conceptos (30%), y un mini-modelado de IA con percepciones simples (40%).

### Unidad 2: UNIDAD 2: Lógica booleana y estructuras de control

## Objetivos de Aprendizaje

- Explicar cómo la lógica booleana determina decisiones en un programa sencillo.
- Identificar y usar estructuras de control de flujo if/else para tomar decisiones.
- Introducir bucles simples y su uso para repetir acciones cuando sea necesario.

## Contenidos Temáticos

1. **Tema 1: Lógica booleana y condiciones** - Descripción: Operadores lógicos, tablas de verdad y su relación con decisiones de programación.
2. **Tema 2: Estructuras de control if/else** - Descripción: Tomar decisiones entre diferentes caminos del programa.
3. **Tema 3: Bucles básicos** - Descripción: Repetir acciones con while/for para automatizar tareas simples.

## Actividades

- **Actividad 1: Juego de condiciones** - Diseñar reglas simples usando operadores booleanos y convertirlas en código con if/else. Aprendizajes: aplicar lógica booleana en decisiones.
- **Actividad 2: Desafío de bucles** - Crear un bucle que repita una acción hasta cumplir una condición y describir cuándo termina. Aprendizajes: control de repetición.
- **Actividad 3: Diálogo de flujo** - Elaborar un diagrama de flujo simple y su traducción a pseudocódigo para un escenario cotidiano.

## Evaluación

Evaluación basada en OA2 y OA5: ejercicios de condicionales y bucles (60%), participación y reflexión (20%), y revisión de código para detectar errores lógicos (20%).

## Unidad 3: UNIDAD 3: Pseudocódigo y diseño de soluciones

### Objetivos de Aprendizaje

- Explicar qué es el pseudocódigo y su utilidad para planificar programas.
- Escribir pseudocódigo que describa soluciones simples con secuencias, condicionales y bucles.
- Identificar y estructurar los pasos lógicos de problemas cotidianos antes de codificar.

### Contenidos Temáticos

1. **Tema 1: ¿Qué es el pseudocódigo?** - Descripción: Propósito del pseudocódigo y su papel en el diseño de algoritmos.
2. **Tema 2: Estructuras básicas en pseudocódigo** - Descripción: Secuencias, condicionales y bucles descritos de forma descriptiva.

3. **Tema 3: De pseudocódigo a código** - Descripción: Cómo convertir pseudocódigo en pasos de implementación en un lenguaje real.

## Actividades

- **Actividad 1: Escribe un pseudocódigo para un problema cotidiano** - Diseña la solución antes de codificar usando secuencias y decisiones. Aprendizajes: claridad de algoritmo y estructura.
- **Actividad 2: Lectura de pseudocódigo** - Analizar ejemplos y predecir resultados, identificando mejoras posibles.
- **Actividad 3: Transformación a código** - Convertir un pseudocódigo simple a un bloque de código de alto nivel (p. ej., Python) estructurado.

## Evaluación

Evaluación basada en OA3: creación de al menos dos fragmentos de pseudocódigo y su explicación; OA4 y OA5 se evalúan a través de la calidad del diseño y la claridad de la transición a código.

## Unidad 4: UNIDAD 4: Programación básica en Python: decisiones de un agente

### Objetivos de Aprendizaje

- Escribir código básico en Python que use variables y tipos de datos simples.
- Implementar una decisión de un agente usando estructuras de control if/else basada en percepciones.
- Explicar, con palabras propias, cómo el código representa la decisión del agente en su entorno.

### Contenidos Temáticos

1. **Tema 1: Python básico** - Descripción: Sintaxis sencilla, impresión, variables y tipos de datos comunes.
2. **Tema 2: Estructuras de control en Python** - Descripción: if/else, operadores y condiciones básicas.
3. **Tema 3: Decisión de un agente basada en percepciones** - Descripción: Implementación de una decisión en función de una percepción simple del entorno.

## Actividades

- **Actividad 1: Tu primer agente en Python** - Escribe un programa que, dada una percepción simple (p. ej., temperatura), decida una acción (p. ej., imprimir "Activar calefacción" si la temperatura es baja). Aprendizajes: usar variables, condicionales y salida de texto.
- **Actividad 2: Debug básico** - Corrige errores de sintaxis y lógica en un programa de agente. Aprendizajes: identificar errores comunes y corregir su impacto.
- **Actividad 3: Explicación oral** - Explica en voz alta cómo el código representa la decisión del agente y su relación con IA.

## Evaluación

Evaluación basada en OA4 y OA5: 60% por el programa correcto que implemente una decisión simple; 40% por claridad de explicación y manejo de percepciones.

## **Unidad 5: UNIDAD 5: Estructuras de datos simples: listas para percepciones y acciones**

### **Objetivos de Aprendizaje**

- Crear y manipular listas para almacenar percepciones y acciones.
- Acceder y modificar elementos de una lista en Python.
- Modelar un agente que toma decisiones basadas en percepciones representadas como listas.

### **Contenidos Temáticos**

1. **Tema 1: Listas en Python** - Descripción: Crear listas, acceder a elementos y métodos básicos (append, len).
2. **Tema 2: Modelando percepciones y acciones con listas** - Descripción: Almacenar percepciones y acciones en listas para un agente simple.
3. **Tema 3: Manipulación básica de listas** - Descripción: Seleccionar acciones según percepciones y operaciones simples de listas.

### **Actividades**

- **Actividad 1: Registro de percepciones** - Construir una lista de percepciones de ejemplo y observar cómo cambia la acción asociada a lo largo del tiempo.
- **Actividad 2: Decisión basada en listas** - Escribir código que, dada una lista de percepciones, seleccione una acción correspondiente y la registre en una lista de acciones.
- **Actividad 3: Mini proyecto** - Diseñar un pequeño agente que lee percepciones de una lista y actualiza una lista de acciones en cada paso.

### **Evaluación**

OA4, OA5 y OA6 cubiertos mediante ejercicios de manipulación de listas y toma de decisiones, con valoración de claridad y eficiencia.

## **Unidad 6: UNIDAD 6: Depuración de errores y buenas prácticas**

### **Objetivos de Aprendizaje**

- Identificar errores de sintaxis comunes en Python y corregirlos.
- Detectar errores lógicos que afectan la solución y proponer correcciones.
- Aplicar estrategias de depuración (impresión de valores, pruebas paso a paso) para mejorar el funcionamiento del código.

### **Contenidos Temáticos**

1. **Tema 1: Errores de sintaxis** - Descripción: Localizar y corregir errores de sintaxis en código.
2. **Tema 2: Errores lógicos** - Descripción: Identificar fallos en la lógica que producen resultados incorrectos.
3. **Tema 3: Estrategias de depuración** - Descripción: Técnicas de depuración, pruebas y registro de resultados para mejorar el código.

## Actividades

- **Actividad 1: Corrección de código** - Se entrega un programa con errores; corregir y justificar cambios realizados.
- **Actividad 2: Pruebas de casos** - Elaborar casos de prueba para un programa y verificar que funcione en distintos escenarios.
- **Actividad 3: Revisión entre pares** - Intercambiar código con un compañero para detectar errores y proponer mejoras.

## Evaluación

OA6 y OA7 cubiertos: 40% depuración de errores de sintaxis, 40% detección de errores lógicos, 20% explicación de prácticas de depuración y mejoras.

## Unidad 7: UNIDAD 7: Simulación de un agente de IA simple

### Objetivos de Aprendizaje

- Describir a nivel conceptual el modelo de agente-entorno y cómo se evalúan percepciones para seleccionar acciones.
- Implementar en Python una versión simple de un agente que toma decisiones basada en percepciones y reglas simples.
- Analizar limitaciones de este enfoque frente a IA más complejas (aprendizaje, percepciones más amplias, etc.).

### Contenidos Temáticos

1. **Tema 1: Modelo agente-entorno** - Descripción: Conceptos de agente y entorno, percepciones y acciones en IA.
2. **Tema 2: Regla simple de decisión** - Descripción: Reglas tipo si/entonces para decidir acciones.
3. **Tema 3: Implementación básica** - Descripción: Programa simple que simula un agente en respuesta a percepciones.

## Actividades

- **Actividad 1: Diálogo entre percepciones y acciones** - Diseña percepciones y escribe reglas de decisión para generar acciones; aprendizaje: entender la correspondencia perceive-acción.
- **Actividad 2: Implementación en Python** - Escribe un programa que modele un agente simple con percepciones y una decisión basada en condiciones. Aprendizajes: aplicar el concepto en código.

- **Actividad 3: Análisis de limitaciones** - Debatar y anotar limitaciones del enfoque simple frente a IA avanzada.

## Evaluación

OA4, OA5 y OA8: evaluación de la capacidad de diseñar y codificar una decisión de agente y reflexión sobre limitaciones.

## Unidad 8: UNIDAD 8: Proyecto integrador: diseño de un agente básico en un entorno simulado

### Objetivos de Aprendizaje

- Crear un programa que integre percepciones, acciones y una regla de decisión en Python.
- Utilizar estructuras de control y estructuras de datos simples para simular un entorno y un agente.
- Analizar y documentar limitaciones frente a enfoques de IA más complejos y proponer mejoras futuras.

### Contenidos Temáticos

1. **Tema 1: Diseño del entorno simulado** - Descripción: Definir percepciones, acciones y reglas de interacción entre entorno y agente.
2. **Tema 2: Implementación modular** - Descripción: Organizar el código para separar percepciones, decisiones y acciones.
3. **Tema 3: Evaluación y reflexión** - Descripción: Pruebas, depuración y análisis de limitaciones y mejoras posibles.

### Actividades

- **Actividad 1: Planificación del proyecto** - Escribir un plan que describa percepciones, entorno y reglas de decisión; identificar entrada y salida. Aprendizajes: diseño de un sistema simple.
- **Actividad 2: Implementación en Python** - Desarrollar el código modular que simula el agente y el entorno, con pruebas básicas.
- **Actividad 3: Presentación y reflexión** - Presentar el proyecto, explicar su funcionamiento y discutir limitaciones y posibles mejoras.

## Evaluación

OA1, OA4, OA5 y OA8 evaluados a través del proyecto final: calidad de diseño, implementación, documentación y reflexión sobre limitaciones.