



Facultad: INGENIERÍA

Programa: INGENIERÍA ELECTRÓNICA

1. Identificación del curso

Nombre: Programación I

Área: Básica

Código: BEINGEL091

Número de créditos: 3

Horas de acompañamiento directo:	64	Horas de trabajo independiente:	80	Total horas:	144
---	-----------	--	-----------	---------------------	------------

Carácter del curso (Teórico, práctico o teórico práctico): Teórico práctico

Componente Básico o complementario: Básico

Requisito: N/A

Unidad responsable del microdiseño: Ingeniería Electrónica

2. Presentación del curso

El curso de Programación I tiene como objetivo principal orientar al estudiante en la adquisición de habilidades fundamentales en lógica de programación. A lo largo de este proceso, se enfoca en la elaboración de algoritmos y su consecuente implementación en un lenguaje específico. Además, este curso introduce al estudiante al paradigma orientado a objetos, haciendo uso de herramientas visuales, como Lenguaje de Modelado Unificado (UML), para simplificar el proceso del diseño y especificación de sistemas computacionales.

Este espacio académico resalta la importancia del análisis de requerimientos de software, así como en el diseño e implementación de soluciones óptimas y escalables. A lo largo del curso, los estudiantes no solo desarrollarán competencias técnicas, sino que también cultivarán una comprensión profunda de cómo abordar de manera efectiva los desafíos asociados con el desarrollo de software en la actualidad. Este curso no se limita a presentar los fundamentos de programación, sino por el contrario comprender las metodologías y estrategias clave que respaldan el proceso de desarrollo de aplicaciones de software de manera eficiente y de calidad.

3. Justificación

El curso de Programación I está enfocado en estudiar los principios fundamentales de la informática, y su posterior aplicación en situaciones reales. En cuanto a la metodología de enseñanza, esté parte de lo sencillo a lo complejo, facilitando la comprensión gradual de los conceptos y temas mediante un proceso inductivo, este enfoque pedagógico facilita el desarrollo de habilidades técnicas y de ingeniería que permiten a los estudiantes solucionar problemas de programación en forma eficiente.

Vigilada Mineducación

La versión vigente y controlada de este documento, solo podrá ser consultada a través del sitio web Institucional www.usco.edu.co, link Sistema Gestión de Calidad. La copia o impresión diferente a la publicada, será considerada como documento no controlado y su uso indebido no es de responsabilidad de la Universidad Surcolombiana.



Actualmente, el mercado laboral exige a los desarrolladores principiantes un dominio profundo en lógica de programación, estructuras de control, y manipulación de datos, ya que son habilidades básicas de programación, las cuales, combinadas con otras, de tipo corporativas o empresariales permiten al desarrollador enfrentar problemas complejos con soluciones innovadoras. A lo largo del curso, se enfatiza la aplicación práctica de estos conocimientos, preparando a los estudiantes no solo para resolver problemas inmediatos, sino también a enfrentar desafíos propios de un profesional en desarrollo software.

Conscientes de esta realidad, el curso se enfoca en introducir de manera exhaustiva los conceptos clave de la programación orientada a objetos, incluyendo clases, objetos, herencia y encapsulamiento. Estos fundamentos son básicos para el desarrollo de software modular, reutilizable y mantenible, competencias altamente valoradas en el sector tecnológico por su impacto directo en la eficiencia y calidad de los proyectos.

Al dominar tanto los principios básicos como los avanzados de la programación orientada a objetos, los estudiantes estarán preparados para enfrentar cursos superiores y desafíos reales en el desarrollo de software. Este conocimiento no solo proporciona una base para el crecimiento académico y profesional, sino que también estimula la creatividad y la innovación, permitiendo la creación de soluciones prácticas y aplicables en diversos contextos. Por ello, el curso de Programación I constituye un primer paso para quienes buscan destacarse en el ámbito tecnológico y contribuir de manera significativa al avance de la industria.

4. Competencias

1. La capacidad de identificar, formular y resolver problemas complejos de ingeniería mediante la aplicación de principios de ingeniería, ciencias y matemáticas.
2. Capacidad para comunicarse de manera efectiva con una variedad de audiencias.
3. Capacidad para funcionar de manera efectiva en un equipo cuyos miembros juntos brindan liderazgo, crean un entorno colaborativo e inclusivo, establecen metas, planifican tareas y cumplen objetivos.
4. Capacidad de desarrollar y realizar experimentos apropiados, analizar e interpretar datos y utilizar el juicio de ingeniería para sacar conclusiones.
5. Capacidad de adquirir y aplicar nuevos conocimientos según sea necesario, utilizando estrategias de aprendizaje adecuadas.



5. Resultados de aprendizaje, actividades académicas y estrategias de evaluación

Resultados de Aprendizaje	Actividades Académicas	Estrategias de Evaluación
Utiliza adecuadamente la lógica de programación, resultando en la identificación y aplicación de soluciones óptimas a diversos problemas.	<ul style="list-style-type: none">• Clase magistral.• Retos de programación.• Aprendizaje colaborativo basado en problemas.• Análisis de casos de estudio.	<ul style="list-style-type: none">• Desarrollo de talleres.• Elaboración de proyecto.• Revisión de código entre pares.• Presentación y defensa de soluciones a problemas propuestos.• Actividades de clase (ejercicios, evaluaciones rápidas, elaboración de mapas conceptuales, entre otros).
Analiza y diseña sistemas de manera idónea, asegurando la conformidad con los requisitos establecidos en un entorno sistémico.	<ul style="list-style-type: none">• Clase magistral.• Estudio de casos.• Uso de herramientas de modelado.• Análisis y diseño de sistemas software.	<ul style="list-style-type: none">• Desarrollo de talleres.• Elaboración de proyecto de modelado.• Presentación y defensa de soluciones.• Actividades de clase (ejercicios, evaluaciones rápidas, elaboración de mapas conceptuales, entre otros).
Implementa los requisitos de un sistema de software en el lenguaje más pertinente, con el propósito de ofrecer soluciones apropiadas y perfectamente adaptadas al sistema en consideración.	<ul style="list-style-type: none">• Clase magistral.• Proyectos integrales de desarrollo software.• Implementación de requerimientos del sistema.	<ul style="list-style-type: none">• Desarrollo de talleres.• Desarrollo de un sistema completo.• Elaboración y defensa del proyecto.• Revisión de código en equipo.• Actividades de clase (ejercicios, cuestionarios de comprensión inmediata, evaluaciones rápidas, elaboración de mapas conceptuales, entre otros).
Aplica con habilidad los conceptos orientados a objetos para generar soluciones de alta calidad y fácilmente extensibles.	<ul style="list-style-type: none">• Clase magistral.• Implementación de patrones de diseño.• Análisis y diseño orientado a objetos.	<ul style="list-style-type: none">• Desarrollo de talleres• Elaboración de proyecto aplicando orientación a objetos y patrones de diseño.• Evaluación de diagramas y documentación.

Vigilada Mineducación

La versión vigente y controlada de este documento, solo podrá ser consultada a través del sitio web Institucional www.usco.edu.co, link Sistema Gestión de Calidad. La copia o impresión diferente a la publicada, será considerada como documento no controlado y su uso indebido no es de responsabilidad de la Universidad Surcolombiana.



MICRODISEÑO CURRICULAR

CÓDIGO

MI-FOR-FO-34

VERSIÓN

2

VIGENCIA

2022

Página

4 de 8

		<ul style="list-style-type: none">• Actividades de clase (ejercicios, cuestionarios de comprensión inmediata, evaluaciones rápidas, elaboración de mapas conceptuales, entre otros).
Comunica ideas y soluciones de programación de forma clara y comprensible. Además, trabaja eficientemente en equipos, demostrando liderazgo, fomentando un ambiente colaborativo e inclusivo, estableciendo metas, planificando tareas y alcanzando objetivos comunes.	<ul style="list-style-type: none">• Presentaciones y sustentaciones orales.• Escritura de informes técnicos.• Desarrollo de proyectos colaborativos.	<ul style="list-style-type: none">• Presentaciones y sustentaciones orales.• Escritura de informes técnicos.• Desarrollo de proyectos colaborativos.

6. Evaluación general del curso

Resultados de aprendizaje	Desempeño deseado				
Utiliza adecuadamente la lógica de programación, resultando en la identificación y aplicación de soluciones óptimas a diversos problemas.	El estudiante aplica de manera coherente la lógica de programación, utilizando estructuras y algoritmos avanzados cuando es apropiado; identifica y justifica soluciones óptimas para una amplia gama de problemas nuevos y complejos; escribe código eficiente y optimizado siguiendo las mejores prácticas; aprovecha diversas herramientas y recursos avanzados de programación para mejorar las soluciones y la eficiencia del desarrollo; y proporciona documentación clara y completa con un estilo de código consistente y profesional que facilita el mantenimiento y la escalabilidad.				
	Completa mente alcanzado (100%)	Alcanzado en alto grado (70-90%)	Alcanzado de manera aceptable (50-70%)	Aun no alcanzado (10-50%)	Aun no intentado (0-10%)
Analiza y diseña sistemas de manera idónea, asegurando la conformidad con los requisitos establecidos en un entorno sistémico.	El estudiante analiza y diseña sistemas de manera organizada, demostrando una comprensión profunda de los principios sistémicos; asegura que todos los requisitos establecidos se cumplen al detalle, integrando eficientemente los componentes del sistema en un entorno coherente y funcional; anticipa y resuelve proactivamente posibles conflictos o inconsistencias, y proporciona soluciones innovadoras y eficientes que optimizan el rendimiento y la eficacia del sistema en su conjunto.				
	Completa mente alcanzado	Alcanzado en alto grado	Alcanzado de manera aceptable	Aun no alcanzado (10-50%)	Aun no intentado (0-10%)

Vigilada Mineducación

La versión vigente y controlada de este documento, solo podrá ser consultada a través del sitio web Institucional www.usco.edu.co, link Sistema Gestión de Calidad. La copia o impresión diferente a la publicada, será considerada como documento no controlado y su uso indebido no es de responsabilidad de la Universidad Surcolombiana.



MICRODISEÑO CURRICULAR

CÓDIGO

MI-FOR-FO-34

VERSIÓN

2

VIGENCIA

2022

Página

5 de 8

	(100%)	(70-90%)	(50-70%)		
Implementa los requisitos de un sistema de software en el lenguaje más pertinente, con el propósito de ofrecer soluciones apropiadas y perfectamente adaptadas al sistema en consideración.	El estudiante implementa los requisitos de un sistema de software utilizando de manera correcta el lenguaje de programación seleccionado; ofrece soluciones apropiadas y perfectamente adaptadas al sistema en consideración, demostrando una comprensión profunda de los principios de desarrollo de software; traduce eficientemente los requisitos en código funcional y optimizado, asegurando que las soluciones sean escalables, mantenibles y alineadas con las mejores prácticas de la industria.	Completamente alcanzado (100%)	Alcanzado en alto grado (70-90%)	Alcanzado de manera aceptable (50-70%)	Aun no alcanzado (10-50%) Aun no intentado (0-10%)
Aplica con habilidad los conceptos orientados a objetos para generar soluciones de alta calidad y fácilmente extensibles.	El estudiante aplica con habilidad los conceptos asociados a la orientación a objetos, demostrando una comprensión profunda y habilidad para utilizarlos de manera eficiente; genera soluciones de software de alta calidad que no solo cumplen con los requisitos funcionales sino que también son fácilmente extensibles y mantenibles; implementa principios como encapsulación, herencia, polimorfismo y abstracción de forma óptima, facilitando la escalabilidad y adaptabilidad del código; además, sigue las mejores prácticas de diseño orientado a objetos, asegurando que las soluciones sean robustas, modulares y alineadas con los estándares de la industria.	Completamente alcanzado (100%)	Alcanzado en alto grado (70-90%)	Alcanzado de manera aceptable (50-70%)	Aun no alcanzado (10-50%) Aun no intentado (0-10%)
Comunica ideas y soluciones de programación de forma clara y comprensible. Además, trabaja eficientemente en equipos, demostrando liderazgo, fomentando un ambiente colaborativo e inclusivo, estableciendo metas, planificando tareas y alcanzando objetivos comunes.	El estudiante comunica ideas y soluciones de programación en forma clara y comprensible, utilizando un lenguaje preciso y adecuado que facilita la comprensión por parte de diversas audiencias; además, trabaja eficientemente en equipo, demostrando liderazgo al dirigir y motivar al grupo, fomentando un ambiente colaborativo e inclusivo, estableciendo metas claras, planificando tareas de manera efectiva y contribuyendo activamente al logro de objetivos comunes.	Completamente alcanzado (100%)	Alcanzado en alto grado (70-90%)	Alcanzado de manera aceptable (50-70%)	Aun no alcanzado (10-50%) Aun no intentado (0-10%)

Vigilada Mineducación

La versión vigente y controlada de este documento, solo podrá ser consultada a través del sitio web Institucional www.usco.edu.co, link Sistema Gestión de Calidad. La copia o impresión diferente a la publicada, será considerada como documento no controlado y su uso indebido no es de responsabilidad de la Universidad Surcolombiana.



MICRODISEÑO CURRICULAR

CÓDIGO

MI-FOR-FO-34

VERSIÓN

2

VIGENCIA

2022

Página

6 de 8

7. Unidades temáticas, estrategias didácticas y tiempo asignado

No.	Unidades y contenidos	Estrategias didácticas	Horas				
			Acompañamiento directo		Trabajo Independiente		sTotal
			Teóricas	Teórico-Prácticas	Prácticas	Independiente	
1	Introducción a Python. Sintaxis y comentarios. Variables y tipos de datos. Estructuras de control y repetitivas.	Clases magistrales. Talleres, proyectos dentro y fuera del aula, y etc.	20			25	45
2	Funciones. Listas, tuplas, diccionarios, conjuntos.	Clases magistrales. Talleres, proyectos dentro y fuera del aula, y etc.	20			25	45
3	Programación orientada a objetos, clases, objetos, herencia, polimorfismo, y relaciones entré objetos.	Clases magistrales. Talleres, proyectos dentro y fuera del aula, y etc.	24			30	54
Totales			64			80	144
Total							

*Entiéndase por práctica las actividades académicas realizadas en espacios formativos, donde se contrastan los fundamentos teóricos y prácticos.

**Especificar la naturaleza de la práctica (Clínica, Pedagógica, Laboratorio, etc.)

Vigilada Mineducación

La versión vigente y controlada de este documento, solo podrá ser consultada a través del sitio web Institucional www.usco.edu.co, link Sistema Gestión de Calidad. La copia o impresión diferente a la publicada, será considerada como documento no controlado y su uso indebido no es de responsabilidad de la Universidad Surcolombiana.



8. Referencias bibliográficas

1. Oliver, R. (2023). Python QuickStart Guide: The Simplified Beginner's Guide to Python Programming Using Hands-On Projects and Real-World Applications. Clydebank Media LLC.
2. Matthes, E. (2023). Python Crash Course, 3rd Edition. No Starch Press.
3. Sweigart, A. (2019). Automate the Boring Stuff with Python, 2nd Edition: Practical Programming for Total Beginners. No Starch Press.
4. Ramalho, L. (2021). Fluent Python: Clear, Concise, and Effective Programming.
5. Lutz, M. (2013). Learning Python (5th ed.). O'Reilly Media.
6. Robbins, P. (2023). Python Programming for Beginners: The Complete Guide to Mastering Python in 7 Days with Hands-on Exercises - Top Secret Coding Tips to Get an Unfair Advantage and Land Your Dream Job!
7. McKinney, W. (2022). Python for data analysis: Data Wrangling with Pandas, NumPy, and Jupyter. O'Reilly Media.
8. Fowler, M. (2004). UML distilled: A Brief Guide to the Standard Object Modeling Language. Addison-Wesley Professional.
9. Miles, R., Hamilton, K. (2008). Learning UML 2.0. O'Reilly Media.
10. Pilone, D., Pitman, N. (2005). UML 2.0 in a Nutshell: A Desktop Quick Reference (1st ed.). O'Reilly Media.
11. Samek, M. (2008). Practical UML State charts in C/C++, Second Edition: Event-Driven Programming for Embedded Systems.
12. Rumbaugh, J., Jacobson, I., Booch, G. (1998). The unified modeling language reference manual.
13. Dennis, A. R., Wixom, B. H., Tegarden, D. P. (2016). Systems Analysis and Design: An Object-Oriented Approach with UML (5th edition).
14. Martin, R. C. (2018). Clean architecture: A Craftsman's Guide to Software Structure and Design. Pearson Professional.
15. Pressman, R. S., Maxim, B. R. (2019). Software Engineering: A Practitioner's Approach.
16. Raúl Gonzáles Duque. Python para todos. <http://mundogeek.net/tutorial-python/> . 2020.
17. Jon Vadillo Romero. Aprende Python de cero a experto. <https://dev.to/jvadillo/aprende-python-desde-cero-a-experto-4eej>. Editorial LeanPub. 2020
18. https://www.w3schools.com/python/python_arrays.asp



MICRODISEÑO CURRICULAR

CÓDIGO

MI-FOR-FO-34

VERSIÓN

2

VIGENCIA

2022

Página

8 de 8

9. Trazabilidad de la evaluación del microdiseño

Fecha de evaluación actualización y aprobación por el comité de currículo (número de acta)	Modificación	Justificación	Responsables
	Se realiza una actualización al último formato MI-FOR-FO-34, versión 2, año 2022.	Es necesario actualizar para atender la solicitud a el registro calificado	Albeiro Cortés Cabezas
10 de octubre de 2024	Se actualiza las secciones: presentación, justificación, resultados de aprendizaje, resultados de aprendizaje, y evaluación.	Ajuste al modelo institucional	José Fernando Barrera Campos

Vigilada Mineducación

La versión vigente y controlada de este documento, solo podrá ser consultada a través del sitio web Institucional www.usco.edu.co, link Sistema Gestión de Calidad. La copia o impresión diferente a la publicada, será considerada como documento no controlado y su uso indebido no es de responsabilidad de la Universidad Surcolombiana.