

Programa de Asignatura

1. Identificación Asignatura

Nombre:	Programación Avanzada				Código:		
Carrera:	Ingeniería Civil Informática		Unidad Académica:		Ciencias Naturales y Tecnología		
Ciclo Formativo:	Ciclo Inicial		Línea formativa:		Básica		
Semestre	III		Tipo de actividad:		Obligatoria		
N° SCT:	6	Horas Cronológicas Semanales					
		Presenciales:	3	Trabajo Autónomo:		6	
Pre-requisitos	Fundamentos de Programación						

2. Propósito formativo

La asignatura de Programación Avanzada tiene como propósito que el/la estudiante aprenda a desarrollar soluciones de programación siguiendo el paradigma orientado a objetos (POO).

La asignatura aborda conceptos de orientación a objetos al mismo tiempo que los aplica usando el lenguaje de programación *Kotlin*¹. El/la estudiante ira conociendo y aplicando elementos como objetos, clases, herencia, abstracción y polimorfismo, entre otros. El contexto del desarrollo de habilidades de aplicación metodológica será a través de problemas específicos, definidos en diversos dominios de aplicación cuyas soluciones se encontrarán delimitadas en cuanto al alcance y complejidad.

Esta asignatura plantea nuevos paradigmas de programación a los vistos hasta este momento, y promueve su aplicación en una asignatura práctica del mismo nivel que es Taller de Ingeniería III. De forma incipiente, los y las estudiantes comenzarán a trabajar con diferentes estructuras de datos que serán profundizadas en la siguiente asignatura de especialidad que es Algoritmos y Estructuras de Datos.

3. Contribución al perfil de egreso

Esta asignatura contribuye a los siguientes desempeños o resultados de aprendizaje globales declarados en el Perfil de Egreso de la carrera:

1. Entiende problemas a través de la construcción de abstracciones conceptuales, cualitativas y cuantitativas, utilizando formalismos establecidos, que permitan formular soluciones
2. Diseña y programa soluciones, utilizando estrategias algorítmicas, que permitan resolver problemas de forma eficaz y acorde a múltiples objetivos de diseño.

4. Resultados de aprendizaje específicos

Resultado de Aprendizaje Específico	Criterios de evaluación	Evidencia
RA1. Reconoce los componentes del paradigma de programación orientado a objetos, definiendo cada uno de ellos y reconociéndolos en contextos de uso.	1.1. Comprende los conceptos del paradigma de programación orientada a objetos tales como: polimorfismo, encapsulamiento, herencia, sobrecarga, abstracción, etc.	Laboratorios, guías de ejercicio.

¹ <https://kotlinlang.org/>

	1.2. Define y diagrama clases y sus componentes: campos, constructor, métodos accesores y mutadores. 1.3. Modela soluciones a problemas utilizando el paradigma de programación orientada a objetos.	
RA2. Aprende a utilizar el lenguaje de programación <i>Kotlin</i> a través de la aplicación de los componentes que subyacen el paradigma de programación orientado a objetos.	1.4. Identifica diferencias entre los lenguajes de programación Python y Kotlin. 1.5. Logra elaborar de manera autónoma programas en Kotlin utilizando el POO que resuelven problemas simples.	Laboratorios, guías de ejercicio.
RA3. Programa/utiliza funciones/librerías en Kotlin considerando clases y objetos, creando soluciones de software de propósito general o adaptando código existente, para satisfacer requerimientos nuevos o desarrollando aplicaciones a partir de requerimientos nuevos.	1.6. El/la estudiante logran desarrollar soluciones de forma autónoma a problemas de diversa complejidad utilizando el lenguaje Kotlin y utilizando los conceptos de POO.	Laboratorios, guías de ejercicio.

5. Unidades de Aprendizaje

1. Introducción a la Orientación a Objetos (OO) 1.1. Paradigmas de programación 1.2. Conceptos generales de OO 1.3. Abstracción y pensamiento inductivo
2. Programación en Kotlin con OO 2.1. Introducción al lenguaje Kotlin 2.2. Manejo y uso de clases y objetos 2.3. Manejo de estado y referencias 2.4. Diseño y uso de colecciones
3. Mantenibilidad y extensibilidad 3.1. Clases abstractas e interfaces 3.2. Polimorfismo 3.3. Principios de Diseño Orientado a Objetos.

6. Recursos de Aprendizaje

Bibliografía:

- B1. Colección de enlaces relevantes para el curso: <https://raindrop.io/enrique-urra/programacion-avanzada-in-1002-23469476>
- B2. Oviedo Regino, E. (2015). Lógica de programación orientada a objetos. Ecoe Ediciones. <https://elibro.net/es/lc/uaysen/titulos/70431>

Recursos materiales e infraestructura:

- Laboratorio de computación.

- Acceso a Ucampus.
- Acceso a Google Classroom con credenciales institucionales.
- Computadores debidamente equipados para utilizar lenguajes de alto nivel (por ej.: Kotlin).

7. Comportamiento y ética académica:

Se espera que los estudiantes actúen en sus diversas actividades académicas y estudiantiles en concordancia con los principios de comportamiento ético y honestidad académica propios de todo espacio universitario y que están estipulados en el *Reglamento de Estudiantes de la Universidad de Aysén*, especialmente aquéllos dispuestos en los artículos 23°, 24° y 26°.

Todo acto contrario a la honestidad académica realizado durante el desarrollo, presentación o entrega de una actividad académica del curso sujeta a evaluación, será sancionado con la suspensión inmediata de la actividad y con la aplicación de la nota mínima (1.0).

Planificación del curso

8. Responsables

Académico (s) Responsable (s) y equipo docente	Profesor: Enrique Urrea Coloma		
Contacto	Correo: enrique.urra@uaysen.cl		
Año	2022	Periodo Académico	Primer semestre
Horario clases	Cátedra: lunes y miércoles, 18.00 a 19.30 hrs.	Horario de atención estudiantes	Jueves, 10:00 hrs.
Sala / Campus	Campus Río Simpson, Sala A1 (lunes) y Sala A2 (miércoles).		

9. Metodología de Trabajo:

La asignatura contiene:			
Actividades de vinculación con el medio	No	Actividades relacionadas con proyectos de investigación	No
En el curso se contemplan cuatro tipos de actividades docentes, las cuales se asocian a requerimientos de sala y al nivel de intervención del profesor:			
Actividad docente	Descripción	Intervención del profesor/ayudante	Requerimiento de sala
Exposición conceptual	El profesor introduce conceptos de programación preliminares y necesarios a otras actividades de índole práctica, de forma expositiva. Se dispone de materiales complementarios en la plataforma Classroom.	Alta	Sala de clases Classroom
Programación expositiva	El profesor profundiza en la comprensión de elementos conceptuales a través de la exposición directa de la resolución de problemas de programación como ejemplos.	Alta	Sala de clases Classroom
Programación tutorial	Funciona como la programación expositiva, pero el profesor realiza pausas para que los alumnos completen "pasos requeridos" antes de continuar. El objetivo es que todos los alumnos completen un paso definido por el profesor antes de continuar al siguiente.	Media	Laboratorio de computación Computador persona
Actividad práctica / Programación autónoma	Los estudiantes abordan y resuelven problemas de programación de forma autónoma, algunas con guía y apoyo docente y otras no.	Baja/Media	Laboratorio de computación Computador persona

10. Evaluaciones:

Evaluación	Ponderaciones específicas	Ponderación nota presentación
Pruebas de Cátedra	- Prueba de Cátedra 1 (PC1): 35% - Prueba de Cátedra 2 (PC2): 65%	70%
Proyecto		30%

Calificación final:

- Nota de presentación: 70%
- Examen Final: 30 %

Condiciones de eximición:

- Nota de presentación igual o superior a nota 5,0
- Ponderación específica Pruebas de Cátedra $\geq 4,0$

Derecho a rendir examen:

Nota de presentación $\geq 3,5$

Requisito de Aprobación

- Asistencia: 70%
- Nota Final: 4,0

11. Otros aspectos asociados al funcionamiento del curso:

Toda la coordinación del curso (comunicaciones, actualización de notas, material, etc.) será realizada a través de Google Classroom. El estudiante deberá informar con tiempo suficiente si presenta dificultades de conexión para trasladar el requerimiento a la coordinación de programa. Adicionalmente los estudiantes deberán acceder al material disponible que el profesor preparó para este propósito.

Es deber del estudiante mantenerse informado de las noticias, avisos y material entregado por el profesor a través de estos medios.

12. Planificación de las actividades de enseñanza- aprendizaje y de evaluación

Semana	Día inicio	Resultado(s) de Aprendizaje	Unidades de aprendizaje y actividades	Recursos utilizados o lecturas	Actividad(es) de Trabajo Autónomo
1	14-mar	RA1	1.1	B1	Revisión de material / ejercicios
2	21-mar	RA1	1.2, 1.3	B1	Revisión de material / ejercicios
3	28-mar	RA2	2.1	B2	Revisión de material / ejercicios
4	4-abr	RA2	2.2, 2.3	B2	Revisión de material / ejercicios
5	11-abr	RA2	2.4	B2	Revisión de material / ejercicios
6	18-abr		Ejercitación		Revisión de material / ejercicios
7	25-abr		27-abr: Prueba de Cátedra 1		Revisión de material / ejercicios
8	2-may	Receso universitario (docencia)			
9	9-may	RA2	2.4	B2	Revisión de material / ejercicios
10	16-may	RA2	3.1	B2	Revisión de material / ejercicios
11	23-may	RA2	25-may: Claustro Académico 3.2	B2	Revisión de material / ejercicios
12	30-may	RA2	3.3	B2	Revisión de material / ejercicios
13	6-jun		8-jun: Prueba de Cátedra 2		Revisión de material / ejercicios

14	13-jun	RA1, RA2	Trabajo proyecto	B1, B2	Trabajo en proyecto
15	20-jun	RA1, RA2	20-jun: Interferiado Trabajo proyecto	B1, B2	Trabajo en proyecto
16	27-jun	RA1, RA2	27-jun: Feriado Trabajo proyecto	B1, B2	Trabajo en proyecto
17	4-jul	RA1, RA2	6-jul: Término Proyecto	B1, B2	Trabajo en proyecto
18	11-jul		Semana de exámenes 13-jul: Examen final		
19	18-jul		Cierre de acta		