PROGRAMA DE CURSO

INTRODUCCION A LA PROGRAMACIÓN

A. Antecedentes generales del curso:

Departamento	Ciencias de la computación					
Nombre del curso	Introducción a la programación		Código	CC1002	Créditos	6
Nombre del curso en inglés	*Introduction to Programming*					
Horas semanales	Docencia	3,5	Auxiliares	2,0	Trabajo personal	4,5
Carácter del curso	Obligatorio	X		Electivo		
Requisitos						

B. Propósito del curso:

El curso tiene como propósito que los estudiantes resuelvan problemas utilizando la programación, siguiendo una ruta metodológica determinada, traduciendo, reformulando y formalizando enunciados con el propósito de generar programas capaces de dar respuestas a las distintas peticiones y finalidades.

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 tamaño.

Se espera que los y las estudiantes desarrollen una metodología de trabajo que los lleve a adquirir rigor procedimental para enfrentarse a la resolución de problemas, en base al razonamiento algorítmico y lógico.

Las competencias especificas (CE) y genéricas (CG) a las que tributa el curso son:

CE6: Aplicar una metodología de diseño e implementación para escribir programas computacionales, en la resolución de problemas, utilizando herramientas computacionales para manejar y visualizar datos.

CE7: Modelar matemáticamente el comportamiento de los agentes (consumidores, empresas, organizaciones, países, entre otros), aplicando conceptos fundamentales del análisis económico parala toma de decisiones.

CG2: Compromiso ético

Reflexionar sobre el propio actuar y sus consecuencias, en el marco de la honestidad, la responsabilidad y el respeto, buscando la excelencia y rigurosidad en su proceder en contextos académicos, en las relaciones interpersonales y con su entorno.

C. Resultados de aprendizaje:

Competencias específicas	Resultados de aprendizaje
CE6 – CE7	RA1: Descompone analíticamente un problema enunciado, deduciendo los datos de entrada, de salida, o efectos esperados de un programa y derivando sus posibles ejemplos de uso, a fin de llegar a la descomposición irreductible del problema.
CE6	RA2: Implementa programas computacionales a partir de la descomposicióndel problema y de los elementos existentes, para obtener una soluciónejecutable al problema.
CE6	RA3: Verifica la solución implementada a partir del comportamiento esperado, con el fin de validar y/o rectificar dichas implementaciones.
Competencias genéricas	**Resultados de aprendizaje**
CG2	RA4: Realiza las tareas de manera responsable y honesta, sin incurrir en plagio,copia o suplantación de identidad.

D. Unidades temáticas:

Número	RA al que tributa	Nombre de la unidad	Duración en semanas
1	RA1, RA2, RA3, RA4	Fundamentos de Programación	4 semanas
Contenidos		**Indicador de logro**	
1.1. Expresiones y tipos de datos básicos. 1.2. Funciones. 1.3. Diseño de programas. 1.4. Programación modular. 1.5. Expresiones booleanas y condicionales. 1.6. Recursión. 1.7. Testing y depuración.		El/la estudiante: 1. Identifica y escribe expresiones usando los tipos de datos básicos (entero, real, texto, boolean). 2. Reconoce el concepto de función en programación. 3. Explica el concepto de módulo, entendiéndolo como un conjunto de funciones relacionadas entresí. 4. Utiliza operadores relacionales y conectores lógicos para escribir expresiones booleanas. 5. Utiliza recursión en la resolución de problemas. 6. Analiza los elementos de un problema propuesto: datos de entrada y sus tipos de datos básico, dato de salida y propósito, a fin de facilitar la descomposición del problema. 7. Descompone el problema de ser necesario en subproblemas, considerando el propósito	

	identificado. 8. Identifica las funciones que resuelven el problemaa partir del análisis realizado. 9. Utiliza la "Receta de diseño" (Felleisen M., 2001) enla programación de una función. 9.1 Documenta su propósito principal. 9.2 Escribe su firma (nombre, tipos de argumentos y tipo de retorno). 9.3 Escribe programas de ejemplos de su uso comentando resultados esperados. 9.4 Programa el cuerpo de la función usando los elementos identificados previamente. 9.5 Utiliza nombres representativos para las funcionesy variables. 9.6 Verifica que las funciones tengan un propósito único. 10. Verifica que la función esté correctamente programada a través de la aplicación de testing, y corrige el código en caso de detectar errores. 11. Cumple obligaciones y acuerdos, respetando los compromisos adquiridos en sus actividades académicas. 12. Planifica y presenta sus trabajos, basándose en sus capacidades, sin incurrir en plagio, copia, suplantación de identidad.
Bibliografía de la unidad	[1] Capítulos 2, 3, 4, 9.

Número	RA al que tributa	Nombre de la unidad	Duración en semanas
2	RA1, RA2, RA3, RA4	Programación Funcional	4 semanas

Contenidos	Indicador de logro
2.1. Datos compuestos. 2.2. Estructuras de datos recursivas. 2.3. Abstracción funcional.	El/ la estudiante: 1. Define y programa tipos de datos compuestos. 2. Define estructuras de datos recursivas. 3. Resuelve problemas utilizando estructuras de datos recursivas. 4. Reconoce el concepto de abstracción funcional. 5. Utiliza abstracción funcional para combinar funciones relacionadas en una única función. 6. Programa funciones que consideran datos compuestos y estructuras de datos recursivas, utilizando la Receta de diseño (Felleisen M., 2001). 7. Cumple obligaciones y acuerdos, respetando los

	compromisos adquiridos en sus actividades académicas. 8. Planifica y presenta sus trabajos, basándose en sus capacidades, sin incurrir en plagio, copia, suplantación de identidad.
Bibliografía de la unidad	[1] Capítulos 6, 9, 10, 12, 14, 15, 17, 19, 20, 21, 22, 29.

Número	RA al que tributa	Nombre de la unidad	Duración en semanas
3	RA1, RA2, RA3, RA4	Programación Imperativa	2 semanas
Contenidos		Indicador de logro	
3.1. Mutación y aliasing 3.2. Estructuras indexadas 3.3. Archivos de texto. 3.4. Testing y depuración en programación imperativa.		El/la estudiante: 1. Identifica los conceptos de mutación y aliasing. 2. Utiliza variables de estado para diseñar funcionescon memoria. 3. Identifica los efectos de una función en las variables de estado. 4. Define y programa estructuras de datos mutables. 5. Programa funciones que modifican estructuras mutables. 6. Identifica los efectos de una función en estructuras mutables. 7. Reconoce estructuras indexadas como una herramienta de programación. 8. Utiliza estructuras indexadas en la resolución de problemas. 9. Utiliza archivos de texto para guardar/leer información desde la memoria secundaria. 10. Programa funciones que consideran: variables de estado, estructuras de datos mutables, efectos de las funciones en las variables de estado y estructuras mutables y estructuras indexadas, utilizando la “Receta de Diseño” (Felleisen M., 2001). 11. Cumple obligaciones y acuerdos, respetando los compromisos adquiridos en sus actividades académicas. 12. Planifica y presenta sus trabajos, basándose en sus capacidades, sin incurrir en plagio, copia, suplantación de identidad.	
Bibliografía de la unidad		[1] Capítulos 35, 36, 40, 41, 42.	

Número	RA al que tributa	Nombre de la unidad	Duración en semanas
4	RA1, RA2, RA3, RA4	Programación Orientada al Objeto	5 semanas
Contenidos		**Indicador de logro**	
4.1. Conceptos básicos de Programación Orientada al Objeto. 4.2. Definición de clases. 4.3. Interacciones entre objetos.		El/la estudiante: 1. Identifica los conceptos de clase y objeto. 2. Utiliza objetos en la resolución de problemas. 3. Define una clase y sus componentes: campos, constructor, métodos accesores y mutadores. 4. Identifica interacciones entre objetos a través de llamadas de método interna y externa. 5. Programa funciones considerando clases y objetos, utilizando la "Receta de Diseño" (Felleisen M., 2001). 6. Cumple obligaciones y acuerdos, respetando los compromisos adquiridos en sus actividades académicas. 7. Planifica y presenta sus trabajos, basándose en sus capacidades, sin incurrir en plagio, copia, suplantación de identidad.	
Bibliografía de la unidad		[2] Capítulos 1, 2, 3.	

E. Estrategias de enseñanza - aprendizaje:

La metodología de enseñanza y aprendizaje fomenta la participación del estudiante en el aula, las clases son principalmente:

- Clase expositiva, en donde el estudiante identificas las herramientas para la programación y realizan ejercicios en papel.
- Casos de estudio. En cada unidad el estudiante es expuesto a problemas de la vida cotidiana, en donde logra usar las herramientas de programación.

A lo anterior se le suman las tareas que deben ser desarrolladas con el computador, estas son enviadas a través de u-cursos.

F. Estrategias de evaluación:

El curso tiene distintas instancias de evaluación de proceso entre ellas:

- Control 1 Unidad 1
- Control 2 la unidad 1y 2
- Control 3 la unidad 1, 2, 3 y 4
- 60% Examen Unidad 1, 2, 3 y 4 (40%)
- Los controles son controles escritos.
- Tareas: existe un mínimo de 4 tareas (1/3).

G. Recursos bibliográficos:

Bibliografía obligatoria:

[1] Matthias Felleisen, Robert Bruce Findlre, Matthew Flatt, Shriram Krishnamurthi.How to Design Programs: An Introduction to Programming and Computing. The MITPress, 2001.

[2] David Barnes, Michael Kölling. Objects First with Java: A Practical IntroductionUsing BlueJ. Prentice Hall, 2012.

Bibliografía complementaria:

[3] Gutiérrez, F; Peña V.; Quezada M.; Bustos B.; Robbes R. (2019 en revisión) Apuntes CC1002 Introducción a la Programación, Santiago.

H. Datos generales sobre elaboración y vigencia del programa de curso:

Vigencia desde:	2019
Elaborado por:	Benjamín Bustos, Romain Robbes, Eric Tanter
Validado por:	CTD de los Departamentos en reunión de validación
Revisado por:	Área de Gestión Curricular