



Nombre del Cuso: Introducción a la Programación y Computación II			
Código:	771	Créditos:	5
Escuela:	CIENCIAS Y SISTEMAS	Área a la que pertenece:	Programación
Pre requisito:	Introducción a la Programación y Computación I (770) Matemática Intermedia (107) Lógica Matemática (795) Matemática de Computo 1 (960)	Post requisito:	Organización Computacional (964) Estructura de Datos (772) Org. Lenguajes y Compiladores 1 (777)
Categoría:	Obligatorio	Semestre:	1er. Semestre 2025
Docente:	Stanley Barrios Innovacion.usac@gmail.com 233723080010@ingenieria.usac.edu.gt	Auxiliar	Piter Angel Esaú Valiente de León
Edificio:	-	Sección:	D
Salón del curso:	https://meet.google.com/ica-yzfy-cgx	Salón de laboratorio:	https://meet.google.com/jiv-wcgx-uev
Horas por semana del curso:	3	Horas por semana del laboratorio:	1.66
Días que se imparte el curso:	Sábado	Días que se imparte el laboratorio	Sábado
Horario del curso:	Sábado 07:10 - 10:30	Horario del laboratorio:	de 16:20-18:00

1. Descripción del curso

Este curso está diseñado para que el estudiante inicie el proceso de modelado de sistemas de software utilizando los conceptos de la programación orientada a objetos, los ambientes en que estas soluciones son dispuestas, temas orientados a las bases de desarrollo sobre la web, Cloud, así como los temas asociados al manejo de versiones y control del software.



2. Objetivos

General

Lograr que el estudiante expanda sus conocimientos de desarrollo de software usando elementos que le brinden una visión general de los procesos de desarrollo, así como elementos de calidad que le ayuden a robustecer los entregables

Específicos

1. Modelar problemas de forma estándar y profesional
2. Entender metodologías de desarrollo para construir aplicaciones de software aplicando buenas prácticas orientada a la calidad de este.
3. Organizar soluciones en diferentes ambientes en el proceso del desarrollo del software usando arquitecturas claves como stand-alone, cliente servidor y web.

3. Metodología

1. El curso se impartirá a través de clase magistral **virtual** un día por semana, con duración de 4 periodos.
2. El laboratorio se impartirá **de manera virtual** una vez por semana, con duración de dos periodos cada día.
3. Durante el semestre, se asignarán tres proyectos de programación a realizarse de manera individual; así como tareas, ejercicios y pruebas cortas.
4. Las evaluaciones parciales y el examen final serán de manera **presencial**.

4. Competencias terminales

Al finalizar el curso el estudiante desarrolla las siguientes competencias:

- Capacidad para aplicar metodologías de programación y desarrollo de aplicaciones de software.
- Capacidad de aplicar los temas de calidad en el desarrollo de su software bajo el uso de prácticas estándares.
- Conocimiento de los aspectos claves de seguridad y calidad en el desarrollo del software.
- Dominio en el manejo de la memoria dinámica y los TDA's básicos requeridos en el curso de estructura de datos.
- Conocimiento de los ambientes necesarios para desarrollar software y garantizar su buen funcionamiento ante los usuarios finales.
- Conocimiento de la gestión de versiones del software y releases correspondientes.
-



5. Observaciones

1. Es obligatorio acumular el 80% de asistencia antes de cada parcial (de lo contrario no se tendrá derecho a examen).
2. El laboratorio se calificará sobre 100, y será equivalente a 30 puntos de zona.
3. Habrá 3 proyectos de programación.
4. El catedrático revisará las notas obtenidas en el curso y el laboratorio. Podrá decidir si es necesaria una segunda revisión a cada proyecto y considerar nuevamente la ponderación obtenida en cada proyecto.
5. Las notas de cada proyecto serán publicadas por el catedrático del curso en el transcurso del semestre, el estudiante tendrá 8 días como máximo para pedir revisión de proyecto.
6. El laboratorio debe aprobarse con 61 puntos sobre 100.
7. Es obligatorio ganar el laboratorio para tener derecho a evaluación final del curso.
8. No habrá proyecto de retrasada, ni reposición de nota de laboratorio.
9. El curso se aprueba con 61 puntos.



6. Contenido temático del curso	
Unidad	Tema
1. Estructura y manejo de la memoria	<ol style="list-style-type: none"> 1. Modelos de administración de memoria <ol style="list-style-type: none"> 1.1. Administración física de la memoria 1.2. Administración lógica de la memoria 1.3. Gestión de memoria en Python 2. Manejo de Memoria dinámica <ol style="list-style-type: none"> 2.1. Apuntadores <ol style="list-style-type: none"> 2.1.1. Apuntadores en Python 2.2. Tipos de datos abstractos (TDA's) <ol style="list-style-type: none"> 2.2.1. Concepto 2.2.2. Listas simples 2.2.3. Listas doblemente enlazadas 2.2.4. Listas circulares 2.2.5. Pilas 2.2.6. Colas 2.2.7. Listas ortogonales 2.2.8. Listas n-encadenadas 2.2.9. Ordenamientos
2. Principios de Diseño de Software	<ol style="list-style-type: none"> 1. Diseño con Objetos <ol style="list-style-type: none"> 1.1. Frameworks orientados a objetos 1.2. Diseños basados en objetos 1.3. Diseños orientados a objetos 2. Modelos de arquitecturas de despliegue del software <ol style="list-style-type: none"> 2.1. Arquitectura Stand Alone 2.2. Arquitectura Cliente-Servidor 2.3. Arquitectura de N-Capas 2.4. Arquitectura Cloud
3. Arquitecturas para soluciones de Software	<ol style="list-style-type: none"> 1. Aplicaciones de escritorio 2. Aplicaciones Cliente-Servidor 3. Web Development <ol style="list-style-type: none"> 3.1. Conceptos iniciales <ol style="list-style-type: none"> 3.1.1. Introducción al diseño web 3.1.2. Introducción a HTML y CSS <ol style="list-style-type: none"> 3.1.2.1. Principios básicos de HTML 3.1.2.2. Principios básicos de CSS 3.1.3. Entendimiento de servidores Web, Exploradores, HTTP y FTP <ol style="list-style-type: none"> 3.1.3.1. Funcionamiento de los servidores Web 3.1.3.2. Web Servers 3.1.3.3. Web Browsers / Exploradores 3.2. Front End Development <ol style="list-style-type: none"> 3.2.1. Programación del lado del Servidor



	<ul style="list-style-type: none"> 3.2.2. Javascript 3.3. Back End Development <ul style="list-style-type: none"> 3.3.1. Programación del lado del servidor 3.3.2. Introducción a las bases de datos y XML <ul style="list-style-type: none"> 3.3.2.1. Conceptos de bases de datos 3.3.2.2. Tipos de bases de datos 3.3.2.3. XML y su uso en desarrollo web
4. Estrategias de Desarrollo de Software	<ul style="list-style-type: none"> 1. Ambientes de Software Configuración y Despliegue <ul style="list-style-type: none"> 1.1. Tipos de ambientes <ul style="list-style-type: none"> 1.1.1. Desarrollo 1.1.2. Quality Assurance (QA) 1.1.3. Unit Acceptance Testing (UAT) 1.1.4. Producción 1.2. Despliegue entre ambientes 1.3. Consideraciones, aprobaciones y aseguramiento 2. Metodologías de desarrollo <ul style="list-style-type: none"> 2.1. Modelo cascada 2.2. Scrum <ul style="list-style-type: none"> 2.2.1. Conceptos básicos 2.2.2. Valores 2.2.3. Equipo 2.2.4. Eventos 3. Principios básicos de versionamiento <ul style="list-style-type: none"> 3.1. Metodologías de numeración de versiones 3.2. Criterio para modificación de versiones 3.3. Versionado de productos complejos 3.4. Clasificación para versiones no estables 3.5. Versionado de parches 3.6. Sistema de versionado de código centralizado
5. Security & Quality Assurance	<ul style="list-style-type: none"> 1. Security OWASp Checklist <ul style="list-style-type: none"> 1.1. Data protection 1.2. File management 1.3. Memory management 2. Quality <ul style="list-style-type: none"> 2.1. Desarrollo y plan de calidad 2.2. Estrategias de pruebas de software <ul style="list-style-type: none"> 2.2.1. White box, black box testing 2.3. Diseño de casos de pruebas 2.4. Pruebas de regresión 2.5. Pruebas de rendimiento 2.6. Pruebas de stress 2.7. Pruebas Alpha y Beta



7. Evaluación del rendimiento académico

Según el Reglamento General de Evaluación y Promoción del Estudiante de la Universidad de San Carlos de Guatemala, la zona tiene valor de 75 puntos, la nota mínima de promoción es de 61 puntos y la zona mínima para optar a examen final es de 36 puntos.

Procedimiento de evaluación		Ponderación
Clase	Ejercicios	04 pts.
	Prácticas	03 pts.
	Primer parcial	12 pts.
	Segundo parcial	13 pts.
	Tercer parcial	13 pts.
Total de clase		45 pts.
Laboratorio	Proyecto I	10 pts.
	Proyecto II	10 pts.
	Proyecto III	10 pts.
Total de laboratorio		30 pts.
Zona		75 pts.
Examen Final		25 pts.
Nota de promoción		100 pts.

8. Cronograma de actividades

Tema principal	Contenido a desarrollar	Fecha
Modelos de administración de memoria	Administración física de la memoria Administración lógica de la memoria Gestión de la memoria en Python	25-enero
Memoria dinámica	Apuntadores Apuntadores en Python	1-feb
	Tipos de datos abstractos (TDA's) - Concepto	1-feb
	- Listas simples	8 - feb
	- Listas doblemente enlazadas - Listas circulares	8 - feb
	- Pilas - Colas	15 – feb
	- Listas ortogonales - Listas n-encadenadas	15 - feb
Principios de diseño de software	- Ordenamientos Diseño con objetos	
Primer parcial	15 – febrero	
Principios de diseño de software	Diseño con Objetos	
	- Frameworks orientados a objetos - Diseños basados en objetos	22 - feb 22 - feb



	<ul style="list-style-type: none"> - Diseños orientados a objetos Modelos de arquitecturas de despliegue del software - Arquitectura Stand Alone - Arquitectura Cliente-Servidor - Arquitectura de N-Capas - Arquitectura Cloud 	1 marzo
		8 marzo
Arquitectura para aplicaciones WEB	Aplicaciones de escritorio	15 - mar
	Aplicaciones Cliente-Servidor	15 - mar
	Web Development	
	<ul style="list-style-type: none"> - Conceptos iniciales - Introducción al diseño web - Introducción a HTML y CSS - Entendimiento de servidores Web, 	
	Exploradores, HTTP y FTP	22 - mar
	<ul style="list-style-type: none"> - Front End Development - Programación del lado del Servidor - Back End Development - Programación del lado del servidor - Introducción a las bases de datos, XML y JSON 	22 - mar
	<ul style="list-style-type: none"> - Conceptos de bases de datos - Tipos de bases de datos - XML y su uso en desarrollo web - Conceptos básico de JSON 	29 - mar
		29 - mar
Segundo Parcial	29 - marzo	
Estrategias de desarrollo de software	Ambientes de Software Configuración y Despliegue	5 - abril
	<ul style="list-style-type: none"> - Tipos de ambientes - Despliegue entre ambientes - Consideraciones, aprobaciones y aseguramiento 	
	Metodologías de desarrollo	12 - abril
	<ul style="list-style-type: none"> - Modelo cascada - Scrum 	
	Principios básicos de versionamiento	
Tercer parcial	26 - abril	
Security & Quality Assurance II	Security OWASp Checklist	26- abril
	<ul style="list-style-type: none"> Quality - Diseño de casos de pruebas - Pruebas de regresión - Pruebas de rendimiento - Pruebas de stress - Pruebas Alpha y Beta Consulta de dudas 	3 de mayo
Último día de clases	3 de mayo	



Período de exámenes finales

Según programación de Junta Directiva

9. Bibliografía

- Python para informáticos Versión 2.7.2 Charles Severance
- Fundamentos de programación, Algoritmos, Estructuras de Datos y Objetos, Luis Joyanes Aguilar, Cuarta Edición, McGraw-Hill
- The Scrum Guide™ Ken Schwaber and Jeff Sutherland. 2017
- Craig Larman; Introducción al análisis y diseño orientado a objetos. Prentice Hall
- Software Quality Assurance, From Theory to Implementation, Daniel Galin 2004.
- State of Software Development, CodingSans 2019
- El control de versiones, Guillem Borrell, 2006.
- Software Design, David Budgen, 2ª. Edición, Pearson Addison Wesley.
- OWASP Application Security Verification Standard 4.0, 2019

10. Normas para la clase virtual

- Todas las Comunicaciones con el profesor y los auxiliares deben ser por los correos electrónicos que se indiquen en clase.
- En toda comunicación escrita se debe mostrar respeto y no utilizar mensajes en mayúsculas.
- Las comunicaciones enviadas por correo electrónico serán atendidas en un máximo de 3 días hábiles.
- Durante las clases los estudiantes deben encender su cámara siempre que el profesor o el auxiliar les hagan una pregunta directa, o bien, cuando el estudiante realice alguna consulta.
- Durante las clases los estudiantes pueden hacer consultas por el chat del curso, según lo indique el profesor, teniendo el cuidado de ser respetuoso y mantener las reglas de cortesía durante la escritura.