UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERIA
ESCUELA DE CIENCIAS

PROGRAMA DEL CURSO
NOMBRE DEL CURSO: Introducción a la Programación y Computación 1

CODIGO:	**0770**	**CREDITOS:**	**4**
ESCUELA:	**Ciencias y Sistemas**	**AREA A LA QUE PERTENECE:**	**Desarrollo de Software**
PRE REQUISITO:	**33 créditos y 0103 Matemática Básica 2**	**POST REQUISITO:**	**0771 Introducción a la Programación y Computación 2 0796 Lenguajes Formales y de Programación.**
CATEGORIA:	**Obligatorio**	**VIGENCIA:**	**Primer Semestre 2020**
CATEDRÁTICO (A):	**Ver anexo**	**AUXILIAR:**	**Staff**
EDIFICIO:	**T-3 y T-7**	**SECCIÓN:**	**A, B, C, D y E**
SALÓN DEL CURSO:	**Ver anexo**	**SALON DEL LABORATORIO:**	**Pendiente**
HORAS POR SEMANA DEL CURSO:	**4**	**HORAS POR SEMANA DEL LABORATORIO:**	**2**
DÍAS QUE SE IMPARTE EL CURSO:	**Martes y Jueves**	**DIAS QUE SE IMPARTE EL LABORATORIO:**	**Pendiente**
HORARIO DEL CURSO:	**7:10 – 8:50**	**HORARIO DEL LABORATORIO:**	**Pendiente**

DESCRIPCIÓN DEL CURSO:

El curso es el acercamiento inicial del estudiante de la carrera de sistemas, a la programación mediante el uso de disciplinas y metodologías especializadas. El curso se fundamenta en el concepto de algoritmo para la resolución de problemas de programación, enfatizando el uso del paradigma orientado a objetos. Se introducen conceptos básicos de UML como guía para el diseño de sistemas orientados a objetos. Se acerca al estudiante al conocimiento de los principales algoritmos de búsquedas y ordenamientos. Se cubre una parte importante de las estructuras de datos, los tipos de datos abstractos. Asimismo, el estudiante conocerá el lenguaje Java como el lenguaje oficial de programación del curso.

OBJETIVOS:

General

- Lograr que el estudiante adquiera la habilidad de programar independientemente de la plataforma y los conocimientos básicos de la programación utilizando el paradigma orientado a objetos.

Específico

1. Integrar al estudiante a la tecnología de la computación.
2. Conocer las diferentes metodologías de programación.
3. Adquirir la habilidad de hacer algoritmos.
4. Analizar los problemas con metodología orientada a objetos.
5. Aprender a elaborar diseños de clases preliminares en UML.
6. Organizar soluciones utilizando un lenguaje de programación.
7. Conocer el lenguaje Java como el primer lenguaje de programación para computadoras para una arquitectura stand-alone.

METODOLOGÍA:

- Clases diarias.
- Elaboración de investigaciones y tareas.
- Práctica de exámenes cortos y parciales.
- Laboratorio taller.

Elaboración de proyectos de programación.

EVALUACION DEL RENDIMIENTO ACADÉMICO:

Clase teórica (70 puntos)		**Clase práctica (30 puntos)**	
Descripción	**Pts.**	**Descripción**	**Pts.**
Tareas, Cortos y Asistencia	5	Tareas	10
Primer parcial	13	Prácticas	20
Segundo parcial	13	Proyectos	40
Tercer parcial	14	Exámenes cortos	20
Laboratorio	30		
Zona total	75	Zona total	90
Examen Final	25	Examen Final	10
Total	100	Total	100

El curso se gana con 61 pts. de 100. Y el laboratorio de gana con 61 pts. de 100.

CONTENIDO:

1. Introducción
   - 1.1. Conceptos computacionales
     - 1.1.1. Computadora
     - 1.1.2. Hardware
     - 1.1.3. Firmware
     - 1.1.4. Software
   - 1.2. Organización
     - 1.2.1. CPU
     - 1.2.2. Memoria principal
     - 1.2.3. Memoria secundaria
     - 1.2.4. Dispositivos E/S
     - 1.2.5. Periféricos
   - 1.3. Lenguajes de programación
     - 1.3.1. Lenguaje de máquina
     - 1.3.2. Lenguajes de bajo nivel
     - 1.3.3. Lenguajes de alto nivel
   - 1.4. Resolución de problemas computacionales
     - 1.4.1. Análisis del problema
     - 1.4.2. Diseño del algoritmo
     - 1.4.3. Codificación
     - 1.4.4. Compilación y ejecución
     - 1.4.5. Verificación y depuración
     - 1.4.6. Documentación
2. Programación modular y estructuras básicas
   - 2.1. Secuencial y procedural: metodología Top-Down.
   - 2.2. Variables: concepto, manipulación y asignación.
   - 2.3. Tipos de datos (primitivos y construidos por el usuario)
   - 2.4. Operadores aritméticos
   - 2.5. Operadores relacionales y lógicos
   - 2.6. Estructuras de control condicionales
     - 2.6.1. Si – Sino (if – else)

2.6.2. En caso (switch / case)
2.7. Estructuras cíclicas (bucles, loops)
2.7.1. Para (for)
2.7.2. Mientras (while)
2.7.3. Repetir - Hasta (Repeat – Until / do-while)
2.8. Las rutinas
2.8.1. Procedimiento y función
2.8.2. Entorno de las variables (alcance o ámbito)
2.8.3. Los parámetros
3.8.3.1 Por referencia
3.8.3.2 Por valor
2.8.4. El valor de retorno
2.9. Modularidad
2.9.1. Segmentos por rutina
2.9.2. Uso adecuado de prefijos
2.9.3. Documentación interna
2.9.4. Legibilidad y entendimiento
2.10. Recursividad

3. Metodología Orientada a Objetos
3.1. Concepto de abstracción y clasificación
3.2. Clases y objetos
3.3. Mensajes y métodos
3.4. El principio el encapsulamiento
3.5. Los miembros de una clase
3.5.1. Atributos
3.5.2. Métodos (operaciones)
3.5.3. Constructores y destructores
3.6. Modificadores de visibilidad
3.6.1. Privado
3.6.2. Público
3.6.3. Protegido
3.7. Relaciones entre clases y objetos
3.7.1. Asociación
3.7.2. Agregación y composición
3.7.3. Herencia (simple y múltiple)
3.8. Polimorfismo
3.8.1. Sobrecarga de métodos
3.8.2. Virtualización
3.9. Construcciones abstractas
3.9.1. Clase abstracta
3.9.2. Interface
3.10. Conceptos avanzados
3.10.1. Miembros estáticos (static) y miembros de instancia
3.10.2. Referencia “this”
3.10.3. Clases paramétricas (plantilla de clases).
3.11. Principios básicos de UML (diagrama de clases)
3.11.1. Definición de clases y sus relaciones
3.11.2. Ámbito de las propiedades, Métodos
3.11.3. Diseño de programas
3.11.4. Asociaciones y restricciones, clases de asociaciones, Multiplicidad, Dependencia
3.11.5. Relaciones múltiples (asociativas) y reflexivas

4. Estructuras algorítmicas
4.1. Arreglos vectoriales de datos
4.1.1. Conceptos: elementos, longitud, indexación, representación en memoria.
4.1.2. Arreglos bidimensionales (matrices): representación en memoria.

- 4.1.3. Arreglos n-dimensionales (multidimensionales).
- 4.2. Las cadenas de caracteres
  - 4.2.1. Concepto: diferencia con arreglos de caracteres.
  - 4.2.2. Cadenas estáticas (ej: String) y dinámicas (ej: StringBuffer).
  - 4.2.3. Operaciones y métodos.
- 4.3. Búsqueda de datos en arreglos
  - 4.3.1. Secuencial
  - 4.3.2. Binaria
- 4.4. Ordenamiento de datos en arreglos
  - 4.4.1. Burbuja
  - 4.4.2. Por inserción
  - 4.4.3. Por selección
  - 4.4.4. Quick Sort
- 4.5. La pila (Stack)
  - 4.5.1. Política de acceso a datos (LIFO) y operaciones.
- 4.6. La cola (Queue)
  - 4.6.1. Política de acceso a datos (FIFO) y operaciones.
  - 4.6.2. Representaciones: simple y circular.
- 4.7. El uso de Heap
  - 4.7.1. Asociación a la pila
  - 4.7.2. Tomar y devolver al heap
  - 4.7.3. Usos con las pilas y las colas

5. Colecciones de Datos y tipos de Datos Abstractos
   - 5.1. Los índices y el apuntador simple
   - 5.2. El apuntador subíndice
   - 5.3. Almacenamiento
   - 5.4. Ordenamiento
   - 5.5. Los registros
   - 5.6. Concepto y definición por campos
   - 5.7. Tipos de apuntadores (estáticos y dinámicos)
   - 5.8. Listas simples
   - 5.9. Listas doblemente encadenadas
   - 5.10. Pilas usando listas
   - 5.11. Colas usando listas
   - 5.12. Listas ortogonales
   - 5.13. Listas n-encadenadas
6. Flujos de bytes y manipulación de archivos
   - 6.1. Concepto: modelo productor-consumidor y flujo (stream).
   - 6.2. Tipos de archivos
   - 6.3. Archivos de texto
   - 6.4. Operaciones básicas
     - 6.4.1. Abrir y cerrar
     - 6.4.2. Lectura, escritura y posicionamiento
     - 6.4.3. Localización del final del archivo
7. Cloud Computing (Investigación requisito para examen final)
   - 7.1. Antecedentes e Historia
   - 7.2. Definición
   - 7.3. Funcionamiento
   - 7.4. Tipos de nube
   - 7.5. Beneficios de la nube
   - 7.6. Servicios de la nube
     - 7.6.1. Modelo SaaS
     - 7.6.2. Modelo PaaS
     - 7.6.3. Modelo IaaS
   - 7.7. Aplicaciones prácticas
   - 7.8. Empresas relacionadas

CLÁUSULAS RESTRICTIVAS:

El perfil del estudiante de la facultad de Ingeniería de la Universidad de San Carlos de Guatemala exige una alta calidad en la excelencia académica y ética profesional. Se establecen en este curso los siguientes lineamientos que regulan el comportamiento del estudiante:

- Copias en exámenes, cortos, proyectos, tareas e investigaciones tienen cero de nota.
- Exámenes parciales y examen final NO tienen reposición.
- No hay prorrogas.
- No hay reposición de proyectos.
- Cualquier proyecto, tarea o investigación que se entregue después de la fecha calendarizada tiene 30 puntos menos, cada día de atraso.
- Los exámenes resueltos a lápiz no tienen derecho a revisión.
- Es obligatorio ganar el laboratorio para tener derecho a evaluación total del curso.
- Para poder optar a la revisión de la zona final es obligatorio haber asistido a los exámenes parciales y al examen final.

BIBLIOGRAFÍA:

- JOYANES, L. y ZAHONERO, I. "**Programación en Java 2 (algoritmos, estructura de datos y programación orientada a objetos)**". España, McGraw-Hill / Interamericana de España, S. A. 2002, PP 725
- JOYANES, L. "**Programación en Tubo Pascal Versiones 5.5, 6.0, y 7.0**", (2da Edición), México, McGraw-Hill / Interamericana de España, S. A. 1995, PP. 914
- Deitel & Deitel. "**Cómo Programar en Java**" (7ma Edición), México, Prentice Hall 2008, PP. 1280
- McLaughlin, B.; Pollice, G. y West, D. "**Head First Object-Oriented Analysis & Design**", EUA, O'Reilly Media 2006, PP. 636
- Freeman, E.; Robson, E.; Bates, B. y Sierra, K. "**Head First Design Patterns**", EUA, O'Reilly Media 2004, PP. 694
- Manuales de Referencia de Java, <http://www.sun.com/java>.
- Cualquier otro material (escrito o digital) entregado en clase.