

| |
|---|
| PROGRAMA DEL CURSO |
| NOMBRE DEL CURSO: INTRODUCCIÓN A LA PROGRAMACIÓN Y COMPUTACIÓN 1 |

| | | | |
|--------------------------------------|--|--|---|
| CÓDIGO: | 0770 | CRÉDITOS: | 4 |
| ESCUELA: | CIENCIAS Y SISTEMAS | ÁREA 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. |
| CATEGORÍA: | OBLIGATORIO | VIGENCIA: | PRIMER SEMESTRE |
| CATEDRÁTICO (A): | VER ANEXO | AUXILIAR: | STAFF |
| EDIFICIO: | T3 | SECCIÓN: | A, B, C, D, E, F |
| SALÓN DEL CURSO: | 309, 310, 314,316, 312 | SALON DEL LABORATORIO: | PENDIENTE |
| HORAS POR SEMANA DEL CURSO: | 4 | HORAS POR SEMANA DEL LABORATORIO: | |
| DÍAS QUE SE IMPARTE EL CURSO: | MARTES Y JUEVES | DÍAS QUE SE IMPARTE EL LABORATORIO: | PENDIENTE |
| HORARIO DEL CURSO: | 7:10 - 8:50 AM | HORARIO DEL LABORATORIO: | PENDIENTE |

| |
|---|
| DESCRIPCIÓN DEL CURSO: |
| <p>El curso busca ser el acercamiento inicial del estudiante de la carrera de sistemas, al mundo de Desarrollo de Software mediante el uso de métodos, técnicas y metodologías especializadas. Se fundamenta en el concepto de algoritmo para la resolución de problemas de programación utilizando computadoras, enfatizando el uso del paradigma de Programación Orientado a Objetos. Se acerca al estudiante al conocimiento de los principales algoritmos de búsquedas y ordenamientos. Asimismo, el estudiante conocerá el lenguaje Java como el lenguaje oficial de programación del curso.</p> |

| |
|--|
| OBJETIVOS: |
| <p>General</p> <ul style="list-style-type: none"> Adquirir, por parte del estudiante, la destreza de analizar, diseñar y codificar software de alta calidad independientemente de la plataforma y lenguaje de programación fundamentado en los conocimientos básicos de la programación utilizando el Paradigma Orientado a Objetos. <p>Específico</p> <ol style="list-style-type: none"> Integrar al estudiante a la tecnología de la computación. Conocer las diferentes metodologías de software. Analizar los problemas bajo la perspectiva de Programación Orientada a Objetos. Diseñar soluciones elegantes basadas en el entendimiento de proceso de Análisis Organizar soluciones utilizando un lenguaje de programación oficial y complementario. |

METODOLOGÍA:

- Clases Presenciales, martes y jueves con apoyo de la plataforma UEDI.
- Elaboración de investigaciones y tareas.
- Práctica de exámenes cortos y parciales.
- Laboratorio y talleres.
- Elaboración de proyectos de programación
- Elaboración de prácticas cortas de programación

EVALUACIÓN DEL RENDIMIENTO ACADÉMICO:

| Clase teórica (70 puntos) | | Clase práctica (30 puntos) | |
|--|------|----------------------------|------|
| Descripción | Pts. | Descripción | Pts. |
| Tareas, Cortos, Asistencia y participación | 5 | Tareas | 10 |
| Primer parcial | 12 | Prácticas | 20 |
| Segundo parcial | 13 | Proyectos | 40 |
| Tercer parcial | 15 | 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. El laboratorio se gana con 61 pts. de 100.

CONTENIDO:

1. Algoritmos, Pseudocódigo y Diagramas de Flujos

- 1.1. Definición de Algoritmos
 - 1.1.1. Conceptos de algoritmo
 - 1.1.2. Análisis y comprensión de un problema
 - 1.1.3. Programas y paradigmas de programación y lenguajes
 - 1.1.4. Transformación de un programa
- 1.2. Nociones básicas: variables, tipos y expresiones
- 1.3. Estructura general del pseudocódigo
- 1.4. Estructuras componentes del Pseudocódigo
- 1.5. Uso de arreglos
- 1.6. Funciones y procedimientos
- 1.7. Ciclos
- 1.8. Diagramas de Flujos

2. Conceptos Computacionales

- 2.1. Concepto de Computadora
- 2.2. Arquitectura/Organización física del Computadora
 - 2.2.1. Dispositivos E/S , Memoria principal, Procesador
- 2.3. Unidades de medida de memoria
- 2.4. Sugerencia de arquitectura del computadora para programar
- 2.5. Representación de la información en las computadoras
 - 2.5.1. Representación de textos
 - 2.5.2. Representación de valores numéricos
 - 2.5.3. Representación de imágenes
 - 2.5.4. Representación de sonidos
- 2.6. Codificación de la información
 - 2.6.1. Decimal, Binario, Octal Hexadecimal
- 2.7. Ciclo Clásico de vida del SW

3. Fundamentos de Programación

- 3.1. Paradigmas de Programación
- 3.2. Lenguajes de desarrollo y evolución generacional
- 3.3. Elementos del lenguaje
 - 3.3.1. Identificadores, Comentarios, tipos de datos, Constantes
 - 3.3.2. Operadores, Prioridad de Operadores
 - 3.3.3. Palabras reservadas
 - 3.3.4. Bibliotecas de funciones
- 3.4. Datos Nativos, Condiciones, Ciclos y Procedimientos, Funciones
- 3.5. Recursividad
- 3.6. Manipulación de Vectores, Cadenas
 - 3.6.1. Conceptos
 - 3.6.2. Cadenas de caracteres
 - 3.6.3. Operaciones
 - 3.6.3.1. Búsqueda Secuencial y Binaria
 - 3.6.3.2. Ordenamiento
 - 3.6.3.2.1. Burbuja
 - 3.6.3.2.2. Inserción
 - 3.6.3.2.3. Selección
 - 3.6.3.2.4. Quick Sort y Shell Sort
- 3.7. Manejo de Archivos
 - 3.7.1. Jerarquía de datos
 - 3.7.2. Streams
 - 3.7.3. Tipos de Archivos
 - 3.7.4. Operaciones sobre archivos
 - 3.7.4.1. Creación
 - 3.7.4.2. Consulta
 - 3.7.4.3. Actualización
- 3.8. Debugging
 - 3.8.1. Principios de Debbug
 - 3.8.1.1. Principio de Confirmación
 - 3.8.1.2. Start Small
 - 3.8.1.3. Enfoque Top Down
 - 3.8.1.4. Ubicación de la falla del segmento
 - 3.8.1.5. Determinando el loop infinito
 - 3.8.1.6. Búsqueda Binaria
 - 3.8.2. Operaciones Principales
 - 3.8.2.1. Breakpoints
 - 3.8.2.2. Single-Stepping
 - 3.8.2.3. Resume Operation
 - 3.8.2.4. Temporary Breakpoints

4. Programación Orientada a Objetos (POO)

- 4.1. Tipos de Datos Abstractos (Clases)

- 4.2. Modelado e identificación de Objetos
- 4.3. Propiedades POO
 - 4.3.1. Abstracción
 - 4.3.2. Encapsulamiento
 - 4.3.3. Herencia
 - 4.3.4. Polimorfismo
- 4.4. Declaración/Especificación de una Clase
- 4.5. Acceso a miembros de una clase
- 4.6. Declaración de métodos
- 4.7. Constructores/Destructores
- 4.8. Clases Compuestas
- 4.9. Garbage Collector/Recolector de basura
- 4.10. Relaciones entre clases y dependencias
 - 4.10.1. Asociación
 - 4.10.1.1. Multiplicidad
 - 4.10.1.2. Restricciones
 - 4.10.2. Agregación
 - 4.10.3. Generalización y Especialización
 - 4.10.4. Herencia de clases derivadas
 - 4.10.5. Simple y Múltiple
 - 4.10.6. Accesibilidad en Herencia
- 4.11. Interfaces

5. Testing, Security & Quality Assurance I

- 5.1. Seguridad en el código
 - 5.1.1. Principios comunes de seguridad el código
 - 5.1.1.1. Estableciendo estándares de convención de código
 - 5.1.1.2. Uso de funciones seguras
 - 5.1.1.3. Uso de herramientas de inspección de código
 - 5.1.1.4. Aseguramiento en el manejo de los datos
 - 5.1.1.5. Manejo de errores
- 5.2. Introducción al QA
 - 5.2.1. Modelos de calidad del Software
 - 5.2.2. Estructura y enfoque de los modelos de calidad del Software
 - 5.2.2.1. Calidad a nivel de proceso
 - 5.2.2.1.1. ITIL
 - 5.2.2.1.2. ISO/IEC 15504
 - 5.2.2.1.3. Bootstrap
 - 5.2.2.1.4. Dromey
 - 5.2.2.1.5. PSP
 - 5.2.2.1.6. TSP
 - 5.2.2.1.7. IEEE / EIA 2207
 - 5.2.2.1.8. Cobit 4.0
 - 5.2.2.1.9. ISO 9003
 - 5.2.2.1.10. CMMI
 - 5.2.2.1.11. ISO/IEC 2000
 - 5.2.2.1.12.
 - 5.2.2.2. Calidad a nivel de producto
 - 5.2.2.2.1. Mc Call
 - 5.2.2.2.2. Bohem
 - 5.2.2.2.3. Furps
 - 5.2.2.2.4. Gilb
 - 5.2.2.2.5. ISO 9126
 - 5.2.2.2.6. SQAЕ
 - 5.2.2.2.7. WebQEM
 - 5.2.2.3. Calidad a nivel de uso

6. Introducción a Cloud Computing

- 6.1. Visión y Definición de Cloud Computing

- 6.2. Modelo de Referencia Cloud
 - 6.2.1. Infraestructura como un Servicio (IAAS)
 - 6.2.2. Software como un Servicio (SAAS)
 - 6.2.3. Plataforma como un Servicio (PAAS)
- 6.3. Proveedores Cloud
 - 6.3.1. Principales competidores según Gartner
 - 6.3.2. Servicios claves
- 6.4. Características y Beneficios
- 6.5. Evolución del desarrollo
 - 6.5.1. Mainframes
 - 6.5.2. Sistemas Distribuidos
 - 6.5.3. Virtualización
 - 6.5.4. Computación Orientada a Servicios
 - 6.5.5. Computación Orientada a la Utilidad

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 prórrogas.
- No hay reposición de proyectos.
- 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 Turbo 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
- Michaela Juganaru Mathieu, Introducción a la programación
- David Evans, Introduction to computing
- Jesus Fernandez-Pablo Guerron, David Zarruck, University of Pensilvania
- Matloff Norman-Jay Peter, The art of Debugging
- OWASP Secure Coding Practices, Quick Reference Guide
- Media 2004, PP. 694
- Manuales de Referencia de Java, <<http://www.sun.com/java>>.
- Cualquier otro material (escrito o digital) entregado en clase.