OPMC-3 PROGRAMACIÓN AVANZADA

NOMBRE DE LA ASIGNATURA:

PROGRAMACIÓN AVANZADA

OBJETIVO GENERAL Y PARTICULARES DE LA ASIGNATURA:

General:

Presentar al estudiante los conceptos y procedimientos de aplicación de diversos paradigmas de programación (orientada a objetos, paralela, reactiva) aplicando procedimientos algorítmicos avanzados y estructuras de datos para diseñar soluciones a problemas, analizando la idoneidad y complejidad de los algoritmos propuestos.

Particulares:

1. Analizar, diseñar, construir y mantener aplicaciones de forma robusta, segura y eficiente, eligiendo el paradigma y los lenguajes de programación más adecuados.
2. Conocer, diseñar y utilizar de forma eficiente los tipos y estructuras de datos más adecuados a la resolución de un problema.
3. Diseñar y evaluar interfaces gráficas que garanticen la accesibilidad y usabilidad a los sistemas, servicios y aplicaciones informáticas.

Duración del ciclo: 16 sesiones	**Horas totales con docente**: 64 hrs	**Horas totales independientes**: 32 hrs	**Instalaciones**: Sala de computo
CICLO, ÁREA O MÓDULO: OPTATIVAS		**CRÉDITOS**: 6	**CLAVE**: **OPMC-3**

TEMAS Y SUBTEMAS:

1. Paradigmas de programación
   - 1.1. Programación Orientada a Objetos
   - 1.2. Programación Paralela
   - 1.3. Programación Reactiva
   - 1.4. Lenguajes de programación (*C/C++, Python, React*)
   - 1.5. Modelación (*Unified Modeling Language* - UML)
2. Estructuras de datos
   - 2.1. Arreglos (vectores, matrices, *cells*)
   - 2.2. Diccionarios, listas, tuplas y secuencias
   - 2.3. Algoritmos de búsqueda
   - 2.4. Algoritmos de ordenación
   - 2.5. Algoritmos recursivos
   - 2.6. Manejo de excepciones (*try/catch/finally*)
3. Diseño de interfaces visuales
   - 3.1. Diseño centrado en el usuario
   - 3.2. Herramientas de diseño de interfaces (*front-end*)
   - *3.3. Socket's*
   - 3.4. Evaluación de la usabilidad (*affordance)*
4. Bases de datos
   - 4.1. Diseño de bases de datos
   - 4.2. Base de datos Relacionales
   - 4.3. Bases de datos No relacionales
   - 4.4. Normalización de bases de datos
   - 4.5. Lenguajes (SQL ,noSQL)

RESULTADOS DEL APRENDIZAJE:

- El estudiante:
  - Entiende el concepto de paradigma y sus implicaciones en el modo de resolver problemas.
  - Diseña algoritmos eficientes con baja complejidad utilizando de manera eficaz estructuras de datos.
  - Construye código robusto utilizando mecanismos de manejo de excepciones.
  - Desarrolla interfaces gráficas user-friendly
  - Conoce y aplica el modelo de diseño de bases de datos de acuerdo al tipo de problemática que debe resolver.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DEL DOCENTE:

- Ejercicios: Práctica en problemas concretos vinculados con la temática de la materia.

- Prácticas de laboratorio: Realización de ejercicios, simulaciones o experimentos para el adiestramiento y adquisición de habilidades y competencias, así como su evaluación. Dichas actividades estarán basadas en la temática.
- Aprendizaje colaborativo: método educativo mediante el cual se busca unir los esfuerzos de los alumnos o de alumnos y profesores para, así trabajar juntos en la tarea de adquirir conocimiento, habilidades y competencias.
- Solución de problemas: Dinámica participativa en la que el profesor expone a los integrantes del grupo una situación problemática a resolver a partir de criterios definidos por el profesor.

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTE:

- Lecturas dirigidas: Análisis crítico de lecturas relacionadas con la temática del curso para su posterior discusión y exposición de conclusiones.
- Investigación y análisis documental: Acopio de información por parte del estudiante a través de la consulta, lectura, análisis y discusión de material escrito y electrónico de manera que le permita establecer nuevas relaciones con los contenidos de la clase y formular conclusiones.
- Ejercicios: Práctica en problemas concretos vinculados con la temática de la materia.
- Proyectos: Elaboración de propuestas de desarrollo y de solución de problemas. Los proyectos deberán ser guiados por el proceso de investigación alrededor de un tema propuesto por el alumno o el profesor, pero alineado al objetivo general de la materia.

MEDIOS Y CRITERIOS DE EVALUACIÓN:

- Evaluación parcial (40 %) de la calificación final del curso.
- Evaluación final (60%) de la calificación final del curso.

La evaluación parcial y final estarán compuesta del siguiente medio de evaluación:

- Tareas/ejercicios: 20%
- Prácticas: 20%
- Exposiciones: 10%
- Proyecto final (avances para evaluación parcial): 50 %

RECURSOS Y MATERIALES

- Software (IDE's, librerías, etc) de propósito específico para la materia.
- Computadora con acceso a internet.
- Libros
- Acervo bibliográfico científico y documental
- Pizarrón
- Diapositivas

BIBLIOGRAFÍA

1. Stroustrup, B. (2013). The C++ programming language. Pearson Education.
2. Jaworski, M., & Ziadé, T. (2019). Expert Python Programming: Become a master in Python by learning coding best practices and advanced programming concepts in Python 3.7. Packt Publishing Ltd.
3. Wengrow, J. (2020). A Common-Sense Guide to Data Structures and Algorithms. Pragmatic Bookshelf.
4. Miller, B., & Ranum, D. (2013). Problem solving with algorithms and data structures. URL: https://www. cs. auckland. ac. nz/.../Problem Solving with Algorithms and Data Structures. pdf (Last accessed: 30.03. 2018).
5. Gordon, Z. (2020). React Explained: Your Step-by-Step Guide to React. OS Training.
6. Lazar, G., & Penea, R. (2018). Mastering Qt 5: Create stunning cross-platform applications using C++ with Qt Widgets and QML with Qt Quick. Packt Publishing Ltd.

REQUISITOS ACADÉMICOS DEL PERSONAL DOCENTE

Preferentemente académico con grado de Doctor en Ciencias, con perfil en tecnologías de la información y telecomunicaciones o en área afín. Experiencia docente mínima de dos años.