UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA	DES:	INGENIERÍA
	Programa académico	Ingeniero en Computación
	Tipo de materia (Obli/Opta):	Optativa
	Clave de la materia:	OPCO804
	Semestre:	Octavo
	Área en plan de estudios:	Específica
	Total de horas por semana:	6
	Teoría: Presencial o Virtual	0
	Laboratorio o Taller:	4
	Prácticas:	0
	Trabajo extra-clase:	2
UNIDAD ACADÉMICA: **FACULTAD DE INGENIERÍA**	***Créditos Totales:***	6
PROGRAMA ANALÍTICO DE	**Total de horas semestre (x sem):**	96
LA UNIDAD DE APRENDIZAJE:	Fecha de actualización:	Octubre 2024
PRUEBAS DE SOFTWARE	*Prerrequisito (s):*	*N/A*

DESCRIPCIÓN:

Introducción al ámbito de las pruebas de software, desde los fundamentos básicos hasta las aplicaciones prácticas en proyectos reales. Comprender los conceptos fundamentales, técnicas de pruebas, herramientas, estrategias y problemas comunes para su adecuada aplicación. Además de adquirir habilidades para planificar, diseñar, ejecutar y automatizar pruebas, puede garantizar la calidad del software en todas las etapas del ciclo de vida del desarrollo de software.

COMPETENCIAS PARA DESARROLLAR:

B4. Transformación Digital

Transforma la cultura digital en la sociedad, en las organizaciones e instituciones educativas para aprovechar al máximo el potencial de las tecnologías y herramientas digitales; propiciar su uso responsable y ético que estimule la creatividad, innovación, la comunicación efectiva y el trabajo colaborativo e interdisciplinar en la solución de problemas de la sociedad digital; promoviendo la privacidad y la seguridad, así como el respeto a los derechos de autor y la propiedad intelectual.

COMPETENCIAS PROFESIONALES.

P2. DESARROLLO DE PROYECTOS DE INGENIERÍA

Desarrolla proyectos de ingeniería complejos en sus etapas de planeación, análisis y diseño, utilizando las tecnologías y los principios de la administración para la optimización de los recursos con base en procesos de calidad, mejora continua y teniendo en cuenta la seguridad, el costo del ciclo de vida, el carbono neto cero y la salud según sea necesario, atendiendo las necesidades de sostenibilidad.

Identifica las principales áreas de oportunidad en proyectos complejos de ingeniería para definir estrategias de solución utilizando herramientas tecnológicas y administrativas, para optimizar los procesos de calidad, mejora continua contemplando las normatividades aplicables.

Desarrolla proyectos complejos de ingeniería que integra la planeación, análisis, diseño y administración con base en los criterios de sostenibilidad.

Identifica los principales factores involucrados en la solución de problemas de ingeniería para desarrollar propuestas utilizando herramientas de ciencias básicas e ingeniería aplicada.

Selecciona configuraciones óptimas de los recursos involucrados en proyectos de ingeniería utilizando como base procesos de calidad y mejora continua.

COMPETENCIAS ESPECÍFICAS.

E1. DISEÑO Y DESARROLLO DE SOFTWARE

Utilizar en el diseño y desarrollo de software, integrando algoritmos avanzados y estructuras de datos para crear soluciones de software robustas y de calidad. Implica una comprensión profunda de los principios de programación, un enfoque metódico para la solución de problemas y la capacidad de adaptar y mejorar continuamente las prácticas de desarrollo para satisfacer las cambiantes necesidades tecnológicas y las demandas de los diversos sectores.

1. Desarrollar código eficiente, aplicando buenas prácticas de programación y aprovechando las características avanzadas del lenguaje, adquiriendo conocimientos sólidos en la programación utilizando lenguajes y paradigmas de programación relevantes para la industria.

2. Diseñar arquitecturas de software robustas, considerando la eficiencia, la seguridad y la escalabilidad del sistema, así como los demás atributos de calidad que sean requeridos.
Seleccionar, implementar y optimizar algoritmos avanzados y estructuras de datos relevantes para resolver problemas complejos de manera eficiente.

3. Proponer soluciones innovadoras en el diseño y desarrollo de software, en diferentes plataformas y dispositivos, aplicando procesos, métodos y mejores prácticas de ingeniería de software, para desarrollar proyectos medibles, repetibles y de calidad.

4. Diseñar y gestionar bases de datos, tanto de datos estructurados, como no estructurados, transaccionales, distribuidos, heterogéneos o no convencionales, que puedan incluirse dentro de una aplicación de software.

DOMINIOS (Se toman de las competencias)	OBJETOS DE ESTUDIO (Contenidos necesarios para desarrollar cada uno de los dominios)	RESULTADOS DE APRENDIZAJE (Se plantean de los dominios y contenidos)	METODOLOGÍA (Estrategias, secuencias, recursos didácticos)	EVIDENCIAS (Productos tangibles que permiten valorar los resultados de aprendizaje)
B4.2 Utiliza de forma responsable las tecnologías de la información, comunicación, conocimiento y aprendizaje (TICCA), en el proceso de construcción de saberes y el desarrollo de proyectos sociales innovadores en el ámbito digital **DESARROLLO DE PROYECTOS DE INGENIERÍA** 1. Identifica las principales áreas de oportunidad en proyectos complejos de ingeniería para definir estrategias de solución utilizando herramientas tecnológicas y administrativas, para optimizar los procesos de calidad, mejora continua	**1. Fundamentos de las Pruebas de Software.** 1.1. Principios de las pruebas de software 1.2. Justificación de las pruebas de software. 1.3. Actividades de un ingeniero de pruebas. 1.4. Clasificaciones de las pruebas de software. 1.5. Relación entre el Proceso Formal de Pruebas de Software y el Ciclo de Vida del Software.	• Describir los principios de las pruebas de software. • Comprender la justificación de pruebas. • Identificar y justificar los riesgos asociados por la falta de pruebas de software. • Conocimiento de costos y beneficios asociados con la implementación de pruebas de software. • Justificar cómo las pruebas de software	La estrategia para este objeto de estudio consiste en utilizar algunas de los siguientes elementos: • Clase interactiva Maestro – Alumno. • Plataforma institucional. • Portafolio de actividades y/o prácticas. • Visualización y discusión de videos sobre los temas. • proyectos cortos usando la terminología utilizada.	• Prácticas y/o Actividades • Actividad Integradora • Examen

contemplando las normatividades aplicables. 2. Desarrolla proyectos complejos de ingeniería que integra la planeación, análisis, diseño y administración con base en los criterios de sostenibilidad. 3. Identifica los principales factores involucrados en la solución de problemas de ingeniería para desarrollar propuestas utilizando herramientas de ciencias básicas e ingeniería aplicada. 4. Selecciona configuraciones óptimas de los recursos involucrados en proyectos de ingeniería utilizando como base procesos de calidad y mejora continua.		contribuyen al cumplimiento de los requisitos del cliente. ● Explicar cómo las pruebas de software son un componente crucial para el aseguramiento de la calidad en el desarrollo de software. ● Conocer las habilidades necesarias para desempeñarse eficazmente como ingenieros de pruebas de software en entornos profesionales. ● Desarrollar una comprensión sobre las clasificaciones de las pruebas de software y la capacidad de aplicarlas de manera efectiva en situaciones prácticas. ● Comprender y aplicar la relación entre el proceso formal de pruebas de software y el ciclo de vida del software de manera efectiva y eficiente.	● Rúbricas y presentacione s. ● Examen	

DISEÑO Y DESARROLLO DE SOFTWARE 1. Desarrollar código eficiente, aplicando buenas prácticas de programación y aprovechando las características	2. Técnicas de Pruebas de Software 2.1. Pruebas de Caja Negra 2.2. Pruebas de Caja Blanca. 2.3. Pruebas de Caja Gris. 2.4. Pruebas funcionales 2.5. Pruebas no funcionales. 2.6. Pruebas de regresión. 2.7. Pruebas de estrés y rendimiento.	• Conocer las diversas técnicas de pruebas de software. • Elegir y aplicar las técnicas de pruebas de software conforme a los objetivos específicos del	La estrategia para este objeto de estudio consiste en utilizar algunas de los siguientes elementos: • Clase interactiva Maestro – Alumno. • Plataforma institucional.	• Prácticas y/o Actividades • Actividad Integradora • Examen

avanzadas del lenguaje, adquiriendo conocimientos sólidos en la programación utilizando lenguajes y paradigmas de programación relevantes para la industria. 2. Diseñar arquitecturas de software robustas, considerando la eficiencia, la seguridad y la escalabilidad del sistema, así como los demás atributos de calidad que sean requeridos. Seleccionar, implementar y optimizar algoritmos avanzados y estructuras de datos relevantes para resolver problemas complejos de manera eficiente. 3. Proponer soluciones innovadoras en el diseño y desarrollo de software, en diferentes plataformas y dispositivos, aplicando procesos, métodos y mejores prácticas de ingeniería de software, para desarrollar proyectos medibles,	2.8. Pruebas de seguridad.	proyecto y del tipo de software que se esté desarrollando.	• Portafolio de actividades y/o prácticas. • Visualización y discusión de videos sobre los temas. • proyectos cortos usando la terminología utilizada. • Rúbricas y presentacion e s. • Examen	

repetibles y de calidad. 4. Diseñar y gestionar bases de datos, tanto de datos estructurados, como no estructurados, transaccionales, distribuidos, heterogéneos o no convencionales, que puedan incluirse dentro de una aplicación de software.				

1.	3. Niveles de las Pruebas de Software. 3.1. Pruebas Unitarias 3.2. Pruebas de Integración. 3.3. Pruebas de Sistema. 3.4. Pruebas de Aceptación.	***Pruebas Unitarias.*** ● Diseñar casos de prueba para validar unidades individuales de código. ● Ejecutar pruebas unitarias utilizando herramientas específicas y comprender la importancia de la cobertura del código. ● Identificar y corregir defectos a nivel de unidades de código. ***Pruebas de integración.*** ● Diseñar escenarios de prueba que aborden la interacción entre módulos o componentes específicos. ● Ejecutar pruebas de integración y entender cómo se comunican los distintos elementos del sistema.	La estrategia para este objeto de estudio consiste en utilizar algunas de los siguientes elementos: ● Clase interactiva Maestro – Alumno. ● Plataforma institucional. ● Portafolio de actividades y/o prácticas. ● Visualización y discusión de videos sobre los temas. ● proyectos cortos usando la terminología utilizada. ● Rúbricas y presentaciones. ● Examen	● Prácticas y/o Actividades ● Actividad Integradora ● Examen

		● Identificar y resolver problemas de integración, asegurando que los componentes funcionen correctamente. ***Pruebas de sistemas.*** ● Planificar y diseñar pruebas que aborden el sistema en su conjunto, incluyendo casos de prueba que consideren múltiples funciones y escenarios de uso. ● Ejecutar pruebas de sistema y verificar que el software cumple con los requisitos especificados. ● Implementar informes detallados sobre la ejecución de pruebas de sistema, incluyendo métricas de calidad y sus posibles áreas de mejora. ***Pruebas de aceptación del usuario.*** ● Colaborar con usuarios finales para comprender sus expectativas y necesidades en el proceso de pruebas. ● Diseñar y ejecutar las pruebas que		

		validen la aceptación del usuario, reflejando escenarios de		

		uso del mundo real. ● Implementar informes detallados sobre los resultados de las pruebas de aceptación.		
	4. Verificación y Validación. 4.1. Conceptos generales de V&V. 4.2. Actividades para la V&V. 4.3. Métricas aplicables a la V&V.	● Conocer los principios, actividades y métricas asociadas con la verificación y validación en el desarrollo de software. ● Implementar habilidades prácticas para diseñar estrategias efectivas y mejorar continuamente la calidad del software a lo largo del ciclo de vida del desarrollo. ***Conceptos generales de V&V.*** ● Comprender los conceptos fundamentales de verificación y validación en el contexto del desarrollo de software. ● Diferenciar entre los conceptos de verificación y validación, comprendiendo sus roles y objetivos específicos. ● Conocer los principios fundamentales que respaldan la verificación y la validación y cómo estos contribuyen a la calidad del software.	La estrategia para este objeto de estudio consiste en utilizar algunas de los siguientes elementos: ● Clase interactiva Maestro – Alumno. ● Plataforma institucional. ● Portafolio de actividades y/o prácticas. ● Visualización y discusión de videos sobre los temas. ● proyectos cortos usando la terminología utilizada. ● Rúbricas y presentaciones. ● Examen	● Prácticas y/o Actividades ● Actividad Integradora ● Examen

		Actividades para la V&V. ● Diseñar estrategias de verificación que aborden diferentes niveles y aspectos del software, desde las pruebas unitarias hasta las pruebas de sistema. ● Conocer las actividades de validación y ser capaz de diseñar pruebas que confirmen que el software cumple con los requisitos y expectativas del usuario. ● Implementar revisiones de código como parte del proceso de verificación, identificando y corrigiendo posibles defectos. ● Planificar y ejecutar pruebas de aceptación del usuario, colaborando con los usuarios finales para validar la funcionalidad del software. ***Métricas aplicables a la V&V.*** ● Conocer y seleccionar métricas relevantes para evaluar la efectividad de las actividades de verificación y validación en el software.		

	5. Herramientas para las Pruebas de Software. 5.1. Diseño de artefactos de pruebas. 5.2. Depuración. 5.3. Pruebas Automatizadas. 5.4. Estrategias de pruebas. 5.4.1. Pruebas manuales vs. Pruebas automatizadas. 5.4.2. Estrategias de prueba ágiles. 5.4.3. Estrategias de prueba para proyectos tradicionales.	***Diseño de artefactos de pruebas.*** ● Diseñar casos de prueba efectivos teniendo en cuenta la funcionalidad específica a probar, los datos de entrada y los resultados esperados. ● Implementación de escenarios de pruebas. ● Documentar los casos de prueba. ***Depuración.*** ● Identificar y resolver los defectos, utilizando herramientas de depuración para investigar y corregir problemas en el código. ● Analizar los errores encontrados durante las pruebas, comprender su causa y proponer soluciones efectivas. ● Optimizar el código a través de la depuración, mejorando la eficiencia y la calidad del software. ***Pruebas Automatizadas.*** ● Conocer y seleccionar herramientas de automatización de pruebas, conforme a los requisitos del proyecto y las	La estrategia para este objeto de estudio consiste en utilizar algunas de los siguientes elementos: ● Clase interactiva Maestro – Alumno. ● Plataforma institucional. ● Portafolio de actividades y/o prácticas. ● Visualización y discusión de videos sobre los temas. ● proyectos cortos usando la terminología utilizada. ● Rúbricas y presentaciones. ● Examen	● Prácticas ● Actividad Integradora ● Examen

		características del software. ● Desarrollar scripts de pruebas automatizadas, abordando diferentes escenarios y funciones del software. ● Integración y ejecución de pruebas automatizadas, monitoreando los resultados y gestionando la salida de las pruebas. ● Implementar informes detallados a partir de las pruebas automatizadas, proporcionando información sobre la calidad del software. ***Estrategias de pruebas.*** ● Comprender y aplicar estrategias de prueba efectivas, considerando el contexto del proyecto, ya sea en un entorno ágil o en un proyecto tradicional. Además, estarán equipados con habilidades específicas para elegir entre pruebas manuales y pruebas automatizadas según las necesidades del proyecto.		

	6. Casos prácticos y proyectos. 6.1. Aplicación de los conceptos aprendidos en proyectos reales. 6.2. Problemas y desafíos comunes en las pruebas de software.	• Aplicar los conceptos y técnicas aprendidos en proyectos prácticos del mundo real, abordando diferentes aspectos del ciclo de vida del software. • Diseñar y ejecutar casos de prueba en un entorno de proyecto real, adaptándose a los requisitos específicos del software. • Integrar estrategias de pruebas adecuadas en proyectos reales, considerando el tipo de software, los plazos y las restricciones. • Colaborar de manera efectiva con equipos de desarrollo y otros profesionales involucrados en el proyecto, comunicando resultados de pruebas y participando en la mejora continua. • Identificar problemas comunes que pueden surgir durante las pruebas de software, como defectos de diseño, cambios de requisitos y fallos de comunicación.	La estrategia para este objeto de estudio consiste en utilizar algunas de los siguientes elementos: • Clase interactiva Maestro – Alumno. • Plataforma institucional. • Portafolio de actividades y/o prácticas. • Visualización y discusión de video sobre los temas. • Rúbricas y presentaciones. • Desarrollo y Presentación del proyecto integrador.	• Prácticas y/o Actividades • Actividad Integradora • Examen

FUENTES DE INFORMACIÓN (Bibliografía, direcciones electrónicas)	**EVALUACIÓN DE LOS APRENDIZAJES** (Criterios, ponderación e instrumentos)
Gómez Palomo, S. R., & Moraleda Gil, E. (2020). *Aproximación a la ingeniería del software*. Editorial Centro de Estudios Ramón Areces SA. ISBN 978-8499613291. Spillner, A., & Linz, T. (2021). *Software testing foundations: A study guide for the certified tester exam - foundation level - ISTQB® compliant* (5th rev. ed.). dpunkt.verlag. ISBN 9781681988535. Barnum, C. M. (2020). *Usability testing essentials: Ready, set ... test!* (2nd ed.). Morgan Kaufmann. ISBN 9780128169421. Eeles & Cripps. (2010). The Process of Software Architecting. (1a Edición). Addison-Wesley. Rueda Sandoval Gary. (2011). Fundamentos de Pruebas de Software. RBCS, Inc. USA Bolaños Alonso & Daniel. (2008). Pruebas De Software Y Junit. Pearson. Sommerville Ian. (2005). Ingeniería de Software. (7ª Edición). Pearson. ISBN: 978-607-32-0603-7	**Primer parcial:** ● Prácticas y/o actividades 15% ● Actividad integradora 20% ● Examen 65% **Segundo parcial:** ● Prácticas y/o actividades 15% ● Actividad integradora 20% ● Examen 65% **Tercer parcial:** ● Prácticas y/o actividades 15% ● Actividad integradora 20% ● Examen 65% **La calificación mínima es 7.0.** **Se usará rúbrica para la entrega de actividades o tareas a realizar.** .

CRONOGRAMA

Objetos de estudio	**Semanas**															
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
1. Fundamentos de las Pruebas de Software																
2. Técnicas de Pruebas de Software																
3. Niveles de las Pruebas de Software																
4. Verificación y Validación																
5. Herramientas para las Pruebas de Software																
6. Casos prácticos y proyectos.																