**Evaluación Parcial 1**

**Creando nuestro Plan y Casos de Prueba**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sigla** | **Nombre Asignatura** | **Tiempo Asignado** | **% Ponderación** |
| **CSY4111** | **Calidad de Software** | **4 semanas** | **30%** |

**1. Situación evaluativa**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Ejecución práctica** |  | X | **Entrega de encargo** |  | X | **Prueba escrita** |  |  | **Presentación** |

**2. Agente evaluativo**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **X** | **Heteroevaluación** |  |  | **Coevaluación** |  |  | **Autoevaluación** |

**3. Instrucciones para el/la estudiante**

|  |
| --- |
| **Indicaciones para el estudiante**  Esta evaluación tiene un **30%**deponderaciónsobre la nota final de la asignatura y se compone de una Prueba de Conocimientos fundamentales (30%) y Encargo sin Presentación (70%). El **tiempo** para desarrollar esta evaluación es de **4 semanas** y se realiza de manera **grupal** en **laboratorio de computación.**  Para la **Prueba de Conocimientos** deberás responder preguntas orientadas a los criterios de calidad en función de los requerimientos, casos de pruebas tanto funcionales como no funcionales  Para **El Encargo sin Presentación**, corresponde a un Diseño Plan de Pruebas para el que debe integrar los siguientes aspectos:   * Alcance de las pruebas * Descripción * Módulos del Software a Probar * Objetivo del plan de pruebas * Detalle de Ejecución de las pruebas * Tipos de Prueba a realizar * Técnicas y estrategias de pruebas a utilizar * Roles y responsabilidades * Entorno y configuración de ambiente de pruebas * Calendarización de Pruebas * Riesgos del proceso de Pruebas * Artefactos y Criterios de Aceptación |

**3. Pauta de Evaluación**

|  |  |  |
| --- | --- | --- |
| **Categoría** | **% logro** | **Descripción niveles de logro** |
| **Muy buen desempeño** | **100%** | Demuestra un desempeño destacado, evidenciando el logro de todos los aspectos evaluados en el indicador. |
| **Buen desempeño** | **80%** | Demuestra un alto desempeño del indicador, presentando pequeñas omisiones, dificultades y/o errores. |
| **Desempeño aceptable** | **60%** | Demuestra un desempeño competente, evidenciando el logro de los elementos básicos del indicador, pero con omisiones, dificultades o errores. |
| **Desempeño incipiente** | **30%** | Presenta importantes omisiones, dificultades o errores en el desempeño, que no permiten evidenciar los elementos básicos del logro del indicador, por lo que no puede ser considerado competente. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Indicador de Evaluación** | |  | **Categorías de Respuesta** | | | | **Ponderación Indicador de Evaluación** |
| **Muy buen desempeño**  **(100%)** | **Buen Desempeño**  **(80%)** | **Desempeño aceptable**  **(60%)** | **Desempeño incipiente**  **(30%)** | **Desempeño no logrado**  **(0%)** |
| 1. Reconoce los estándares de calidad de software vigentes en la industria. | |  |  |  |  |  | **10%** |
| 1. Reconoce las etapas de planificación y diseño del plan de pruebas, que le permitan evaluar el producto considerando los estándares del aseguramiento de calidad | |  |  |  |  |  | **10%** |
| 1. Define los objetivos y el propósito del Plan de Pruebas, para asegurar la calidad del producto a desarrollar | |  |  |  |  |  | **10%** |
| 1. Define estrategias de pruebas de software, para asegurar el cumplimiento del plan de pruebas | |  |  |  |  |  | **10%** |
| 1. Identifica los criterios de aceptación del proceso de prueba, en función de los distintos requerimientos y objetivos del plan de pruebas. | |  |  |  |  |  | **10%** |
| 1. Utiliza técnicas de partición de equivalencia y análisis de valores límite para el diseño de casos de prueba. | |  |  |  |  |  | **10%** |
| 1. Diseña una estructura clara de casos de prueba funcionales con sus respectivos cursos normales y flujos alternativos, que permitan validar el cumplimiento de requerimientos y calidad del software. | |  |  |  |  |  | **10%** |
| 1. Diseña casos de prueba que verifiquen los requerimientos funcionales del proyecto | |  |  |  |  |  | **10%** |
| 1. Diseña casos de prueba que verifiquen los requerimientos no funcionales del proyecto | |  |  |  |  |  | **10%** |
| 1. Crea una trazabilidad entre los requerimientos, funcionalidades del software y los casos de prueba. | |  |  |  |  |  | **5%** |
| 1. Diseña casos de prueba acorde a las correcciones realizadas priorizando los objetivos del Software | |  |  |  |  |  | **5%** |
|  | **Total** | | | | | | **100%** |