|  |
| --- |
| **Fundamentos de Ing. De Software** |
| Carrera: Inteligencia Artificial | Fecha Actualización: 20 de abril de 2024 |

**PUNTAJES Y NOTA**

|  |  |
| --- | --- |
| **Puntaje máximo: 100** | **Nota:** |
| **Puntaje mínimo de aprobación: 60** |

**Nombre Completo** Nicole Lozada Leon

**C.I.** 10982345 **Fecha:** 13/12/2024 **Link del repositorio de GitHub:** https://github.com/NicoleLozadaLeon/CIFRADOCESAR

**INSTRUCCIONES GENERALES:**

|  |
| --- |
| El objetivo de este proyecto es aplicar los principios de ingeniería de software en el desarrollo de un programa de cifrado César en C# utilizando .NET Core. Los estudiantes deberán demostrar su capacidad para:   * Implementar el algoritmo de cifrado y descifrado César. * Escribir código limpio, estructurado y eficiente en C#. * Crear documentación clara y concisa. * Definir historias de usuario que reflejen las necesidades del usuario. * Utilizar un sistema de control de versiones (GitHub) para gestionar el proyecto. * Desarrollar pruebas unitarias para verificar el correcto funcionamiento del programa.   **Requisitos Funcionales:**   * El programa debe permitir al usuario ingresar una frase y un desplazamiento (número de posiciones a desplazar cada letra). * El programa debe cifrar la frase utilizando el algoritmo César y mostrar el resultado. * El programa debe descifrar una frase cifrada con el algoritmo César, dado el desplazamiento original, y mostrar el resultado. * El programa debe manejar mayúsculas, minúsculas y caracteres especiales.   **Requisitos No Funcionales:**   * El código debe estar bien documentado, siguiendo las convenciones de C#. * El proyecto debe incluir un archivo README.md con instrucciones de instalación y uso. * El proyecto debe incluir documentación, como visto en clases. * El proyecto debe incluir al menos 3 historias de usuario. * El proyecto debe ser gestionado en un repositorio de GitHub. * El proyecto debe incluir pruebas unitarias para las funciones principales.   **Evaluación:**  La evaluación se basará en la rúbrica adjunta, que considera la funcionalidad del programa, la calidad del código, la documentación, las historias de usuario, la gestión del repositorio y las pruebas unitarias. |

A partir de la entrega del planteamiento de este Proyecto final de la asignatura, **usted dispone de los días asignados** para desarrollar el encargo según la planificación establecida por la universidad.

**FECHA DE ENTREGA:**

- **1ra Instancia,** Ver cronograma

- **2da Instancia,** Ver cronograma

El encargo debe ser entregado en la fecha y el horario de clases.

Entregas posteriores, incompletas o atrasadas no serán revisadas y se calificarán con nota 1.0.

**Link del repositorio de GitHub:**

https://github.com/NicoleLozadaLeon/CIFRADOCESAR

**Rúbrica de Evaluación para el Examen de Calculadora de Física**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Criterio | Excelente (hasta 20 puntos) | Bueno (hasta 16 puntos) | Suficiente (hasta 12 puntos) | Insuficiente (hasta 8 puntos) |
| Funcionalidad del cifrado y descifrado | El programa cifra y descifra correctamente frases utilizando el algoritmo César, manejando mayúsculas, minúsculas y caracteres especiales. | El programa funciona correctamente en la mayoría de los casos, pero puede tener errores menores en el manejo de caracteres especiales. | El programa cifra y descifra frases, pero tiene errores significativos en el manejo de mayúsculas/minúsculas o caracteres especiales. | El programa no cifra ni descifra correctamente. |
| Calidad del código (C#) | El código es limpio, bien estructurado, utiliza nombres de variables y métodos descriptivos, y sigue buenas prácticas de programación en C#. | El código es funcional, pero podría mejorarse en cuanto a estructura, nombres de variables/métodos o buenas prácticas. | El código es difícil de entender, mal estructurado o no sigue buenas prácticas de programación. | El código es ilegible o no funciona. |
| Documentación (README.md) | El archivo README.md proporciona una descripción clara del proyecto, instrucciones de instalación y uso, y ejemplos de cómo utilizar el programa. | El README.md proporciona información básica sobre el proyecto, pero podría ser más completo o claro. | El README.md es incompleto o no proporciona información suficiente para entender y usar el proyecto. | No se proporciona un archivo README.md o es inadecuado. |
| Historias de usuario | Se definen al menos 3 historias de usuario claras y concisas que describen las necesidades y expectativas de los usuarios del programa. | Se definen historias de usuario, pero podrían ser más claras, concisas o relevantes. | Se definen pocas historias de usuario o no son lo suficientemente claras. | No se definen historias de usuario. |
| Gestión de repositorio (GitHub) | El repositorio de GitHub está bien organizado, con commits descriptivos, uso de ramas y un historial de desarrollo claro. | El repositorio de GitHub está organizado, pero podría mejorarse en cuanto a commits, uso de ramas o historial de desarrollo. | El repositorio de GitHub está desorganizado o no se utiliza de manera efectiva. | No se utiliza un repositorio de GitHub. |
| Pruebas unitarias | Se incluyen pruebas unitarias que cubren las funciones principales del programa y aseguran su correcto funcionamiento. | Se incluyen algunas pruebas unitarias, pero no cubren todas las funciones o casos de uso. | Se incluyen pocas pruebas unitarias o no son efectivas para verificar el funcionamiento del programa. | No se incluyen pruebas unitarias. |