fff

**TAREA**

| **Nombre de la asignatura** | | | *Lógica de Programación* |
| --- | --- | --- | --- |
| **Unidad N°** | **3-4** | *Metodología de la programación* | |
| **Tema N°** | **1-4** | *Herramientas tecnológicas para el diseño de algoritmos* | |

|  |
| --- |
| **Tipo de actividad** |
| Trabajo de Investigacion |

|  |
| --- |
| **Objetivo de la actividad** |
| Desarrollo de ejercicios de lógica interactivos con JavaScript. |

|  |
| --- |
| **INSTRUCCIONES** |
| Bienvenido(a), a la actividad de aprendizaje practico **No 2:**Trabajo de Investigación.  **Instrucciones:**   1. El profesor, coordinará la organización de los equipos de trabajo colaborativo dentro del aula de clases, máximo 5 estudiante por equipo. 2. Subir como evidencia el proyecto al repositorio de GitHub.     c) La entrega de trabajo grupal es individual, y su calificación será grupal, deberá describir los integrantes de grupo en el trabajo compartido.  **Caso de Estudio para el Proyecto Final del Curso de Lógica de Programación**  **1. Antecedentes**  Este proyecto tiene como objetivo principal consolidar los conocimientos adquiridos por los estudiantes a lo largo del curso de Lógica de Programación mediante la elaboración de un sitio web interactivo. El sitio web servirá como una plataforma de ejercicios prácticos que abordan distintas categorías de problemas de programación. Estas categorías incluyen ejercicios de simulación, manejo de números, manipulación de cadenas y operaciones con arreglos.  El desarrollo de este proyecto permitirá a los estudiantes no solo aplicar los conceptos teóricos aprendidos, sino también desarrollar habilidades en la creación de sitios web, integrando HTML, CSS, JavaScript y posiblemente frameworks como Bootstrap.  **2. Objetivo General**  Crear un sitio web que contenga una interfaz de usuario amigable e interactiva, donde los estudiantes puedan acceder a ejercicios categorizados por temáticas y resolver problemas prácticos. El objetivo es fomentar el aprendizaje de la programación mediante la resolución de problemas lógicos y algorítmicos.  **3. Objetivos Específicos**   * Desarrollar un sitio web que contenga las siguientes secciones: Dashboard, Números, Cadenas, Arreglos y un simulador. * Implementar páginas individuales para cada categoría de ejercicios, con una lista de problemas y un formulario para la solución de cada uno. * Permitir al usuario ingresar soluciones y obtener retroalimentación automática. * Proporcionar un entorno de desarrollo que facilite la interacción con las herramientas y tecnologías de programación. * Incluir un proyecto base como referencia para guiar el desarrollo.   **4. Funcionalidades del Sistema**  El sistema deberá permitir a los estudiantes navegar por las categorías de ejercicios, seleccionar un ejercicio, leer la descripción y resolver el problema en un formulario interactivo.  **5. Estructura del Sitio Web**  El sitio web tendrá la siguiente estructura de menús:   1. **Dashboard**    * Visión general del sistema y acceso rápido a todas las categorías. 2. **Categorías de Ejercicios**    * **Simulador de Moodle**      + Ejercicios simulando un entorno de aprendizaje similar a Moodle.    * **Números**      + Ejercicios enfocados en operaciones numéricas.    * **Cadenas**      + Ejercicios de manipulación y análisis de cadenas.    * **Arreglos**      + Ejercicios relacionados con la manipulación de arreglos y listas.   **6. Descripción de las Categorías y Ejercicios**  **Categoría 1: Números** Los ejercicios en esta categoría se centran en operaciones y cálculos numéricos. Ejemplos de problemas:   1. **Imprimir números impares**: Imprimir todos los números impares entre 1 y 50. 2. **Calcular la suma de los primeros 100 números**: Sumar todos los números del 1 al 100. 3. **Suma de números pares**: Sumar todos los números pares desde 1 hasta un número dado por el usuario. 4. **Números divisibles por 3 y 5**: Imprimir todos los números del 1 al 100 que sean divisibles por 3 y 5. 5. **Generar la tabla de multiplicar**: Generar la tabla de multiplicar de un número ingresado por el usuario. 6. **Números Fibonacci**: Imprimir los primeros n números de la serie de Fibonacci.   Ejemplo: Si n= 7 la serie es: 0 1 1 2 3 5 8 13   1. **Calcular la multiplicación de n números**: Multiplicar todos los números del 1 al 10 2. **Factorial de un número**: Calcular el factorial de un número dado por el usuario. Num=5 = resp= 5\*4\*3\*2\*1 = 120 3. **Imprimir múltiplos de un número**: Imprimir todos los divisores de un número dado entre 1 y 20. 4. **Presentar los divisores de un número**: Mostrar los divisores de un número excluyendo el propio número. 5. **Número perfecto**: Verificar si un número es perfecto (la suma de sus divisores excluyendo el número da como resultado el propio número). 6. **Comprobación de números primos**: Verificar si un número dado es primo. 7. **Invertir un número**: Invertir los dígitos de un número ingresado por el usuario. 8. **Número de dígitos de un número**: Contar cuántos dígitos tiene un número entero positivo.   **15. Dibujo de un triángulo de asteriscos**: Imprimir un triángulo de asteriscos de altura n tal como se indica en el ejemplo. Ejemplo para **n = 5: \***  **\*\***  **\*\*\***  **\*\*\*\***  **\*\*\*\*\***  **Categoría 2: Cadenas** Esta categoría se enfoca en la manipulación y análisis de cadenas de texto. Ejemplos de problemas:   1. **Ingresar un nombre** **y un apellido:** dado un nombre y apellido ingresado presentarlo concatenado: apellido y nombre 2. **Comparar frases:** Dada dos frases indicar si son iguales o la primera en mayor o menor que la segunda. 3. **Ingresar palabra:** Ingresar una palabra y crear otra palabra con cada carácter de la primera palabra dejando un espacio entre cada caracter. 4. **Dada una frase:** presentar cada carácter con su respectivo código asccii. Para este ejercicio usar una función de javascript que devuelve el código asccii del carácter 5. **Invertir una palabra**: Invertir una frase ingresada por el usuario 6. **Contar vocales en una palabra**: Contar cuántas vocales tiene una palabra ingresada por el usuario. 7. **Contar las letras del alfabeto en una palabra**: Contar cuántas letras del alfabeto tiene una palabra ingresada por el usuario. 8. **Contar dígitos de una frase**: Contar cuántos dígitos tiene una frase ingresada por el usuario. 9. **Contar los signos de puntuación en una palabra**: Contar cuántos signos de puntuación tiene una palabra ingresada por el usuario. 10. **Contar palabras en una frase**: Contar cuántas palabras tiene una frase ingresada por el usuario.   **Categoría 3: Arreglos** Los ejercicios en esta categoría permiten trabajar con estructuras de datos como arreglos. Ejemplos de problemas:   1. **Dado un arreglo de 10 números** generados aleatoriamente recorrer el arreglo y presentar cada número del arreglo. 2. **Promedio de calificaciones**: Calcular el promedio de cinco calificaciones ingresadas por el usuario. 3. **Encontrar el máximo de una lista de números**: Encontrar el número más grande en una lista de números. 4. **Encontrar el promedio de números mayores a un valor dado**: Calcular el promedio de los números mayores a un valor límite en un arreglo. 5. **Encontrar el mínimo de un arreglo de números**: Encontrar el número más pequeño de un arrelo. 6. **Promedio de números pares de un arreglo**: Calcular el promedio de los números pares en un arreglo de 10 números. 7. **Arreglo suma dígitos**: Almacenar los dígitos de un número en un arreglo y luego calcular la suma de los dígitos. 8. **Contar y sumar números positivos y negativos**: Contar y sumar cuántos de 10 números ingresados son positivos o negativos. 9. **Dado un arreglo** de números crear otro arreglo con el cuadrado de cada elemento del primer arreglo 10. **Dado dos arreglos** con la misma dimensión crear otro arreglo con la suma de cada elemento del arreglo1 con el arreglo2 en el orden de la misma posición.   **Categoría 4: Simulador** Esta sección incluye ejercicios prácticos similares a los exámenes en Moodle. Los estudiantes podrán practicar problemas que requieren un enfoque similar al de un entorno de examen.  **7. Diseño del Sitio Web**   * **Página de Inicio (Dashboard)**: Página principal con un resumen de todas las categorías y acceso rápido a las mismas. * **Página de Categorías**: Al hacer clic en una categoría, se mostrará una lista de ejercicios. * **Página de Ejercicios**: Cada ejercicio tendrá una descripción detallada, un área para ingresar la respuesta y un botón para evaluar la solución.   **8. Requerimientos Técnicos**   * **Frontend**:   + HTML, CSS, JavaScript y Bootstrap para un diseño responsivo y atractivo.   + Utilización de formularios para la interacción del usuario. * **Backend**:   + Implementación de lógica en JavaScript para evaluar las respuestas ingresadas por los estudiantes.   **9. Metodología**   * **Fase 1: Análisis y diseño**   + Definir la estructura del sitio web y los requisitos funcionales.   + Diseñar las páginas y formularios necesarios para cada categoría. * **Fase 2: Desarrollo**   + Implementar la estructura del sitio web y las funcionalidades interactivas.   + Desarrollar los ejercicios con sus respectivas soluciones y retroalimentación. * **Fase 3: Pruebas**   + Realizar pruebas para asegurar que el sistema funcione correctamente y que los ejercicios se evalúen de manera precisa. * **Fase 4: Implementación**   + Publicar el sitio web en un servidor para que los estudiantes puedan acceder y practicar.   **10. Proyecto Base**  Se adjunta un **proyecto base** que incluye la estructura inicial del sitio web con ejemplos de las páginas de las categorías y un formulario de ejemplo para un ejercicio. Los estudiantes deberán utilizar este proyecto como punto de partida y personalizarlo según los requisitos específicos de los ejercicios. |

|  |
| --- |
| **RECOMENDACIONES** |
| * Recuerda que el trabajo será validado mediante preguntas de como ha realizado * el desarrollo. * Revise el material de diapositivas y ejercicios realizados en el curso. * Antes de enviar el trabajo, tome en consideración estos aspectos: * Revise la gramática y ortografía. * Organice las ideas que vaya a utilizar. * Tome en cuenta los parámetros y tiempos establecidos. * Revise la rúbrica de evaluación. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **RÚBRICA DE LA ACTIVIDAD** | | | | |
| **INDICADORES** | **MALO** | **BUENO** | **MUY BUENO** | **PUNTAJE** |
|  |  |  |
| **Presentación** | La estructura del trabajo no cumple con lo mínimo exigible.  0 Punto | La estructura del trabajo cumple con lo mínimo exigible.  0.5 Punto | La estructura del trabajo se presenta en forma adecuada y el contenido se ajusta plenamente a lo solicitado  1 Punto | **1.5** |
| **Diseño** | El diseño del trabajo no se ajusta plenamente a lo solicitado  0.5 Punto | El diseño del trabajo presenta ineficiencia  1 Punto | El diseño del trabajo se presenta en forma eficiente y se ajusta plenamente a lo solicitado  2 Puntos | **3.5** |
| **Desarrollo** | El desarrollo del trabajo presenta errores  1 Punto | El desarrollo del trabajo presenta ineficiencia y no se ajusta plenamente a lo solicitado  2 Puntos | El desarrollo del trabajo se presenta en forma eficiente y se ajusta plenamente a lo solicitado  3 Puntos | **5.0** |
| **TOTAL** | | | | ***10*** |

|  |
| --- |
| **INFORMACIÓN PARA EL**  **DESARROLLO DEL RECURSO EN PLATAFORMA** |

|  |  |  |  |
| --- | --- | --- | --- |
| **N° UNIDAD** | **TEMA** | **N° SEMANA** | **FECHA** |
| 3-4 | *Herramientas tecnológicas para el diseño de algoritmos* | 8-14 | 15/05/2020 |

|  |  |
| --- | --- |
| **CRITERIOS DE CONFIGURACIÓN DE ACTIVIDAD** | |
| **Permitir entregas desde:** | 15/05/2020 |
| **Fecha límite de entrega:** | 15/05/2020 |
| **Fecha de publicación de la calificación:** | 15/05/2020 |

|  |  |
| --- | --- |
| **Formato en el que el estudiante debe entregar la actividad** (Marque una casilla con una X) | |
| PDF | **X** |
| WORD |  |
| POWER POINT |  |

|  |
| --- |
| **Detalles adicionales para configurar en la actividad** |
|  |