Ejercicio #1

Positivo.	Negativo.	Interesante.
Facilita la interacción con el usuario mediante la entrada de un número natural.	No valida si el número ingresado es realmente un número natural.	Usa la librería estándar <iostream> para entrada y salida de datos.
Calcula la suma de los números naturales desde 1 hasta el número ingresado.	No maneja errores si el usuario ingresa un valor no válido.	Declara e inicializa variables para almacenar el número ingresado y la suma de los números naturales.
Utiliza un bucle for para iterar sobre los números naturales.	No proporciona información sobre cómo se calcula la suma de los números naturales.	

Ejercicio #2

Positivo.	Negativo.	Interesante.
Imprime números impares en orden descendente desde 100 hasta 1.	No proporciona flexibilidad para especificar el rango de números a imprimir.	Usa el operador de módulo (%) para verificar si un número es impar.
Utiliza un bucle for para iterar desde 100 hasta 1.	No incluye una función principal (main) con el tipo de retorno adecuado (int).	Utiliza una función (imprimir) para encapsular la lógica de impresión de números impares.
Verifica si un número es impar antes de imprimirlo.	No maneja errores si no se pueden imprimir los números impares.	Imprime los números impares en orden descendente, lo que puede ser útil en ciertos contextos.

Ejercicio #3

Positivo.	Negativo.	Interesante.
Calcula correctamente la suma de los cuadrados de los números del 1 al 100.	Utiliza el operador incorrecto para elevar al cuadrado (debería ser i * i o pow(i, 2) en lugar de i^2).	Intenta calcular la suma de los cuadrados de los números utilizando una expresión concisa.
Utiliza un bucle for para iterar sobre los números del 1 al 100.	La función principal (main) no tiene el tipo de retorno especificado (debería ser int main()).	Imprime el resultado de la suma de los cuadrados de los números entre 1 y 100.
Actualiza correctamente la variable de suma con el cuadrado de cada número.	El bucle incluye el número 0, lo que no es necesario para la tarea.	

Ejercicio #4

Positivo.	Negativo.	Interesante.
Permite al usuario agregar las notas de los estudiantes y calcular el promedio de la sección.	No valida si las notas ingresadas están dentro del rango válido (0-10).	Informa al usuario si no se han ingresado todas las notas antes de calcular el promedio.
Utiliza funciones para separar la lógica de agregar notas y calcular el promedio.	No verifica si el usuario ingresa una opción válida en el menú principal.	Usa una estructura de switch para manejar las diferentes opciones del menú.
Proporciona opciones claras al usuario para agregar notas, calcular el promedio o salir del programa.	Utiliza la recursión en la función principal (main), lo que puede causar problemas de legibilidad y eficiencia.	
Calcula correctamente el promedio de las notas de los estudiantes.		

Ejercicio #5

Positivo.	Negativo.	Interesante.
Permite al usuario agregar las notas de los estudiantes y realizar diferentes cálculos estadísticos.	No valida si las notas ingresadas están dentro del rango válido (0-100).	Informa al usuario si no se han ingresado todas las notas antes de calcular la cantidad de aprobados y reprobados o el promedio.
Divide la funcionalidad en funciones separadas para agregar notas, calcular la cantidad de aprobados y reprobados, y calcular el promedio.	No verifica si el usuario ingresa una opción válida en el menú principal.	Utiliza una estructura de switch para manejar las diferentes opciones del menú.
Presenta opciones claras al usuario para agregar notas, ver la cantidad de aprobados y reprobados, calcular el promedio o salir del programa.	Utiliza la recursión en la función principal (main), lo que puede causar problemas de legibilidad y eficiencia.	
Calcula correctamente la cantidad de alumnos aprobados y reprobados, así como el promedio general del grupo.		

Ejercicio #6

Positivo.	Negativo.	Interesante.
Suma correctamente los números pares entre 100 y 200.	No verifica si el rango especificado incluye números pares.	Utiliza una estructura de control (if) para verificar si un número es par antes de sumarlo.
Utiliza una función separada para realizar la suma de los números pares.	No verifica si no hay números pares en el rango especificado.	Imprime el resultado de la suma de los números pares.
Utiliza un bucle for para iterar sobre los números entre 100 y 200.	La función principal (main) no tiene el tipo de retorno especificado (debería ser int main()).	
Presenta el resultado de la suma de los números pares.		

Ejercicio #7

Positivo.	Negativo.	Interesante.
Permite al usuario ingresar los elementos de dos vectores de igual longitud.	No valida si los elementos ingresados son números enteros.	Utiliza un bucle for para ingresar los elementos de los vectores y para sumar los elementos correspondientes.
Suma correctamente los elementos de los dos vectores y almacena el resultado en un tercer vector.	No verifica si los dos vectores tienen la misma longitud antes de realizar la suma.	Muestra el resultado de la suma en un formato adecuado, con los elementos del vector separados por comas y encerrados entre corchetes.
Utiliza funciones separadas para ingresar los vectores y realizar la suma.	Utiliza la recursión en la función principal (main), lo que puede causar problemas de legibilidad y eficiencia.	
Presenta opciones claras al usuario para ingresar los vectores, realizar la suma o salir del programa.		

Ejercicio #8

Positivo.	Negativo.	Interesante.
Permite al usuario ingresar los elementos de dos vectores de la misma longitud.	No valida si los elementos ingresados son números enteros.	Utiliza un bucle for para ingresar los elementos de los vectores y para calcular el producto punto.
Calcula correctamente el producto punto de los dos vectores.	No verifica si los dos vectores tienen la misma longitud antes de calcular el producto punto.	Muestra el resultado del producto punto al finalizar el cálculo.
Utiliza una función separada para ingresar los vectores y otra función para calcular el producto punto.	Utiliza la recursión en la función principal (main), lo que puede causar problemas de legibilidad y eficiencia.	
Presenta opciones claras al usuario para ingresar los vectores, calcular el producto punto o salir del programa.		

Ejercicio #9

Positivo.	Negativo.	Interesante.
Permite al usuario ingresar los elementos de dos matrices de acuerdo a sus dimensiones.	No valida si las dimensiones de las matrices ingresadas son compatibles para la multiplicación.	Utiliza una función separada para ingresar los elementos de cada matriz.
Calcula correctamente el producto de las dos matrices si las dimensiones son compatibles.	No verifica si los elementos ingresados son números enteros.	Utiliza una función separada para realizar la multiplicación de matrices.
Muestra las dos matrices ingresadas y la matriz resultante del producto.	Utiliza la recursión en la función principal (main), lo que puede causar problemas de legibilidad y eficiencia.	Presenta opciones claras al usuario para ingresar las matrices, realizar la multiplicación o salir del programa.

Ejercicio #10

Positivo.	Negativo.	Interesante.
Permite al usuario ingresar los elementos de una matriz.	No valida si las dimensiones de la matriz ingresada son válidas.	Utiliza una función separada para ingresar los elementos de la matriz.
Calcula correctamente la transposición de la matriz ingresada.	No verifica si los elementos ingresados son números enteros.	Utiliza una función separada para calcular la transposición de la matriz.
Muestra la matriz ingresada y su transpuesta como salida.	Utiliza la recursión en la función principal (main), lo que puede causar problemas de legibilidad y eficiencia.	Presenta opciones claras al usuario para ingresar la matriz, calcular la transposición o salir del programa.