UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
ESCUELA NACIONAL DE ESTUDIOS SUPERIORES
UNIDAD JURIQUILLA

Ingeniería Aeroespacial

Semestre II

Fundamentos de Programación

Práctica II

"Solución de problemas y algoritmos"

EQUIPO 3

Integrante 1: Edgar Iván Padilla Huesca **Cuenta:** 317183243
Integrante 2: Jesús Daniel Andrade Mendoza **Cuenta:** 425062704
Integrante 3: Fryda Itzel Bastida Reyes **Cuenta:** 322119071

Profesor: Ing. Andres David Flores Ferro

Fecha de entrega: 14 – Febrero - 2025

Fecha de revisión:

Calificación:

Observaciones:

Objetivo General

Elaborar algoritmos correctos y eficientes en la solución de problemas siguiendo las etapas de análisis y diseño pertenecientes al Ciclo de vida del software.

Objetivo Particular

A partir del enunciado de un problema:

- identificar el conjunto de entrada y el conjunto de salida.
- Elaborar un algoritmo que resuelva un problema determinado (dado por el profesor), identificando los módulos de entrada, de procesamiento y de salida.

Introducción

Este informe de laboratorio explora la generación de algoritmos y el desarrollo de software como herramientas fundamentales para la resolución de problemas. Se profundiza en el ciclo de vida del software, desde la concepción inicial del problema hasta la implementación de soluciones y su mantenimiento continuo. El objetivo principal es desarrollar la capacidad de crear algoritmos correctos y eficientes, que no solo resuelvan el problema planteado, sino que también sean optimizados para su rendimiento.

El informe abarca una variedad de temas relevantes, incluyendo el análisis de protocolos y requerimientos, donde se aprende a identificar las necesidades del usuario y a traducir estas en especificaciones técnicas claras. Se discute la importancia de comprender el dominio del problema, como en el análisis del radio y área del círculo, donde se exploran diferentes métodos para calcular el área y se considera la precisión de los datos.

Además, se examinan aspectos cruciales como la definición precisa de variables y mensajes de salida, la inicialización de números reales para evitar errores, y el diseño de un flujo de mensajes claro e intuitivo para el usuario. Se enfatiza la necesidad de un enfoque metódico para la resolución de problemas, similar a seguir una receta, y se destaca la importancia de la experiencia del usuario al diseñar aplicaciones.

El informe también aborda temas como la optimización del código, el manejo de errores y la importancia de las pruebas exhaustivas. Se discute la necesidad de considerar factores externos como la distribución del teclado y se exploran temas como la generación de programas complejos y el análisis de resultados.

En particular, se abordan dos problemas fundamentales: el primero consiste en la elaboración de un programa capaz de calcular el promedio, la mediana y la moda de un conjunto de n números, lo que requiere la correcta identificación y procesamiento de los datos de entrada. El segundo problema implica la ordenación de tres números de menor a mayor, un ejercicio clave en la comprensión de estructuras condicionales y algoritmos de ordenamiento. En ambos casos, el diseño de los algoritmos incluyó la identificación del problema, análisis de requerimientos y restricciones, identificación de entradas y salidas, planificación del proceso a seguir y finalmente una prueba de escritorio, asegurando una solución eficiente y estructurada. A continuación se encuentran detalladamente.

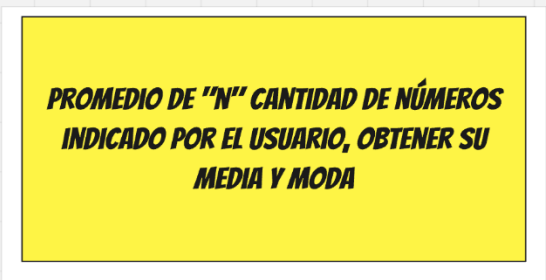
Ejercicio I

Promedio de "n" cantidad de numeros indicado por el usuario, obtener su media y moda.

Desarrollo

1.- Definir el problema

Localizamos de manera clara el enunciado del problema a solucionar.



2.- Analizar los requerimientos del problema

Del enunciado del problema, identificar de manera concisa solamente lo que requiere el problema.

Calcular el promedio de números ingresados por el usuario.
Calcular la media de los números ingresados por el usuario.
Calcular la moda de los números ingresados por el usuario.

Como se observa en la imagen, solo contamos con tres requerimientos del programa, que es solamente lo que pide el problema. Son los requerimientos esenciales: calcular promedio, mediana y moda de n números, sin embargo, el programador puede proponer más requerimientos para una mejor definición del programa.

3.- Analizar los requerimientos propuestos por el programador

Ya se identifico lo esencial del problema, pero no es suficiente para la solución del problema, por lo que los siguientes requerimientos son propuestos por el programador, ya que el programa no necesariamente lo pide, pero el programador los considera necesarios.

Se usará lenguaje de programación C++, por su facilidad en el manejo de listas.
Se solicita al usuario solo utilizar números enteros
Se solicita al usuario el número "n" de valores a calcular.
Se solicita al usuario los n valores numéricos y se almacenan en una lista.
Se propone la siguiente operación el promedio: suma de todos los valores, entre "n" (cantidad de datos ingresados).
Se propone la siguiente para la mediana: ordenar los números de menor a mayor y encontrar el punto medio.
Se propone los siguiente para la moda: identificar el número que más se repite del listado numérico.
Se muestran en la pantalla los resultados de promedio, media, moda de forma clara.

Ejercicio I — Promedio de "n" cantidad de numeros indicado por el usuario, obtener su media y moda.

Por sugerencia del programador:

- Es importante definir el lenguaje de programación en el que se le dará resolución al problema, ya que hay muchos, y unos cuentan con sintaxis más eficientes para ciertas funciones o simplemente por comodidad y facilidad del programador.
- El programa necesita entradas numéricas, pero estas necesitan ser limitadas ya que existen diferentes clasificaciones de números. Los números enteros son más frecuentemente usados para el calculo de: promedio, mediana y moda.
- El usuario debe indicar la cantidad de valores a calcular (n) ya que el programa estará habilitado para n cantidad de números, es decir, no hay un parámetro definido.
- El usuario va a ingresar los números de los cálculos, estos deberán ser almacenados en una lista para las tres operaciones a realizar, y el usuario no tenga que introducir los números nuevamente cada que se termine de ejecutar el promedio, la mediana, o la moda.
- El problema no menciona como tiene que ser la operación a realizar, por lo tanto para promedio se usará la suma de las entradas del usuario entre cantidad de valores (n)
- El problema no menciona como tiene que ser la operación a realizar, por lo tanto para la mediana se guardarán todas las entradas del usuario en una lista, que será ordenada de menor a mayor.
- El problema no menciona como tiene que ser la operación a realizar, por lo tanto para la moda se identificará como el valor que aparece con mayor frecuencia en la lista, previamente realizada.

4.- Analizar las restricciones del problema

Son restricciones esenciales del problema que aparecen en el enunciado de manera explícita o implícita.

El número de valores "n" debe ser un entero positivo mayor a 0
Los valores ingresados deben ser números reales (positivos, negativos, decimales)
Si el usuario ingresa letras o caracteres especiales, será invalidado
En el caso del promedio: no se permite la división entre 0
En el caso de la mediana: debe calcularse solo después de ordenar la lista de valores
En el caso de la mediana: Si n es impar, la mediana es el valor central de la lista ordenada
En el caso de la mediana: Si n es par, la mediana es el promedio de los dos valores centrales
En el caso de la moda: Si más de un número se repite la misma cantidad de veces, el programa debe mostrar todas las modas y la frecuencia con la que aparecen
En el caso de la moda: Si todos los números aparecen con la misma frecuencia, el programa deberá mostrar que no hay moda
El programa debe mostrar los resultados en formato numérico con al menos dos decimales

- El número n o cantidad de valores a ingresar necesariamente tiene que ser un número positivo, mayor a 0 porque las divisiones entre números negativos y 0 son indefinidas.
- Los números a calcular tienen que ser reales, aquí si pueden entrar positivos, negativos, decimales.
- Si el usuario ingresa cualquier valor que no sea numérico (letras, caracteres especiales e incluso entradas vacías) será invalidado ya que este problema esta restringido solo por números.
- La mediana solo se puede calcular luego de ordenar los números de menor a mayor, sino no es mediana. Se presentan dos casos:
  Si (n) es impar se tomará el valor medio
  Si (n) es par se tomarán los dos valores centrales, que se promediarán.
- La moda es el numero que más se repite en un listado numérico, pero hay casos donde hay dos o más modas, por lo tanto el programa tiene que ser capaz de identificarlas e indicar la frecuencia con la que aparecen, e incluso si no hay moda.
- Las salidas tienen que ser muy claras, para no confundir al usuario y darle respuestas optimas.

Ejercicio I — Promedio de "n" cantidad de numeros indicado por el usuario, obtener su media y moda.

5.- Analizar las restricciones propuestas por el programador

Así como el programador sugiere requerimientos, de estos salen restricciones, que no son mencionados en el enunciado del problema, por lo tanto son propuestas por el programador.

Se debe validar que el usuario ingrese un número entero positivo para n, y diferente de O

Se deben manejar excepciones para evitar que el programa se detenga si el usuario introduce un valor no válido

En caso de error en la entrada, el programa debe solicitar al usuario que reingrese los datos hasta que sean correctos.

No se debe permitir que el programa termine abruptamente sin mostrar un mensaje de error en caso de fallo

El código debe ser fácilmente modificable y mantenible si llegará a requerir modificaciones

Por sugerencia del programador:

- Un requerimiento del problema es que sean números, pero es importante validar que cada entrada cumpla con las condiciones mencionadas
- En caso de error en la entrada, el programa no se tiene que cerrar, sino, solicitar que ingresa un valor correcto, y el algoritmo seguirá hasta que lo haga
- No se debe permitir que se cierre el programa, sin indicarle al usuario el error que se esta cometiendo, para que lo pueda corregir
- El código debe ser de fácil acceso, por si en el futuro necesita ajustes o algún cambio.

6.- Entradas

Se deben identificar las entradas que hace el usuario.

Un número entero positivo mayor que O que indica cuántos valores ingresara el usuario. (n)

n valores numéricos ingresados por el usuario

- El usuario solamente tiene que hacer dos entradas, si hace uso correcto del programa: cantidad de números a calcular, y todos los números correspondientes. La segunda entrada depende de la n de la primera entrada.
- El usuario puede cometer errores, por lo tanto se abre brecha a entradas de corrección cuantas veces sea necesario, para evitar que se cierre el programa, sin una ejecución correcta.

7.- Salidas

Se deben identificar las salidas que dará el programa para llevar de la mano al usuario, o bien, darle instrucciones sobre como tiene que ir llevando el programa y así evitar errores.

Salidas Intermedias

Mensaje de bienvenida: "Bienvenido, con este programa podrás calcular el promedio, moda y mediana de n números. Da clic para continuar "

Mensaje pidiendo al usuario ingresar n: "Ingrese la cantidad de números"

Mensaje pidiendo al usuario cada número: "Ingrese el número uno" ... etc.

Van dándole instrucciones al usuario, indican lo que deberá ingresar al programa

- Comienza indicando que hace el programa
- Pide las dos entradas esenciales que tiene que hacer el usuario para obtener los cálculos necesarios

Salidas de validación y corrección

Si el usuario ingresa un n inválido: "Error: ingrese un número entero positivo mayor que O"

Si el usuario ingresa un valor no numérico: "Error: solo se permiten números, intente de nuevo"

Van dándole instrucciones al usuario, indican lo que deberá hacer si ocurre un error

- En caso de cualquier error, el usuario es informado sobre lo que no esta reconociendo el programa con un mensaje, y se le indica que continue
- Como los números a calcular solo se introducen una vez, se le muestra un mensaje con los datos ingresados para que note si estos son los deseados.

Salidas de cierre

Mensaje mostrando las respuestas:
"Resultados
Promedio:.....
Mediana: .....
Moda: ..... "

Si hay varias modas, se mostrarán todas: "En este caso hay más de una moda, son: ....., ..., ..."

Si no hay moda: "No hay moda, todos los valores aparecen con la misma frecuencia"

Mensaje indicando que el programa terminó correctamente: "Cálculos finalizados. Gracias por usar el programa"

Van dándole instrucciones al usuario cuando se terminan procesos

- Cuando se acaban de ejecutar las operaciones aparecerá un mensaje con las respuestas.
- Dado sea el caso de cada operación, tenga una particularidad, también aparecerá un mensaje con lo que ocurrió.
- Se le indicara al usuario que el programa ha acabado de ejecutarse, con un mensaje.

8.- Proceso

Se analiza meticulosamente como se ejecutará el programa paso a paso.

Las dividimos en fases para distinguir las partes del algoritmo

FASE 1
Inicio del programa

1.- El programa inicia

2.- Se muestra un mensaje en la pantalla con el siguiente texto : " Bienvenido, con este programa podrás calcular el promedio, moda y mediana de n números. Da clic para continuar"

En la fase del inicio del programa se distingue por indicar al usuario de lo que hace con un mensaje de bienvenida

Ejercicio I

Promedio de "n" cantidad de numeros indicado por el usuario, obtener su media y moda.

FASE 2
Entrada de datos

3.- Se muestra un mensaje en la pantalla con el siguiente texto: "Ingrese la cantidad de números (n) ".
- Si el usuario ingresa un valor correcto (n es un número entero positivo mayor que 0). Seguir con el paso 4
- Si el usuario ingresa un n inválido, mostrar un mensaje con el siguiente texto : "Error: ingrese un número entero positivo mayor que 0"
- Si el usuario ingresa un valor invalido (no numérico, caracteres especiales, etc.), mostrar un mensaje con el siguiente texto:
  "Error: solo se permiten números, intente de nuevo"
- Este proceso de validación se repetirá hasta que el usuario ingrese un valor válido.

4.- Solicitar al usuario que ingrese los n número, mostrando un mensaje con el siguiente texto: "Ingrese el número uno, ..., etc."
- Cada número ingresado se almacena en una lista para su posterior procesamiento.
- Si el usuario ingresa un valor inválido, el programa mostrará error y se solicita el número de nuevo
- Este proceso se repite hasta que todos los números ingresados sean correctos.

5.- No borrar los datos de pantalla para que el usuario confirme de que los números ingresados son los deseados.

En la fase de entrada de datos es la etapa donde el usuario indicará los números que desea calcular, así mismo cada entrada tendrá que ser validada, de no ser reconocida se solicitará intentarlo de nuevo tantas veces sea necesario.

FASE 3
Cálculos y análisis del programa

6.- Cálculo del promedio
- Se suman todos los números ingresados
- Se divide la suma entre n (cantidad de números)

7.- Cálculo de la mediana
- Los números se ordenan de menor a mayor
- Si n es impar, la mediana es el número central de la lista ordenada
- Si n es par , la mediana se obtiene promediando los dos valores centrales

8.- Cálculo de la moda
- Se cuentan cuántas veces aparece cada número en la lista.
- El número que más veces aparece es la moda
- Si hay varias modas se mostrará un mensaje en la pantalla con el siguiente texto:
  "En este caso hay más de una moda, son: ....., ..., ..."
- Si todos los números aparecen la misma cantidad de veces se mostrará un mensaje con el siguiente texto:
  "No hay moda, todos los valores aparecen con la misma frecuencia"

En la fase de cálculos y análisis del problema, ya no hay entradas por parte del usuario, solamente el análisis del programa, se harán todas las operaciones necesarias para obtener el promedio, mediana, moda

Ejercicio I *Promedio de "n" cantidad de numeros indicado por el usuario, obtener su media y moda.*

FASE 4
El programa muestra los resultados

9.- Se muestra un mensaje en la pantalla con el siguiente texto:
"Resultados
Promedio:.....
Mediana: .....
Moda: ..... "

- Dados los diferentes casos, aparecerá el mensaje que corresponda.

En la fase donde se muestran los resultado, ya no hay entradas por parte del usuario, ni análisis del programa, solamente se muestran los resultados de las operaciones que se realizaron para obtener el promedio, mediana, moda

FASE 5
Final del programa

10.- Finaliza el programa mostrando un mensaje con el siguiente texto:
"Cálculos finalizados. Gracias por usar el programa"

11.- Acaba el programa

En la fase donde se muestra que el programa ha terminado, con un mensaje de despedida.

COMENTARIOS

- **Los pasos generales se encuentran escritos con color negro**
- **Los casos particulares que pueden surgir en cada paso se encuentran escritos con color azul**
- **Lo que va a ejecutar el código de este programa se encuentran escritos con verde**
- **Las generalidad se encuentran escritos con color rosa, si no se ha cumplido con lo anterior el programa no continuará**

Estos comentarios son para todo el proceso, es la "simbología" para poder entender mejor el proceso

Ejercicio I — Promedio de "n" cantidad de numeros indicado por el usuario, obtener su media y moda.

9.- Prueba de escritorio

Si el usuario hace uso adecuado del programa

Iteración	Entradas	Salidas
1	-	"Bienvenido, con este programa podrás calcular el promedio, moda y mediana de n números. Da clic para continuar "
2	El programa espera enter	"Ingrese la cantidad de números (n)"
3	(Cantidad de números) **4**	
4	El programa espera enter	"Ingrese el número uno"
5	(número uno) **3**	
6	El programa espera enter	"Ingrese el número dos"
7	(números dos) **5**	
8	El programa espera enter	"Ingrese el número tres"
9	(número tres) **2**	
10	El programa espera enter	"Ingrese el número cuatro"
11	(número cuatro) **5**	
12	El programa analiza los números	
13	Cálculo del promedio (3+5+2+5)/4	Promedio 3.75
14	Ordenar lista: 2, 3.5, 5 n= 4 (par) (3+5)/2	Mediana 4.00
15	Cálculo de la moda	Moda: 5
16	Finalización del programa	"Cálculos finalizados. Gracias por usar el programa"

COMENTARIOS

- **De la fila 1 a 11 el programa recibe las entradas del usuario.**
- **De la fila 12 a 16 que se encuentra sombreada con otro tono, ya no hay entradas, solo el análisis del programa y la salida con los resultados**
- **Los mensajes que el usuario recibe en pantalla están entrecomilladas**
- **Las entradas del usuario están en negritas**
- **Lo que ejecuta o espera el programa está subrayado**
- **Las especificaciones de las entradas están entre paréntesis**

ANÁLISIS DE LA PRUEBA DE ESCRITORIO

- **Este es un ejemplo práctico con un n = 4**
- **Promedio:**
- no hay caso particular, ni error.

- **Mediana:**
- El usuario ingreso un n = 4 que es un número par. Los números ingresados son ordenados de menor a mayor en una lista de donde se tomaron los dos números de en medio y se promediaron para obtener la mediana.

- **Moda:**
- se repite el número 5 con una frecuencia de 2

Si el usuario comete errores

Iteración	Entradas	Salidas
1	-	"Bienvenido, con este programa podrás calcular el promedio, moda y mediana de n números. Da clic para continuar "
2	El programa espera enter	"Ingrese la cantidad de números (n)"
3	(cantidad de números) **-3**	
4	El programa espera enter	"Error: ingrese un número entero positivo mayor que 0"
5	(corrección n) **abc**	"Error: solo se permiten números, intente de nuevo"
6	(corrección n) **3**	
7	El programa espera enter	"Ingrese el número uno"
8	(número 1) **5**	
9	El programa espera enter	"Ingrese el número dos"
10	(número dos) **-2**	
11	El programa espera enter	"Ingrese el número tres"
12	(número tres) **3.5**	
13	El programa analiza los números	
14	Cálculo del promedio	Promedio: 2.16
15	Ordenar lista: -2, 3.5,5	Mediana: 3.50
16	Cálculo de la moda	Moda: "No hay moda, todos los valores aparecen con la misma frecuencia"
17	Finalización del programa	"Cálculos finalizados. Gracias por usar el programa"

COMENTARIOS

- **De la fila 1 a 12 el programa recibe las entradas del usuario.**
- **De la fila 13 a 17 que se encuentra sombreada con otro tono, ya no hay entradas, solo el análisis del programa y la salida con los resultados**
- **Los mensajes que el usuario recibe en pantalla están entrecomilladas**
- **Las entradas del usuario están en negritas**
- **Lo que ejecuta o espera el programa está subrayado**
- **Las especificaciones de las entradas están entre paréntesis**

ANÁLISIS DE LA PRUEBA DE ESCRITORIO

Este es un ejemplo con los casos particulares:

- **Promedio:**
- Si el usuario ingresa un n inválido (n tiene que ser un entero positivo mayor que 0). Sin embargo el usuario ingreso un número negativo, por lo tanto manda un mensaje de error

- Si el usuario ingresa un valor no numérico (solamente se pueden ingresar números) sin embargo el usuario ingreso letras, por lo tanto manda un mensaje de error

- El programa siguió ejecutándose hasta que se introdujeron valores correctos

- **Mediana:**
- El usuario ingreso un n = 3 que es impar, por lo tanto se tomo el valor central de la lista ordenada de menor a mayor

- **Moda:**
- No hay moda ya que todos los valores aparecen con la misma frecuencia. el -2 solo aparece una vez, el 3.5 solo aparece una vez y el 5 también solo aparece una sola vez. Por lo tanto, aparece un mensaje indicando la situación.

Ejercicio II

Ordenar 3 números de menor a mayor, ingresados por el usuario.

Ejercicio II

Ordenar 3 números de menor a mayor, ingresados por el usuario

Desarrollo

1.- Definir el problema

Localizamos de manera clara el enunciado del problema a solucionar.

ORDENAR 3 NÚMEROS DE MENOR A MAYOR, INGRESADOS POR EL USUARIO

2.- Analizar los requerimientos del problema

Del enunciado del problema, identificar de manera concisa solamente lo que requiere el problema.

- El usuario debe ingresar tres números
- Comparar los tres números ingresados.
- Determinar su orden de menor a mayor.

Como se observa en la imagen, solo contamos con tres requerimientos del programa, que es solamente lo que pide el problema. Son los requerimientos esenciales: ingresar solo 3 números, compararlos para poder ordenar los números de menor a mayor, sin embargo, el programador puede proponer más requerimientos para una mejor definición del programa.

3.- Analizar los requerimientos propuestos por el programador

Ya se identifico lo esencial del problema, pero no es suficiente para la solución del problema, por lo que los siguientes requerimientos son propuestos por el programador, ya que el programa no necesariamente lo pide, pero el programador los considera necesarios.

- Se usará lenguaje de programación C++.
- Se podrán usar estructuras condicionales (if-else) o un algoritmo de ordenamiento simple
- Los números pueden ser enteros
- Se debe permitir cualquier orden de ingreso de los números
- Mostrar los tres números ordenados de menor a mayor.
- Se considera la posibilidad de manejar números iguales
- El formato de salida será claro y legible

Ejercicio II *Ordenar 3 números de menor a mayor, ingresados por el usuario*

Por sugerencia del programador:

- Es importante definir el lenguaje de programación en el que se le dará resolución al problema, ya que hay muchos, y unos cuentan con sintaxis más eficientes para ciertas funciones o simplemente por comodidad y facilidad del programador.
- Es claro que en este problema tenemos que comparar números para determinar cual es mayor, menor que otro número, por lo tanto se usarán estructuras condicionales, una opción podría ser (if-else) o un algoritmo de ordenamiento simple
- El problema solo indica que se tienen que comparar tres números, pero no de que tipo, entonces se proponen los enteros.
- Los números deben ingresar en cualquier orden, ya que el programa debe ordenarlos, mas no el usuario
- Mostrar los números acomodados de manera clara
- El programa debería aceptar valores iguales.
- El formato de salida cuando se muestren los resultados debe ser claro para el usuario

4.- Analizar las restricciones del problema

Son restricciones esenciales del problema que aparecen en el enunciado de manera explícita o implícita.

Solo se pueden ingresar tres números
Si el usuario ingresa letras o caracteres especiales, será invalidado
Si el usuario ingresa valores iguales, se deben mostrar en el mismo orden sin afectar la funcionalidad del programa.
El orden de los números ingresados no debe influir en el resultado final.
La salida debe ser en orden ascendente
El programa no debe depender de una cantidad variable de entradas; siempre se deben manejar exactamente tres números .

- El problema menciona que solo tienen que ser tres números
- El programa solo puede comparar y ordenar números, si el usuario ingresa algún otro valor que no sea numérico (letras, caracteres especiales, etc.) será invalidado.
- Si el usuario ingresa valores iguales no debe afectar la funcionalidad del programa.
- El orden en el que se ingresan los números no debe afectar el resultado, porque precisamente esa es la finalidad del programa: ordenar.
- El orden debe ser estrictamente ascendente: menor a mayor

Ejercicio II *Ordenar 3 números de menor a mayor, ingresados por el usuario*

5.- Analizar las restricciones propuestas por el programador

Así como el programador sugiere requerimientos, de estos salen restricciones, que no son mencionados en el enunciado del problema, por lo tanto son propuestas por el programador.

Los valores ingresados deben ser números enteros
Se validará que los valores ingresados sean numéricos antes de procesarlos.
Se implementarán gestiones de errores para prevenir fallos por entradas incorrectas.
Se deben manejar excepciones para evitar que el programa se detenga si el usuario introduce un valor no válido
Si el usuario ingresa un número inválido, el programa pedirá nuevamente la entrada hasta que sea correcta.
En caso de error en la entrada, el programa debe solicitar al usuario que reingrese los datos hasta que sean correctos.
El código debe ser fácilmente modificable y mantenible si llegará a requerir modificaciones
No son validas las entradas vacías
El código debe ser fácilmente modificable y mantenible si llegará a requerir modificaciones

Por sugerencia del programador:

- Un requerimiento del programa es que sean números, pero nunca menciona el tipo de número, por lo tanto se usarán solo enteros.
- Un requerimiento del problema es que sean números, pero es importante validar que cada entrada cumpla con las condiciones mencionadas
- En caso de error en la entrada: (valor no numérico, caracteres especiales, entradas vacías, etc.) el programa no se tiene que cerrar, sino, solicitar que ingresa un valor correcto, y el algoritmo seguirá hasta que lo haga
- No se debe permitir que se cierre el programa, sin indicarle al usuario el error que se esta cometiendo, para que lo pueda corregir
- El código debe ser de fácil acceso, por si en el futuro necesita ajustes o algún cambio.

6.- Entradas

Se deben identificar las entradas que hace el usuario.

Tres números ingresados por el usuario
Números enteros
En cualquier orden
Pueden ser iguales o diferentes

- El usuario solamente tiene que hacer tres entradas, si hace uso correcto del programa: números a ordenar
- El usuario puede cometer errores, por lo tanto se abre brecha a entradas de corrección cuantas veces sea necesario, para evitar que se cierre el programa, sin una ejecución correcta.

Ejercicio II *Ordenar 3 números de menor a mayor, ingresados por el usuario*

7.- Salidas

Se deben identificar las salidas que dará el programa para llevar de la mano al usuario, o bien, darle instrucciones sobre como tiene que ir llevando el programa y así evitar errores.

Salidas Intermedias

Mensaje de bienvenida: "Bienvenido, con este programa podrás ordenar exclusivamente tres números de menor a mayor. Da clic para continuar "

Mensaje pidiendo al usuario ingresar el primer número: "Ingrese el primer número"

Mensaje pidiendo al usuario ingresar el segundo número: "Ingrese el segundo número"

Mensaje pidiendo al usuario ingresar el tercer número: "Ingrese el tercer número"

Mensaje pidiendo al usuario confirmar que el ingreso de los números sea el correcto : "Los números antes de ordenar son:"

Van dándole instrucciones al usuario, indican lo que deberá ingresar al programa

- Comienza indicando que hace el programa con un mensaje de bienvenida
- Pide las tres entradas esenciales que tiene que hacer el usuario para ordenarlos.
- Confirmación de los datos ingresados, ya que en adelante no hay más entradas, sino el programa ya se ejecuta

Salidas de validación y corrección

Si el usuario ingresa un valor no numérico: "Error: solo se permiten números, intente de nuevo"

Si el usuario deja una entrada vacía: "Error: entrada no válida. No se permiten espacios vacíos"

Van dándole instrucciones al usuario, indican lo que deberá hacer si ocurre un error

- En caso de cualquier error, el usuario es informado sobre lo que no esta reconociendo el programa con un mensaje, y se le indica que continue
- Como los números a ordenar se ingresan uno por uno, estos deberán ser validos uno por uno, en dado caso que no sean reconocidos, se le pedirá al usuario que lo intente de nuevo

Salidas de cierre

Mostrar el resultado obtenido: "Números ordenados de menor a mayor:"

Mensaje indicando que el programa terminó correctamente: "Cálculos finalizados. Gracias por usar el programa"

Van dándole instrucciones al usuario cuando se terminan procesos

- Cuando se acaban de ordenar las operaciones aparecerá un mensaje con las respuestas..
- Se le indicara al usuario que el programa ha acabado de ejecutarse, con un mensaje.

Ejercicio II — Ordenar 3 números de menor a mayor, ingresados por el usuario

8.- Proceso

Se analiza meticulosamente como se ejecutará el programa paso a paso.

Las dividimos en fases para distinguir las partes del algoritmo

FASE 1
Inicio del programa

1.- El programa inicia

2.- Se muestra un mensaje en la pantalla con el siguiente texto: "Bienvenido, con este programa podrás ordenar exclusivamente tres números de menor a mayor. Da clic para continuar"

En la fase del inicio del programa se distingue por indicar al usuario de lo que hace con un mensaje de bienvenida

FASE 2
Entrada de datos

3.- Se muestra un mensaje en la pantalla con el siguiente texto: "Ingrese el primer número ".

- Se recibe la entrada del usuario.
- Si el usuario ingresa un valor correcto (número real). Seguir con el paso 4
- Si el usuario ingresa un valor no numérico, mostrar un mensaje con el siguiente texto: "Error: solo se permiten números, intente de nuevo"
- Si el usuario deja una entrada vacía, mostrar un mensaje con el siguiente texto: "Error: entrada no válida. No se permiten espacios vacíos"
- Si se cumple cualquier caso de error, se deberá ingresar el número nuevamente
- Este proceso de validación se repetirá hasta que el usuario ingrese un valor válido.

4.- Solicitar al usuario que ingrese el segundo número, mostrando un mensaje con el siguiente texto: "Ingrese el número dos"

- Se recibe la entrada del usuario.
- Se repite la validación de entradas (anteriormente mencionada)
- Este proceso de validación se repetirá hasta que el usuario ingrese un valor válido.
- Si el usuario ingresa un valor correcto (número real). Seguir con el paso 5

5.- Solicitar al usuario que ingrese el tercer número, mostrando un mensaje con el siguiente texto: "Ingrese el tercer número"

- Se recibe la entrada del usuario.
- Se repite la validación de entradas (anteriormente mencionada)
- Este proceso de validación se repetirá hasta que el usuario ingrese un valor válido.
- Si el usuario ingresa un valor correcto (número real). Seguir con el paso 6

6.- Ya que el usuario ha terminado de ingresar los números, se mostrará un mensaje con el siguiente texto;
"Los números antes de ordenar son:"

Ejercicio II

Ordenar 3 números de menor a mayor, ingresados por el usuario

En la fase de entrada de datos es la etapa donde el usuario indicará los tres números que desea ordenar de forma ascendente , así mismo cada entrada tendrá que ser validada, de no ser reconocida se solicitará intentarlo de nuevo tantas veces sea necesario.

FASE 3
Cálculos y análisis del programa

7.- Se comparan los tres números
- Se aplican condicionales
- Se ordenan los números de forma ascendente
- Si hay números repetidos o iguales, se mantienen en el mismo orden sin afectar el resultado.

En la fase de cálculos y análisis del problema, ya no hay entradas por parte del usuario, solamente el análisis del programa, se harán comparaciones de los números para poder ordenarlos de forma ascendente usando condicionales. Y si se llegase a presentar el caso de que los números se repitan, eso no debe afectar el resultado

FASE 4
El programa muestra los resultados

8.- Se muestra el resultado final con los números ordenados con un mensaje en la pantalla con el siguiente texto:
"Números ordenados de menor a mayor:"

En la fase donde se muestran los resultado, ya no hay entradas por parte del usuario, ni análisis del programa, solamente se muestran los resultados de las comparaciones, para obtenerlos de manera ascendente.

FASE 5
Final del programa

9.- Finaliza el programa mostrando un mensaje con el siguiente texto:
"Cálculos finalizados. Gracias por usar el programa"

10.- Acaba el programa

En la fase donde se muestra que el programa ha terminado, con un mensaje de despedida.

COMENTARIOS
- **Los pasos generales se encuentran escritos con color negro**
- **Los casos particulares que pueden surgir en cada paso se encuentran escritos con color azul**
- **Lo que va a ejecutar el código de este programa se encuentran escritos con verde**
- **Las generalidad se encuentran escritos con color rosa, si no se ha cumplido con lo anterior el programa no continuará**
- **Las reiteraciones de las condiciones de validación que se encuentran en color azul, se encuentran escritos en color anaranjado.**

Estos comentarios son para todo el proceso, es la “simbología“ para poder entender mejor el proceso

Ejercicio II

Ordenar 3 números de menor a mayor, ingresados por el usuario

9.- Prueba de escritorio

Si el usuario hace uso adecuado del programa

Iteración	Entradas	Salidas
1	-	"Bienvenido, con este programa podrás ordenar exclusivamente tres números de menor a mayor. Da clic para continuar "
2	El programa espera enter	"Ingrese el primer número"
3	(primer número) **5**	
4	El programa espera enter	"Ingrese el segundo número"
5	(segundo número) **2**	
6	El programa espera enter	"Ingrese el tercer número"
7	(tercer número) **7**	
8	El programa espera enter	"Los números antes de ordenar son: 5, 2, 7"
9	El programa espera enter	
10	El programa ordena los valores	"Números ordenados de menor a mayor: 2, 5, 7"
11	Finalización del programa	"Cálculos finalizados. Gracias por usar el programa"

COMENTARIOS

- **De la fila 1 a 9 el programa recibe las entradas del usuario.**
- **De la fila 10 a 11 que se encuentra sombreada con otro tono, ya no hay entradas, solo el análisis del programa y la salida con los resultados**
- **Los mensajes que el usuario recibe en pantalla están entrecomilladas**
- **Las entradas del usuario están en negritas**
- **Lo que ejecuta o espera el programa está subrayado**
- **Las especificaciones de las entradas están entre paréntesis**

ANÁLISIS DE LA PRUEBA DE ESCRITORIO

Este es un ejemplo práctico donde se ordenan tres números reales sin casos particulares

Si el usuario comete errores

Iteración	Entradas	Salidas
1	-	"Bienvenido, con este programa podrás ordenar exclusivamente tres números de menor a mayor. Da clic para continuar "
2	El programa espera enter	"Ingrese el primer número"
3	(primer número) **15**	
4	El programa espera enter	"Ingrese el segundo número"
5	(segundo número) **A**	
6	El programa espera enter	"Error: solo se permiten números, intente de nuevo"
7	(corrección segundo número) **4**	
8	El programa espera enter	"Ingrese el tercer número"
9	(tercer número) **100**	
10	El programa espera enter	"Los números antes de ordenar son: 15, 4, 100"
11	El programa espera enter	
12	El programa ordena los valores	"Números ordenados de menor a mayor: 4, 15, 100"
13	Finalización del programa	"Cálculos finalizados. Gracias por usar el programa"

COMENTARIOS

- **De la fila 1 a 11 el programa recibe las entradas del usuario.**
- **De la fila 12 a 13 que se encuentra sombreada con otro tono, ya no hay entradas, solo el análisis del programa y la salida con los resultados**
- **Los mensajes que el usuario recibe en pantalla están entrecomilladas**
- **Las entradas del usuario están en negritas**
- **Lo que ejecuta o espera el programa está subrayado**
- **Las especificaciones de las entradas están entre paréntesis**

ANÁLISIS DE LA PRUEBA DE ESCRITORIO

Este es un ejemplo con los casos particulares:

- Si el usuario ingresa un valor no numérico (solamente se pueden ingresar números) sin embargo el usuario ingreso letras, por lo tanto manda un mensaje de error
- El programa siguió ejecutándose hasta que se introdujeron valores correctos

Conclusiones particulares

Integrante 1: Edgar Iván Padilla Huesca

Esta práctica ha sido de gran utilidad ya que ahora puedo darme cuenta que el desarrollo de un programa requiere más que solo escribir código; es esencial una planificación previa que garantice claridad y eficiencia en la solución del problema. Con la práctica, se evidenció la importancia de definir entradas, procesos, restricciones y salidas antes de la implementación, ya que esto evita muchos errores, y nos permite darnos cuenta de requerimientos o restricciones que no veíamos.

Integrante 2: Jesús Daniel Andrade Mendoza

A lo largo de este laboratorio, he podido profundizar en la generación de algoritmos y el desarrollo de software, comprendiendo su importancia fundamental en la resolución de problemas. He aprendido a analizar los requerimientos de un problema, a diseñar algoritmos eficientes y a considerar diversos aspectos cruciales durante el proceso de desarrollo, como la optimización del código, la experiencia del usuario y el manejo de errores. Además, he explorado temas relevantes como la calidad del software y la importancia de las pruebas exhaustivas. En resumen, este laboratorio ha sido una experiencia enriquecedora que me ha proporcionado conocimientos teóricos y prácticos valiosos para mi futuro como desarrollador de software.

Integrante 3: Fryda Itzel Bastida Reyes

En base a mi experiencia en la práctica puedo decir que la planificación y estructuración previa a la implementación de un programa es un paso esencial en el desarrollo de software, ya que permite una solución más eficiente y organizada. A lo largo de la práctica, se demuestro la importancia de seguir el ciclo de vida del software al diseñar algoritmos para resolver problemas específicos, que en nuestro caso fue el cálculo de medidas estadísticas y la ordenación de números. Al identificar correctamente los conjuntos de entrada y salida, así como los módulos de procesamiento, se logra una mejor comprensión del problema y una implementación más clara y precisa. Además, el uso del seudocódigo como guía facilita la detección de errores antes de escribir el código definitivo, optimizando el tiempo y asegurando un resultado funcional. La practica ha sido de utilidad, y en conclusión la correcta planificación y diseño de algoritmos no solo mejora la calidad del software, sino que también fortalece el pensamiento lógico y la capacidad de resolución de problemas en el ámbito de la programación.

Conclusion General

A lo largo de esta práctica, hemos explorado una variedad de temas cruciales para la generación de algoritmos y el desarrollo de software. Desde la identificación y definición precisa de problemas, hasta la creación de soluciones algorítmicas eficientes.

La práctica de desarrollar algoritmos para resolver problemas específicos, como ordenar números o calcular promedios, nos ha permitido aplicar los conocimientos teóricos y comprender la importancia de cada etapa del proceso. Hemos aprendido a analizar los requerimientos de un problema, identificar las entradas y salidas necesarias, y diseñar algoritmos que resuelvan el problema de manera correcta y eficiente.

Además, hemos discutido la importancia de considerar diversos aspectos durante el desarrollo de software, la optimización del código, la experiencia del usuario, errores y la necesidad de realizar pruebas. Hemos aprendido a analizar protocolos y requerimientos, a comprender la importancia de las unidades de medida.

A través de la discusión de temas, ya sean los objetivos de las prácticas y los ejemplos de protocolos físicos, hemos ampliado nuestra comprensión del contexto en el que se desarrolla el software y la importancia de la comunicación y la colaboración en este proceso.

Referencias

- Flores, A. (2025, 07 de febrero). Fundamentos de programación (Clase).Escuela Nacional de Estudios Superiores Unidad Juriquilla.

- Alpaca Tech. (2024, 6 agosto). Optimiza tu Código en Minutos: 7 Técnicas Básicas [Vídeo]. YouTube. https://www.youtube.com/watch?v=KUiKzclxnUc

- Ciclo de vida del software: todo lo que necesitas saber. (s. f.). Intelequia. https://intelequia.com/es/blog/post/ciclo-de-vida-del-software-todo-lo-que-necesitas-saber

- Hamilton, T. (2024, 13 febrero). Software Quality in Software Engineering. Guru99. https://www.guru99.com/software-quality-software-engineering.html