Guía 2: Algoritmos –
Estructuras Condicionales-
Perez Marchi Diego

Ejercicio 1) Parciales
Dadas las notas de 3 parciales, calcular promedio y decir si promocionó o rinde final.

Análisis

Entradas:

- Nota del primer parcial (parcial1) - tipo: float - La nota obtenida en el primer parcial.
- Nota del segundo parcial (parcial2) - tipo: float - La nota obtenida en el segundo parcial.
- Nota del tercer parcial (parcial3) - tipo: float - La nota obtenida en el tercer parcial.

Relaciones:

- Promedio de notas (promedio) - tipo: float - Calculado como la suma de las tres notas dividido por 3.
- Estado de aprobación - tipo: string - Determinado por un condicional **if** que evalúa si el promedio es mayor o igual a 7.

Salida:

- Estado de aprobación - tipo: string - Indica si el estudiante ha promocionado o si debe rendir final. Puede ser "Promociona" o "Rinde final".

Estrategia:

1. Leer las notas de los tres parciales.
2. Calcular el promedio sumando las tres notas y dividiendo por 3.
3. Utilizar un condicional **if** para determinar el estado de aprobación basado en el promedio:

- Si el promedio es mayor o igual a 7, imprimir "Promociona".
- Si el promedio es menor que 7, imprimir "Rinde final".

4. Mostrar el estado de aprobación.

Ejercicio 2) Par o impar
Se necesita un algoritmo que informe si un número ingresado es PAR o IMPAR mediante un mensaje.

Análisis

Entrada:

- Número (numero) - tipo: entero - El número ingresado por el usuario que se desea verificar si es par o impar.

Relaciones:

- Paridad del número (paridad) - tipo: string - Determinada por un condicional **if** que evalúa si el número es divisible por 2.

Salida:

- Mensaje de paridad - tipo: string - Indica si el número ingresado es par o impar.

Estrategia:

1. Leer el número ingresado por el usuario.
2. Calcular el resto de la división del número entre 2.
3. Utilizar un condicional **if** para determinar si el resto de la división es igual a 0:
    - Si el resto es 0, el número es par. Imprimir "El número es PAR".
    - Si el resto no es 0, el número es impar. Imprimir "El número es IMPAR".

Ejercicio 3) Positivo, negativo o cero
Se desea saber si el número ingresado es positivo, negativo o cero.

Análisis

Entrada:

- Número (numero) - tipo: float o entero - El número ingresado por el usuario que se desea verificar si es positivo, negativo o cero.

Relaciones:

- Signo del número (signo) - tipo: string - Determinado por un condicional **if** que evalúa si el número es mayor que 0, igual a 0 o menor que 0.

Salida:

- Estado del número - tipo: string - Indica si el número ingresado es positivo, negativo o cero.

Estrategia:

1. Leer el número ingresado por el usuario.
2. Utilizar un condicional **if** para determinar el estado del número:

- Si el número es mayor que 0, imprimir "El número es POSITIVO".
- Si el número es igual a 0, imprimir "El número es CERO".
- Si el número es menor que 0, imprimir "El número es NEGATIVO".

Ejercicio 4) Triangulo

Se requiere de un algoritmo que permita determinar si 3 segmentos de recta pueden formar un triángulo.

Nota: En cualquier triangulo el mayor de los lados es menor que la suma de los restantes, o en general, la suma de 2 lados debe ser mayor que el lado restante.

Análisis

Entradas:

- Longitud del primer lado (lado1) - tipo: float o entero - La longitud del primer lado del supuesto triángulo.
- Longitud del segundo lado (lado2) - tipo: float o entero - La longitud del segundo lado del supuesto triángulo.
- Longitud del tercer lado (lado3) - tipo: float o entero - La longitud del tercer lado del supuesto triángulo.

Relaciones:

- Verificación de la condición de existencia de un triángulo - tipo: booleano - Determinada por un condicional **if** que evalúa si la suma de las longitudes de dos lados es mayor que la longitud del tercer lado, para todos los pares de lados.

Salida:

- Mensaje de validez del triángulo - tipo: string - Indica si los tres segmentos de recta pueden formar un triángulo.

Estrategia:

- Leer las longitudes de los tres lados del supuesto triángulo.
- Utilizar un condicional **if** para verificar la condición de existencia de un triángulo:
    1. Verificar si la suma de las longitudes de dos lados es mayor que la longitud del tercer lado, para todos los pares de lados.

- Si la condición se cumple para todos los pares de lados, imprimir "Se puede formar un triángulo".
- Si la condición no se cumple para al menos un par de lados, imprimir "No se puede formar un triángulo".

Ejercicio 5) Números de libreta
Se ingresa el nombre y Nro de libreta de 3 alumnos. Muestre la lista ordenada por Nro de libreta.

Análisis

Entradas:

- Nombre del primer alumno (nombre1) - tipo: string - El nombre del primer alumno.
- Número de libreta del primer alumno (libreta1) - tipo: string o entero - El número de libreta del primer alumno.
- Nombre del segundo alumno (nombre2) - tipo: string - El nombre del segundo alumno.
- Número de libreta del segundo alumno (libreta2) - tipo: string o entero - El número de libreta del segundo alumno.
- Nombre del tercer alumno (nombre3) - tipo: string - El nombre del tercer alumno.
- Número de libreta del tercer alumno (libreta3) - tipo: string o entero - El número de libreta del tercer alumno.

Relaciones:

- Comparación de números de libreta - tipo: condicionales - Se utilizarán condicionales para comparar los números de libreta de los alumnos y determinar su orden.

Salidas:

- Mensajes de ordenación por número de libreta - tipo: string - Mensajes que indican el orden de los alumnos por número de libreta.

Estrategia:

- Leer el número de libreta de cada alumno.
- Utilizar condicionales para comparar los números de libreta y determinar el orden de los alumnos.
- Imprimir los nombres de los alumnos en el orden correcto basado en la comparación de sus números de libreta.

Ejercicio 6) Rectángulo
Realice un algoritmo que, tomando como datos la base y la altura de un rectángulo, informe si este es horizontal o vertical. Sin dejar de considerar el caso particular del cuadrado. Finalmente calcule el área de la figura.

Análisis

Entradas:

- Base del rectángulo (base) - tipo: float o entero - La longitud de la base del rectángulo.
- Altura del rectángulo (altura) - tipo: float o entero - La longitud de la altura del rectángulo.

Relaciones:

- Orientación del rectángulo (orientacion) - tipo: string - Determinada por un condicional **if** que compara la base y la altura del rectángulo para determinar si es horizontal, vertical o un cuadrado.
- Área del rectángulo (area) - tipo: float - Calculada multiplicando la base por la altura del rectángulo.

Salidas:

- Orientación del rectángulo - tipo: string - Indica si el rectángulo es horizontal, vertical o un cuadrado.
- Área del rectángulo - tipo: float - Indica el área del rectángulo calculada a partir de su base y altura.

Estrategia:

- Leer la base y la altura del rectángulo.
- Utilizar un condicional **if** para determinar la orientación del rectángulo:
    1. Si la base es mayor que la altura, el rectángulo es horizontal.
    2. Si la altura es mayor que la base, el rectángulo es vertical.
    3. Si la base es igual a la altura, el rectángulo es un cuadrado.
- Calcular el área del rectángulo multiplicando la base por la altura.
- Imprimir la orientación del rectángulo y el área calculada.

Ejercicio 7) Mayor valor
Realice un algoritmo que pida 5 valores al usuario y luego informe cual es el mayor de los ingresados.
Restricción: la aplicación solo puede tener 2 variables.

Análisis

Entradas:

- Valor 1 (valor1) - tipo: entero o float - El primer valor ingresado por el usuario.
- Valor 2 (valor2) - tipo: entero o float - El segundo valor ingresado por el usuario.

Variables:

- Mayor valor encontrado (mayor) - tipo: entero o float - Variable que almacena el mayor valor encontrado hasta el momento.
- Valor actual (valor_actual) - tipo: entero o float - Variable temporal que almacena el valor actual que está siendo evaluado.

Estrategia:

- Leer el primer valor ingresado por el usuario y asignarlo a la variable "mayor".
- Leer el segundo valor ingresado por el usuario y asignarlo a la variable "valor_actual".
- Utilizar un condicional **if** para comparar "valor_actual" con "mayor":
  1. Si "valor_actual" es mayor que "mayor", actualizar "mayor" con el valor de valor actual.
- Repetir los pasos 2 y 3 para los siguientes tres valores ingresados por el usuario, actualizando "mayor" si es necesario.
- Al finalizar la comparación de los cinco valores, la variable "mayor" contendrá el mayor valor de todos.
- Imprimir el valor almacenado en "mayor".

Ejercicio 8) Orden que ocurrió el menor
Realice un algoritmo que pida 5 valores al usuario y luego informe cual es el número de orden en que se ingresó el menor de ellos.
Restricción: la aplicación solo puede tener 3 variables.

Análisis

Entradas:

- Valor 1 (valor1) - tipo: entero o float - El primer valor ingresado por el usuario.
- Valor 2 (valor2) - tipo: entero o float - El segundo valor ingresado por el usuario.
- Valor 3 (valor3) - tipo: entero o float - El tercer valor ingresado por el usuario.

Variables:

- Menor valor encontrado (menor) - tipo: entero o float - Variable que almacena el menor valor encontrado hasta el momento.
- Orden del menor valor (orden_menor) - tipo: entero - Variable que almacena el número de orden en que se ingresó el menor valor.
- Número de orden actual (orden_actual) - tipo: entero - Variable que lleva la cuenta del número de orden actual.

Salida:

- Orden del menor valor - tipo: entero - Indica en qué orden se ingresó el menor valor.

Estrategia:

- Inicializar la variable "menor" con un valor inicial grande (por ejemplo, infinito positivo) y "orden_menor" con 0.
- Leer el primer valor ingresado por el usuario y asignarlo a la variable "menor".
- Inicializar la variable "orden_actual" con 1.
- Leer el segundo valor ingresado por el usuario y asignarlo a la variable "valor_actual".
- Utilizar un condicional **if** para comparar "valor_actual" con "menor":
    1. Si "valor_actual" es menor que "menor", actualizar "menor" con el valor de "valor_actual" y "orden_menor" con el número de orden actual.
- Incrementar el número de orden actual en 1.
- Repetir los pasos 4-6 para el tercer valor ingresado.
- Imprimir el valor almacenado en "orden_menor".

Ejercicio 9) Tenis
Ingresado el nombre de los jugadores y el resultado de cada set (3) de un partido de tenis, informe en pantalla cual es el ganador.
Ejemplo: Nadal, Del Potro: 7,5,4,6,6,2
Ganador Nadal

Análisis

Entradas:

- Nombre del jugador 1 (jugador1) - tipo: cadena de caracteres - El nombre del primer jugador.
- Nombre del jugador 2 (jugador2) - tipo: cadena de caracteres - El nombre del segundo jugador.
- Resultado del set 1 (set1) - tipo: entero - El resultado del primer set.
- Resultado del set 2 (set2) - tipo: entero - El resultado del segundo set.
- Resultado del set 3 (set3) - tipo: entero - El resultado del tercer set.

Variables:

- Puntos del jugador 1 (puntos_jugador1) - tipo: entero - Variable que almacena el total de puntos del primer jugador.
- Puntos del jugador 2 (puntos_jugador2) - tipo: entero - Variable que almacena el total de puntos del segundo jugador.

Salida:

- Ganador - tipo: cadena de caracteres - El nombre del jugador ganador.

Estrategia:

- Calcular los puntos totales de cada jugador sumando los resultados de los tres sets.
- Utilizar un condicional **if** para comparar los puntos de los jugadores y determinar quién tiene más puntos.
- Si los puntos del jugador 1 son mayores que los del jugador 2, el ganador es el jugador 1. De lo contrario, el ganador es el jugador 2.
- Imprimir el nombre del jugador ganador.

Ejercicio 10) Año bisiesto!
Implemente un algoritmo que permita determinar si un año es bisiesto o no.
Un año es bisiesto si es múltiplo de 4 (por ejemplo 1984). Los años múltiplos de 100 no son bisiestos, salvo si ellos son también múltiplos de 400 (2000 es bisiesto, pero; 1800 no lo es).

Entradas:

- Año (year) - tipo: entero - El año que se quiere verificar si es bisiesto o no.

Salida:

- Mensaje - tipo: cadena de caracteres - Un mensaje que indica si el año es bisiesto o no.

Estrategia:

- Utilizar condicionales **if** para verificar las reglas de los años bisiestos:
    1. Si el año es divisible por 4 pero no por 100, o si es divisible por 400, entonces es bisiesto. Devolver un mensaje indicando que el año es bisiesto.
    2. De lo contrario, devolver un mensaje indicando que el año no es bisiesto.

Ejercicio 11) Día del mes

Realice un algoritmo que permita ingresar el número del mes y determine cuantos días tiene. Para el caso de Febrero, el algoritmo deberá indicar que no cuenta con la información necesaria para dar la respuesta.

Análisis

Entradas:

- Número del mes (numero_mes) - tipo: entero - El número del mes que se quiere evaluar.

Salida:

- Número de días (dias) - tipo: entero - El número de días que tiene el mes.

1. En el caso de febrero, si se desconoce si es un año bisiesto o no, se mostrará un mensaje indicando que no se tiene la información necesaria.

Estrategia:

- Utilizar condicionales **if** para determinar el número de días en función del número del mes:

1. Para los meses con 31 días (enero, marzo, mayo, julio, agosto, octubre, diciembre), devolver 31.
2. Para los meses con 30 días (abril, junio, septiembre, noviembre), devolver 30.
3. Para febrero, devolver un mensaje indicando que no se tiene la información necesaria para determinar el número de días.

- Devolver el número de días o el mensaje correspondiente.

Ejercicio 12) Ruleta

Se desea simular parte de un juego de ruleta donde el usuario ingresa un número entre 0 y 36 (el sistema debe verificarlo) y luego informar si es:

a. 0 (banca gana)
b. Mayor o Menor
c. 1ra, 2da o 3ra Docena
d. 1ra, 2da o 3ra Columna

Entradas:

- Número seleccionado por el usuario (numero) - tipo: entero - El número seleccionado por el usuario en la ruleta.

Relaciones:

Rango de números:

- Rango 0 a 36: Los números válidos en la ruleta.
- Categorías de juego:
  Banca gana: Si el número es 0.
  Mayor o Menor: Si el número es mayor que 0 y menor o igual que 18, o mayor que 18 y menor o igual que 36.
    - Docenas: División del rango en tres grupos de 12 números.
    - Columnas: División del rango en tres grupos de 12 números.

Salida:

- Resultado del juego - tipo: cadena de caracteres - Un mensaje que indica el resultado del juego para el número seleccionado.

Estrategia:

1. Verificar si el número ingresado por el usuario está en el rango válido de 0 a 36.
2. Utilizar condicionales **if** para determinar el resultado del juego en función del número seleccionado:
    - Si el número es 0, informar que la banca gana.
    - Si el número está en el rango de 1 a 18, informar que es Menor.
    - Si el número está en el rango de 19 a 36, informar que es Mayor.
    - Si el número está en el rango de la 1ra, 2da o 3ra docena, informar la docena correspondiente.
    - Si el número está en el rango de la 1ra, 2da o 3ra columna, informar la columna correspondiente.
3. Devolver el resultado del juego como un mensaje.

Ejercicio 13) Azar Modifique el algoritmo anterior utilizando la función Azar() para generar un número aleatorio. ¿Qué modificaciones debe realizar?

- Eliminar la entrada del número seleccionado por el usuario, ya que ahora utilizaremos la función **Azar()** para obtener el número aleatorio.
- Llamar a la función **Azar()** para generar el número aleatorio dentro del rango válido de 0 a 36.
- Utilizar el número aleatorio generado para determinar el resultado del juego de la misma manera que antes, mediante condicionales **if**

Análisis

Salida:

- Resultado del juego - tipo: cadena de caracteres - Un mensaje que indica el resultado del juego para el número aleatorio generado.

Estrategia:

1. Generar un número aleatorio dentro del rango válido de 0 a 36 utilizando la función **Azar()**.
2. Utilizar condicionales **if** para determinar el resultado del juego en función del número aleatorio generado:
    - Si el número es 0, informar que la banca gana.
    - Si el número está en el rango de 1 a 18, informar que es Menor.
    - Si el número está en el rango de 19 a 36, informar que es Mayor.
    - Si el número está en el rango de la 1ra, 2da o 3ra docena, informar la docena correspondiente.
    - Si el número está en el rango de la 1ra, 2da o 3ra columna, informar la columna correspondiente.
3. Devolver el resultado del juego como un mensaje.