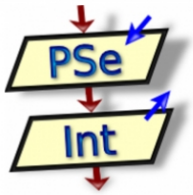
CURSO DE PROGRAMACIÓN FULL STACK

ESTRUCTURAS DE CONTROL CON PSEINT



EGG

GUÍA DE ESTRUCTURAS DE CONTROL

Estructuras de Control

Las Estructuras de Control determinan el orden en que deben ejecutarse las instrucciones de un algoritmo, es decir, si serán recorridas una después de la otra (estructuras secuenciales), si habrá que tomar decisiones sobre si ejecutar o no alguna acción (estructuras selectivas o de decisión) o si habrá que realizar repeticiones (estructuras repetitivas).

Estructura Secuencial

Es la estructura en donde una acción (instrucción) *sigue a otra de manera secuencial*. Las tareas se dan de tal forma que *la salida de una es la entrada de la que sigue* y así en lo sucesivo hasta cumplir con todo el proceso. Esta estructura de control es la más simple, permite que las instrucciones que la constituyen se ejecuten una tras otra en el orden en que se listan.

Estructuras Selectivas

Estas estructuras de control son de gran utilidad para cuando el algoritmo a desarrollar requiera una descripción más complicada que una lista sencilla de instrucciones. Este es el caso cuando existe un número de posibles alternativas que resultan de la evaluación de una determinada condición. Este tipo de estructuras son utilizadas para tomar decisiones lógicas, es por esto que también se denominan estructuras de decisión o selectivas.

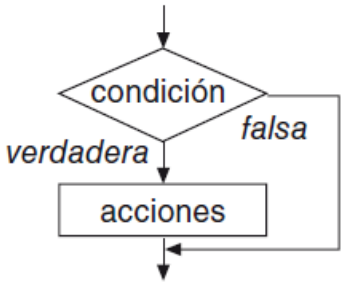
En estas estructuras, se realiza una evaluación de una condición y de acuerdo al resultado, el algoritmo realiza una determinada acción. Las condiciones son especificadas utilizando expresiones lógicas.

Las estructuras selectivas/alternativas pueden ser:

- Simples
- Dobles
- Múltiples

Condición Simple

La estructura alternativa simple *si-entonces* lleva a cabo una acción siempre y cuando se cumpla una determinada *condición*.



La selección si-entonces evalúa la condición y luego:

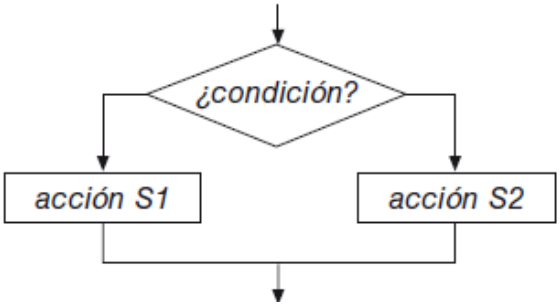
- *Si la condición es verdadera, ejecuta el bloque de acciones*
- *Si la condición es falsa, no ejecuta nada.*

Pseudocódigo en PSeInt:

```
Si expresión lógica Entonces
      acciones
Fin Si
```

Condición Doble

La estructura anterior es muy limitada y normalmente se necesitará una estructura que permita elegir entre dos opciones o alternativas posibles, en función del cumplimiento o no de una determinada condición. Si la condición es verdadera, se ejecuta la acción S1 y, si es falsa, se ejecuta la acción S2.



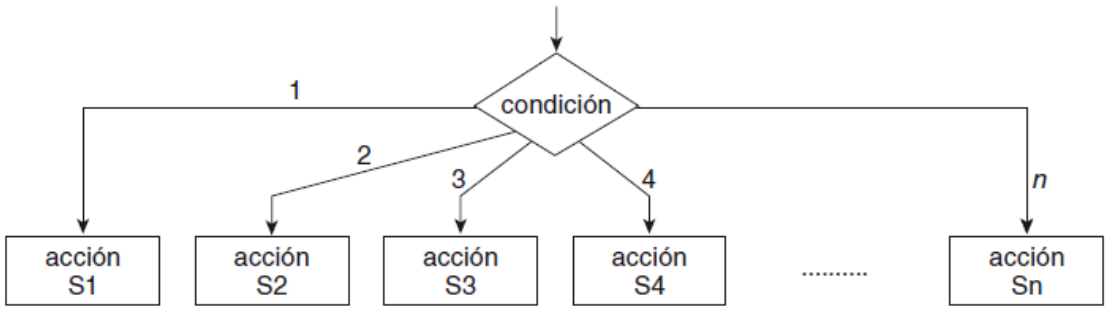
Pseudocódigo en PSeInt:

```
Si expresión lógica Entonces
      acciones_por_verdadero
Sino
      acciones_por_falso
Fin Si
```

Condición Múltiple

Se utiliza cuando existen más de dos alternativas para elegir. En esta estructura, se evalúa una condición o expresión que puede tomar n valores. Según el valor que la expresión tenga en cada momento se ejecutan las acciones correspondientes al valor.

La estructura de decisión múltiple evaluará una expresión que podrá tomar n valores distintos, 1, 2, 3, 4, ..., n. Según que elija uno de estos valores en la condición, se realizará una de las n acciones, o lo que es igual, el flujo del algoritmo seguirá un determinado camino entre los n posibles.



Pseudocódigo en PSeInt:

```
Segun variable_numerica Hacer
      opcion_1:
            secuencia_de_acciones_1
      opcion_2:
            secuencia_de_acciones_2
      opcion_3:
            secuencia_de_acciones_3
      De Otro Modo:
            secuencia_de_acciones_dom
Fin Segun
```

NOTA:

Cuando el valor de la variable que se evalúa no coincide con ninguno de los valores que se evalúa, entonces se ejecutan las acciones dentro del bloque "De Otro Modo" (secuencia_de_acciones_dom), el cual equivale a realizar un "Sino" dentro de las estructuras condicionales.

Este problema, se podría resolver por estructuras alternativas simples o dobles, anidadas o en cascada; sin embargo, este método si el número de alternativas es grande puede plantear serios problemas de escritura del algoritmo y naturalmente de legibilidad.

Condicionales Anidados o en Cascada

Es posible también utilizar la instrucción *Si* para diseñar estructuras de selección que contengan más de dos alternativas. Por ejemplo, una estructura *Si-entonces* puede contener otra estructura *Si-entonces*, y esta estructura *Si-entonces* puede contener otra, y así sucesivamente cualquier número de veces; a su vez, dentro de cada estructura pueden existir diferentes acciones.

Pseudocódigo en PSeInt:

```
Si expresion_logica1 Entonces
      acciones_por_verdadero1
Sino
      Si expresion_logica2 Entonces
            acciones_por_verdadero2
      Sino
            Si expresion_logica4 Entonces
                  acciones_por_verdadero3
            Sino
                  acciones_por_falso
            Fin Si
      Fin Si
Fin Si
```

Estructuras Repetitivas

Durante el proceso de creación de programas, es muy común, encontrarse con que una operación o conjunto de operaciones deben repetirse muchas veces. Para ello es importante conocer las estructuras de algoritmos que permiten repetir una o varias acciones, un número determinado de veces.

Las estructuras que repiten una secuencia de instrucciones un número determinado de veces se denominan *bucles*, y se denomina *iteración* al hecho de repetir la ejecución de una secuencia de acciones.

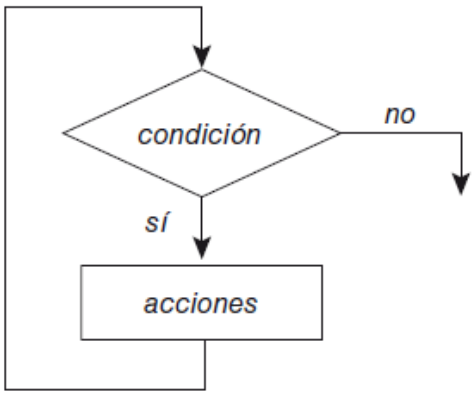
Todo bucle tiene que llevar asociada una condición, que es la que va a determinar cuándo se repite el bucle y cuando deja de repetirse.

Hay distintos tipos de bucles:

- *Mientras, en inglés: While*
- *Hacer Mientras, en inglés: Do While.*
- *Para, en inglés: For*

Estructura Mientras

Esta estructura repetitiva *Mientras*, es en la que el cuerpo del bucle se repite siempre que se cumpla una determinada *condición*. Cuando se ejecuta la instrucción mientras, la primera cosa que sucede es que se evalúa la condición (una expresión lógica). Si se evalúa *falsa*, no se toma ninguna acción y el programa prosigue en la siguiente instrucción del bucle. Si la expresión lógica es *verdadera*, entonces se ejecuta el cuerpo del bucle, después de lo cual se evalúa de nuevo la expresión lógica. Este proceso se repite una y otra vez mientras la expresión lógica (condición) sea verdadera.



Pseudocódigo en PSeInt:

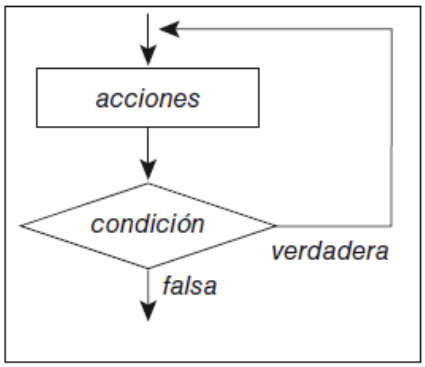
```
Mientras expresion_logica Hacer
      secuencia_de_acciones
Fin Mientras
```

Regla práctica

Las pruebas o test en las expresiones lógica es conveniente que sean mayor o menor que en lugar de pruebas de igualdad o desigualdad. En el caso de la codificación en un lenguaje de programación, esta regla debe seguirse rígidamente en el caso de comparación de números reales, ya que como esos valores se almacenan en cantidades aproximadas las comparaciones de igualdad de valores reales normalmente plantean problemas. Siempre que realice comparaciones de números reales use las relaciones <, <=, > o >=.

Estructura Hacer-Mientras

Esta estructura es muy similar a la anterior, sólo que a diferencia del *Mientras* el contenido del bucle *Hacer-Mientras* se ejecuta siempre al menos una vez, ya que la evaluación de la condición lógica se encuentra al final del bucle. De esta forma garantizamos que las acciones dentro de este bucle sean llevadas a cabo al menos una vez, incluso aunque la expresión lógica sea falsa.



Pseudocódigo en PSeInt:

```
Hacer
      secuencia_de_acciones
Mientras Que expresion_logica
```

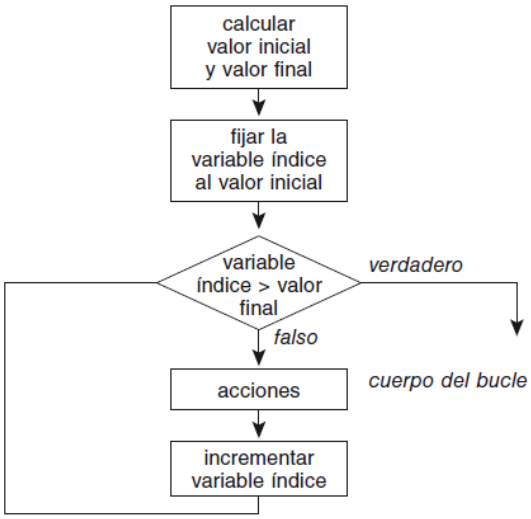
Regla práctica

El bucle *hacer-mientras* se termina de ejecutar cuando el valor de la condición es falsa. La elección entre un bucle mientras y un bucle hacer-mientras depende del problema de cómputo a resolver. En la mayoría de los casos, la condición de entrada del bucle mientras es la elección correcta. Por ejemplo, si el bucle se utiliza para recorrer una lista de números (o una lista de cualquier tipo de objetos), la lista puede estar vacía, en cuyo caso las sentencias del bucle nunca se ejecutarán. Si se aplica un bucle hacer-mientras nos conduce a un código de errores.

Estructura Para

La estructura *Para* es un poco más compleja que las anteriores y nos permite ejecutar un conjunto de acciones para cada elemento de una lista, o para cada paso de un conjunto de elementos. Su implementación depende del lenguaje de programación, pero en términos generales podemos identificar tres componentes: la *inicialización*, la *condición* de corte y el *incremento*.

La estructura *Para* comienza con un valor inicial de la variable índice y las acciones especificadas se ejecutan, *a menos que el valor inicial sea mayor que el valor final*. La variable índice se incrementa en uno y si este nuevo valor no excede al final, se ejecutan de nuevo las acciones. Por consiguiente, las acciones específicas en el bucle se ejecutan para cada valor de la variable índice desde el valor inicial hasta el valor final con el incremento de uno en uno.



Pseudocódigo en PSeInt:

```
Para variable_numerica<-valor_inicial Hasta valor_final Con Paso paso Hacer
      secuencia_de_acciones
Fin Para
```

El incremento de la variable índice (variable_numerica) siempre es 1 si no se indica expresamente lo contrario en el valor del paso. Dependiendo del tipo de lenguaje, es posible que el incremento sea distinto de uno, positivo o negativo. La variable índice o de control (variable_numerica) normalmente será de tipo entero y es normal emplear como nombres las letras I, J, K.

Si el valor_inicial de la variable índice es menor que el valor_final, los incrementos, es decir los pasos, deben ser positivos, ya que en caso contrario la secuencia de acciones no se ejecutaría. De igual modo, si el valor_inicial es mayor que el valor_final, el paso debe ser en este caso negativo, es decir, decremento.

PREGUNTAS DE APRENDIZAJE

1) Un condicional es:

a) Una sentencia que permite decidir si se ejecuta o no un bloque de código
b) Una sentencia que ejecuta otra sentencia que a su vez ejecuta la primera sentencia
c) Una sentencia que permite ejecutar un bloque de código varias veces
d) Ninguna de las anteriores

2) En una expresión condicional se pueden utilizar:

a) Operadores lógicos y de comparación simultáneamente
b) Operadores lógicos únicamente
c) Operadores de comparación únicamente
d) Operadores lógicos o de comparación, pero nunca ambos simultáneamente

3) Estructura que se aplica en problemas que luego de tomar una decisión y marcar el camino correspondiente a seguir, es necesario tomar otra decisión:

a) Estructura selectiva doble "si entonces /sino"
b) Estructura simple " si entonces"
c) Estructura selectiva múltiple "si múltiple"
d) Estructura selectiva en cascada (anidadas)

4) Dado el siguiente pseudocódigo, en el cual la sentencia leer permite al usuario introducir un valor entero, ¿cuál será el valor final de la variable "i"?

```
Algoritmo valorFinal
   Definir i, n como entero
   i = 0
   leer n
   Mientras i < n hacer
         i = i + 1
   Fin Mientras
   escribir “El valor de i es”, i
FinAlgoritmo
```

a) 0 si el valor introducido es igual o menor que 0; el valor introducido menos uno en cualquier otro caso
b) 0 si el valor introducido es igual o menor que 0; el valor introducido en cualquier otro caso
c) 0 si el valor introducido es igual o menor que 0; el valor introducido más uno en cualquier otro caso
d) Ninguna de las anteriores

5) Un bucle es:

a) Una sentencia que permite decidir si se ejecuta o no se ejecuta una sola vez un bloque de código
b) Una sentencia que ejecuta otra sentencia que a su vez ejecuta la primera sentencia
c) Una sentencia que permite ejecutar un bloque de código varias veces hasta que se cumpla (o deje de cumplirse) la condición asignada al bucle
d) Ninguna de las anteriores

6) ¿Qué diferencia hay entre un bucle *mientras* y un bucle *para*?

a) El bucle *para* puede no llegar a ejecutarse nunca pero el bucle *mientras* siempre se ejecuta al menos una vez
b) El bucle *para* se ejecuta un número determinado de veces y el mientras un número indeterminado de veces.
c) El bucle *para* no puede convertirse en un bucle *mientras*, pero sí al contrario
d) El bucle *mientras* permite su inicialización, pero el bucle *para* no

7) De acuerdo a la sintaxis del bucle mientras, señalar cuál es la afirmación falsa:

```
Mientras condición hacer
     <sentencias>
Fin Mientras
```

a) La condición debe ser una expresión lógica
b) La condición se evalúa cada vez que se ejecuta una nueva iteración del bucle
c) Si la condición es falsa, no se ejecuta el bloque de sentencias
d) Ninguna de las anteriores es falsa

8) Si, según, mientras, hacer-mientras y para son:

a) Funciones de acceso a datos
b) Instrucciones de acceso a datos
c) Sentencias de control
d) Tipos de datos

9) ¿Qué diferencia hay entre un bucle mientras y un hacer-mientras?

a) El bucle *hacer-mientra*s puede no llegar a ejecutarse nunca pero el bucle *mientras* siempre se ejecuta al menos una vez
b) El bucle *hacer-mientras* se ejecuta un número determinado de veces y el *mientras* un número indeterminado de veces.
c) El bucle *mientras* puede no llegar a ejecutarse nunca pero *el hacer-mientras* siempre se ejecuta al menos una vez
d) El bucle *mientras* permite utilizar contadores y el *hacer-mientras* no lo permite

10) De acuerdo a la sintaxis del bucle *hacer-mientras*, señalar cuál es la afirmación falsa:

```
Hacer
   <sentencias>
Mientras Que condición
```

a) Si condición = verdadero, entonces el bucle se sigue ejecutando
b) Aunque se cumpla condición = falso, el bucle se llega a ejecutar alguna vez
c) Si condición = falso, el bucle no se llega a ejecutar nunca
d) Ninguna de las anteriores es falsa

11) La estructura repetitiva *Para* se caracteriza por:

a) No conocer el número de repeticiones
b) No se puede repetir más de 10000 veces
c) Conocer de antemano el número de repeticiones
d) Ninguna de las anteriores

EJERCICIOS DE APRENDIZAJE

Para cada uno de los siguientes ejercicios realizar el análisis del problema e indicar cuáles son los datos de entrada y cuáles son los datos de salida. Escribir luego el programa en PSeInt.

VER VIDEO: Condicional Simple

1. Escriba un programa en donde se pida la edad del usuario. Si el usuario es mayor de edad se debe mostrar un mensaje por pantalla indicándolo.

2. Un hombre desea saber si su sueldo es mayor al sueldo mínimo, el programa le pedirá al usuario su sueldo actual y el sueldo mínimo. Si el sueldo el mayor se debe mostrar un mensaje por pantalla indicándolo.

VER VIDEO: Condicional Doble

3. Realizar un programa que pida un número al usuario. Si el número es mayor que 100 se deberá de imprimir en pantalla "Es Mayor", y en caso contrario se deberá imprimir "Es Menor".

4. Realiza un programa que sólo permita introducir los caracteres 'S' y 'N'. Si el usuario ingresa alguno de esos dos caracteres se deberá de imprimir un mensaje por pantalla que diga "CORRECTO", en caso contrario, se deberá imprimir "INCORRECTO".

5. Realizar un programa que pida un numero y determine si ese numero es par o impar. Mostrar en pantalla un mensaje que indique si el numero es par o impar. Nota: investigar la función mod de Pseint.

6. Realizar un programa que pida tres notas y determine si un alumno aprueba o reprueba un curso, sabiendo que aprobará el curso si su promedio de tres calificaciones es mayor o igual a 70; y reprueba en caso contrario.

7. Realizar un programa que pida introducir solo frases o palabras de 6 de largo. Si el usuario ingresa una frase o palabra de 6 de largo se deberá de imprimir un mensaje por pantalla que diga "CORRECTO", en caso contrario, se deberá imprimir "INCORRECTO". Nota: investigar la función Longitud() de Pseint.

8. Realizar un programa que pida una frase o palabra y si la frase o palabra es de 4 caracteres de largo, el programa le sumará un signo de exclamación al final, y si no es de 4 caracteres el programa le sumará un signo de interrogación al final. El programa mostrará después la frase final. Nota: investigar la función Longitud() y Concatenar() de Pseint.

9. Una tienda ofrece para los meses de septiembre, octubre y noviembre un descuento de 500 pesos sobre el total de la compra que realiza un cliente. Solicitar al usuario que ingrese un mes y el importe de la compra. El programa debe calcular cuál es el monto total que se debe cobrar al cliente e imprimirlo por pantalla.

10. Solicitar al usuario que ingrese dos números enteros y determinar si ambos son pares o impares. Mostrar en pantalla un mensaje que indique "Ambos números son pares" siempre y cuando cumplan con la condición. En caso contrario se deberá imprimir el siguiente mensaje "Los números no son pares, o uno de ellos no es par".
    Nota: investigar la función mod de Pseint.

11. Escriba un programa que pida 3 notas y valide si esas notas están entre 1 y 10. Si están entre esos parámetros se debe poner en verdadero una variable de tipo lógico y si no ponerla en falso. Al final el programa se debe decir si las 3 notas son correctas usando la variable de tipo lógico.

12. Escriba un programa que pida una frase o palabra y valide si la primera letra de esa frase es una 'A'. Si la primera letra es una 'A', se deberá de imprimir un mensaje por pantalla que diga "CORRECTO", en caso contrario, se deberá imprimir "INCORRECTO". Nota: investigar la función Subcadena de Pseint.

13. Continuando el ejercicio anterior, ahora se pedirá una frase o palabra y se validara si la primera letra de la frase es igual a la ultima letra de la frase. Se deberá de imprimir un mensaje por pantalla que diga "CORRECTO", en caso contrario, se deberá imprimir "INCORRECTO".

14. La empresa "Te llevo a todos lados" está destinada al alquiler de autos y tiene un sistema de tarifa que consiste en cobrar el alquiler por hora. Si el cliente devuelve el auto dentro de las 2 horas de uso el valor que corresponde pagar es de $400 pesos y la nafta va de regalo.
    Cuando el cliente regresa a la empresa pasadas las 2 horas, se ingresan la cantidad de litros de nafta gastados y el tiempo transcurrido en horas. Luego, se le cobra 40 pesos por litro de nafta gastado, y la hora se fracciona en minutos, cobrando un total de $5,20 el minuto de uso. Realice un programa que permita registrar esa información y el total a pagar por el cliente.

VER VIDEO: Condicional Múltiple

15. Solicitar al usuario que ingrese un valor entre 1 y 7. El programa debe mostrar por pantalla un mensaje que indique a qué día de la semana corresponde. Considere que el número 1 corresponde al día "Lunes", y así sucesivamente.

16. Construir un programa que simule un menú de opciones para realizar las cuatro operaciones aritméticas básicas (suma, resta, multiplicación y división) con dos valores numéricos enteros. El usuario, además, debe especificar la operación con el primer carácter de la operación que desea realizar: 'S' o 's' para la suma, 'R' o 'r' para la resta, 'M' o 'm' para la multiplicación y 'D' o 'd' para la división.

17. Leer tres números que denoten una fecha (día, mes, año) y comprobar que sea una fecha válida. Si la fecha no es válida escribir un mensaje de error por pantalla. Si la fecha es válida se debe imprimir la fecha cambiando el número que representa el mes por su nombre. Por ejemplo: si se introduce 1 2 2006, se deberá imprimir "1 de febrero de 2006".

VER VIDEO: Condicionales Anidados (primera mitad del video)

18. Realizar un programa que, dado un número entero, visualice en pantalla si es par o impar. En caso de que el valor ingresado sea 0, se debe mostrar "el número no es par ni impar" (para que un número sea par, se debe dividir entre dos y su resto debe ser igual a 0). Nota: investigar la función mod de PSeInt

19. Realice un programa que, dado un año, nos diga si es bisiesto o no. Un año es bisiesto bajo las siguientes condiciones: Un año divisible por 4 es bisiesto y no debe ser divisible por 100. Si un año es divisible por 100 y además es divisible por 400, también resulta bisiesto. Nota: recuerde la función mod de PSeInt

20. Escriba un programa para obtener el grado de eficiencia de un operario de una fábrica de tornillos, de acuerdo a las siguientes dos condiciones que se le imponen para un período de prueba:

    - Producir *menos* de 200 tornillos defectuosos.
    - Producir *más* de 10000 tornillos sin defectos.

    El grado de eficiencia se determina de la siguiente manera:

    - Si no cumple ninguna de las condiciones, grado 5.
    - Si sólo cumple la primera condición, grado 6.
    - Si sólo cumple la segunda condición, grado 7.
    - Si cumple las dos condiciones, grado 8

    Nota: para trabajar este ejercicio de manera prolija, ir probando cada inciso que pide el ejercicio. No hacer todos al mismo tiempo y después probar.

21. Una empresa tiene personal de distintas áreas con distintas condiciones de contratación y formas de pago. El departamento de contabilidad necesita calcular los sueldos semanales (lunes a viernes) en base a las 3 modalidades de sueldo:

    a) comisión
    b) salario fijo + comisión, y
    c) salario fijo.

    a) Para la modalidad salario *por comisión* se debe ingresar el monto total de las ventas realizadas en la semana, y el 40% de ese monto total corresponde al salario del empleado.

    b) Para la condición de *salario fijo + comisión*, se debe ingresar el valor que se paga por hora, la cantidad de horas trabajadas semanalmente y el monto total de las ventas en esa semana. En este tipo de contrato las horas extras no están contempladas y se fija como máximo 40 horas por semana. La comisión por las ventas se calcula como 25% del valor de venta total.

    c) Finalmente, para la modalidad de *salario fijo* se debe ingresar el valor que se paga por hora y la cantidad de horas trabajadas en la semana. En el caso de exceder las 40 horas semanales, las horas extras se deben pagar con un extra del 50% del valor de la hora. Realizar un menú de opciones para poder elegir el tipo de contrato que tiene un empleado.

VER VÍDEO: Bucle "Mientras"

22. Escriba un programa en el cual se ingrese un numero y mientras ese numero sea mayor de 10, se pedirá el numero de nuevo.

23. Escriba un programa que valide si una nota está entre 0 y 10, sino está entre 0 y 10 la nota se pedirá de nuevo hasta que la nota sea correcta.

24. Escriba un programa que solicite dos números enteros (mínimo y máximo). A continuación, se debe pedir al usuario que ingrese números enteros situados entre el máximo y mínimo. Cada vez que un numero se encuentre entre ese intervalo, se sumara uno a una variable. El programa terminará cuando se escriba un número que no pertenezca a ese intervalo, y al finalizar se debe mostrar por pantalla la cantidad de números ingresados dentro del intervalo.

25. Escriba un programa en el cual se ingrese un valor límite positivo, y a continuación solicite números al usuario hasta que la suma de los números introducidos supere el límite inicial.

26. Dada una secuencia de números ingresados por teclado que finaliza con un –1, por ejemplo: 5,3,0,2,4,4,0,0,2,3,6,0,.....,-1; realizar un programa que calcule el promedio de los números ingresados. Suponemos que el usuario no insertará número negativos.

27. Calcular las calificaciones de un grupo de alumnos. La nota final de cada alumno se calcula según el siguiente criterio: la parte práctica vale el 10%; la parte de problemas vale el 50% y la parte teórica el 40%. El programa leerá el nombre del alumno, las tres notas obtenidas, mostrará el resultado por pantalla, y a continuación volverá a pedir los datos del siguiente alumno hasta que el nombre sea una cadena vacía. Las notas deben estar comprendidas entre 0 y 10, y si no están dentro de ese rango no se imprimirá el promedio y se mostrará un mensaje de error.

28. Escribir un programa que calcule cuántos dígitos tiene un número entero positivo (pista: se puede hacer dividiendo varias veces entre 10). Nota: investigar la función trunc().

VER VÍDEO: Bucle "Hacer – Mientras Que"

29. Teniendo en cuenta que la clave es "eureka", escribir un programa que nos pida ingresar una clave. Sólo se cuenta con 3 intentos para acertar, si fallamos los 3 intentos se deberá mostrar un mensaje indicándonos que hemos agotado esos 3 intentos. Si acertamos la clave se deberá mostrar un mensaje que indique que se ha ingresado al sistema correctamente.

30. Se debe realizar un programa que:

    1º) Pida por teclado un número (entero positivo).
    2º) Pregunte al usuario si desea introducir o no otro número.
    3º) Repita los pasos 1º y 2º mientras que el usuario no responda n/N (no).
    4º) Muestre por pantalla la suma de los números introducidos por el usuario.

31. Se pide escribir un programa que calcule la suma de los N primeros números pares. Es decir, si ingresamos el número 5 como valor de N, el algoritmo nos debe realizar la suma de los siguientes valores: 2+4+6+8+10.

32. Escribir un programa que lea números enteros hasta teclear 0 (cero). Al finalizar el programa se debe mostrar el máximo número ingresado, el mínimo, y el promedio de todos ellos.

33. Programar un juego donde la computadora elige un número al azar entre 1 y 10, y a continuación el jugador tiene que adivinarlo. La estructura del programa es la siguiente:

    1º) El programa elige al azar un número n entre 1 y 10.
    2º) El usuario ingresa un número x.
    3º) Si x no es el número exacto, el programa indica si n es más grande o más pequeño que el número ingresado.
    4º) Repetimos desde 2) hasta que x sea igual a n.

    El programa tiene que imprimir los mensajes adecuados para informarle al usuario qué hacer y qué pasó hasta que adivine el número.

    NOTA: Para generar un número aleatorio entre 1 y 10 se puede utilizar la función Aleatorio(limite_inferior, limite_superior) de PSeInt.

VER VÍDEO: Bucle "Para"

34. Escribir un programa que calcule el cuadrado de los 9 primeros números naturales e imprima por pantalla el número seguido de su cuadrado. Ejemplo: "2 elevado al cuadrado es igual a 4", y así sucesivamente.

35. Realizar un programa que muestre la cantidad de números que son múltiplos de 2 o de 3 comprendidos entre 1 y 100.

36. Realizar un programa que pida una frase y el programa deberá mostrar la frase en con un espacio entre cada letra. La frase se mostrara así: H o l a. Nota: recordar el funcionamiento de la función Subcadena().

    NOTA:. En PSeInt, si queremos escribir sin que haya saltos de línea, al final de la operación "escribir" escribimos "sin saltar". Por ejemplo:

    Escribir sin saltar "Hola, "
    Escribir sin saltar "cómo estás?"
    Imprimirá por pantalla: Hola, cómo estás?

37. Siguiendo el ejercicio anterior, ahora deberemos hacer lo mismo pero que la cadena se muestre al revés. Por ejemplo, si tenemos la cadena: Hola, deberemos mostrar **a l o H.**

38. Un docente de Programación tiene un listado de 3 notas registradas por cada uno de sus N estudiantes. La nota final se compone de un trabajo práctico Integrador (35%), una Exposición (25%) y un Parcial (40%). El docente requiere los siguientes informes claves de sus estudiantes:

    - Nota promedio final de los estudiantes que reprobaron el curso. Un estudiante reprueba el curso si tiene una nota final inferior a 6.5
    - Porcentaje de alumnos que tienen una nota de integrador mayor a 7.5.
    - La mayor nota obtenida en las exposiciones.
    - Total de estudiantes que obtuvieron en el Parcial entre 4.0 y 7.5.

    El programa pedirá la cantidad de alumnos que tiene el docente y en cada alumno pedirá las 3 notas y calculará todos informes claves que requiere el docente. **Nota: para trabajar este ejercicio de manera prolija, ir probando cada inciso que pide el ejercicio. No hacer todos al mismo tiempo y después probar.**

VER VIDEO: Bucles Anidados (segunda mitad del video)

39. Realizar un programa que lea un número entero (tamaño del lado) y a partir de él cree un cuadrado de asteriscos de ese tamaño. Los asteriscos sólo se verán en el borde del cuadrado, no en el interior. Por ejemplo, si se ingresa el número 4 se debe mostrar:

    ```
    * * * *
    *     *
    *     *
    * * * *
    ```

    Nota: Recordar el uso del escribir sin saltar en Pseint.

40. Escriba un programa que lea un número entero (altura) y a partir de él cree una escalera invertida de asteriscos con esa altura. Por ejemplo, si ingresamos una altura de 5 se deberá mostrar:

    ```
    *****
    ****
    ***
    **
    *
    ```

41. Una compañía de seguros tiene contratados a n vendedores. Cada vendedor realiza múltiples ventas a la semana. La política de pagos de la compañía es que cada vendedor recibe un sueldo base más un 10% extra por comisiones de sus ventas. El gerente de la compañía desea saber por un lado, cuánto dinero deberá pagar en la semana a cada vendedor por concepto de comisiones de las ventas realizadas, y por otro lado, cuánto deberá pagar a cada vendedor como sueldo total (sueldo base + comisiones).

42. La función factorial se aplica a números enteros positivos. El factorial de un número entero positivo (!n) es igual al producto de los enteros positivos desde 1 hasta n:

    n! = 1 * 2 * 3 * 4 * 5 * (n-1) * n

    Escriba un programa que calcule los factoriales de todos los números enteros desde el 1 hasta el 5. El programa deberá mostrar la siguiente salida:

    !1 = 1
    *!2 = 1*2 = 2*
    ...
    *!5 = 1*2*3*4*5 = 120*

    Nota: si necesitas saber más sobre la función factorial revisar este link: Funcion Factorial

EJERCICIOS DE APRENDIZAJE EXTRAS

Condicional Múltiple

1. Hacer un algoritmo que lea un número por el teclado y determinar si tiene tres dígitos.

Condicionales Anidados

2. Si se compran menos de cinco llantas el precio es de $3000 cada una, si se compran entre 5 y 10 el precio es de $2500, y si se compran más de 10 el precio es $2000. Obtener la cantidad de dinero que una persona tiene que pagar por cada una de las llantas que compra, y el monto total que tiene que pagar por el total de la compra.

3. El promedio de los trabajos prácticos de un curso se calcula en base a cuatro notas de las cuales se elimina la nota menor y se promedian las tres notas más altas. Escriba un programa que determine cuál es la nota eliminada y el promedio de los trabajos prácticos de un estudiante.

4. Una verdulería ofrece las manzanas con descuento según la siguiente tabla:

Nº DE KILOS COMPRADOS	% DESCUENTO
0 – 2	0%
2.01 – 5	10%
5.01 – 10	15%
10.01 en adelante	20%

Determinar cuánto *pagará* una persona que compre manzanas en esa verdulería.

Bucle "Mientras"

5. Escriba un programa que solicite al usuario números decimales mientras el usuario escriba números mayores que el primero que se ingresó. Por ejemplo: si el usuario ingresa como primer número un 3.1, y luego ingresa un 4, el programa debe solicitar un tercer número. El programa continuará solicitando valores sucesivamente mientras los valores ingresados sean mayores que 3.1, caso contrario, el programa finaliza.

Bucle "Hacer – Mientras Que"

6. Realizar un programa que solicite al usuario su código de usuario (un número entero mayor que cero) y su contraseña numérica (otro número entero positivo). El programa no le debe permitir continuar hasta que introduzca como código 1024 y como contraseña 4567. El programa finaliza cuando ingresa los datos correctos.

7. Hacer un algoritmo para calcular la media de los números pares e impares, sólo se ingresará diez números.

Bucle “Para”

8. Escribir un programa que calcule la suma de los N primeros números naturales. El valor de N se leerá por teclado.

9. Calcular el cuadrado de los N primeros números. Mostrar por pantalla.

Bucles Anidados

10. Realizar un programa que calcule la siguiente sumatoria:

    $1 + 1/2! + 1/3! + \ldots + 1/n!$

    donde **n** es un valor entero ingresado por el usuario y **n!** es el factorial de ese número.

11. Escribir un programa que calcule los primeros 4 centros numéricos. Un centro numérico es un número que separa una lista de números enteros (comenzando en 1) en dos grupos de números, cuyas sumas son iguales. El primer centro numérico es el 6, el cual separa la lista (1 a 8) en los grupos: (1, 2, 3, 4, 5) y (7, 8) cuyas sumas son ambas iguales a 15. El segundo centro numérico es el 35, el cual separa la lista (1 a 49) en los grupos: (1 a 34) y (36 a 49) cuyas sumas son ambas iguales a 595.
    Nota: investigar que es un centro numérico en caso de no saber que es.