Bootcamp Inteligencia Artificial
Nivel Explorador

Semana 3: Python Para Inteligencia Artificial

Sentencias de control IF Sentencia de control ciclos (for while)

Agenda

1. Sentencias de Control
2. Estructura Condicional IF
3. Estructura de repetición While
4. Estructura de repetición For

1.1 Sentencias de control

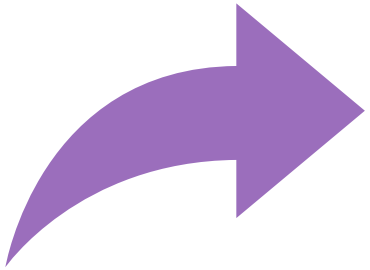
La estructura secuencial es aquella en la que una acción (instrucción) sigue a otra en secuencia. Las tareas se suceden de tal modo que la salida de una es la entrada de la siguiente y así sucesivamente hasta el fin del proceso. Una estructura secuencial se representa de la siguiente forma:

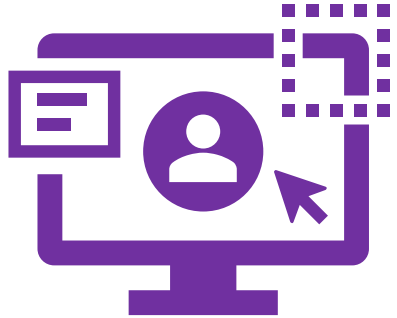
1.2 Sentencias de control

Inicio

Accion1
Accion2
.
.
.
AccionN

Fin





1.3 Sentencias de control

Es importante recordar lo siguiente:

Asignación: La asignación consiste, en el paso de valores o resultados a una zona de la memoria. Dicha zona será reconocida con el nombre de la variable que recibe el valor. La asignación se puede clasificar de la siguiente forma:

- Simples: Ejem. (a=15)
- Contador: Ejem. (a=a+1)
- Acumulador: Ejem. (a=a+b)
- De trabajo: Ejem.(a=c+b*2/4)

1.4 Sentencias de control

Lectura: consiste en recibir desde un dispositivo de entrada (p.ej. el teclado) un valor. Esta operación se representa en un pseudocodigo como sigue:

Leer a, b

Donde “a” y “b” son las variables que recibirán los valores

En Python: una de las instrucciones de lectura es input("....")

1.5 Sentencias de control

Escritura: Consiste en mandar por un dispositivo de salida (p.ej. monitor o impresora) un resultado o mensaje. Este proceso se representa en un pseudocódigo como:

Escribe “El resultado es:”, R

Donde “El resultado es:” es un mensaje que se desea aparezca y R es una variable que contiene un valor.

En Python: instrucción print("....")

1.6 Sentencias de control. Ejemplo

- Suponga que un individuo desea invertir su capital en un banco y desea saber cuánto dinero ganará después de un mes si el banco paga a razón de 15% efectivo anual.

- Un vendedor recibe un sueldo base, más un 10% extra por comisión de sus ventas, el vendedor desea saber cuánto dinero obtendrá por concepto de comisiones por las tres ventas que realiza en el mes y el total que recibirá en el mes tomando en cuenta su sueldo base y comisiones.

1.7 Sentencias de control. Ejemplo

- Una tienda ofrece un descuento del 15% sobre el total de la compra y un cliente desea saber cuánto deberá pagar finalmente por su compra.

- Un alumno desea saber cuál será su calificación final en la materia de Algoritmos. Dicha calificación se compone de los siguientes porcentajes: 40% del promedio de sus tres calificaciones parciales, 50% de la calificación del examen final y 10% de la calificación de un trabajo final.

1.8 Sentencias de control. Ejercicio

Para los dos primeros, como guía se muestra el algoritmo

- **Realizar un algoritmo que calcule la edad de una persona**.

```
Inicio
        Leer fnac, fact
        edad = fact - fnac
        Imprimir eda
Fin
```

1.9 Sentencias de control. Ejercicio

Un maestro desea saber que porcentaje de hombres y que porcentaje de mujeres hay en un grupo de estudiantes

```
Inicio
    Leer nh, nm
    ta = nh + nm
    ph = nh * 100 / ta
    pm = nm * 100 / ta
    Imprimir ph, pm
Fin
```

1.10 Sentencias de control. Ejercicio

- Dada una cantidad en pesos, obtener la equivalencia en dólares, asumiendo que la unidad cambiaría a un dato desconocido.
- Capturar un número y escribir el valor absoluto del mismo.
- La presión, el volumen y la temperatura de una masa de aire se relacionan por la fórmula: masa = (presión * volumen)/(0.37 * (temperatura + 460))
- Calcular el número de pulsaciones que una persona debe tener por cada 10 segundos de ejercicio, si la fórmula es: numPulsaciones = (220 - edad)/10

2.1 Estructura Condicional. Simple

Las estructuras condicionales comparan una variable contra otro(s) valor(es), para que con base al resultado de esta comparación, se siga un curso de acción dentro del programa. Cabe mencionar que la comparación se puede hacer contra otra variable o contra una constante, según se necesite. Existen dos tipos básicos, las simples y las múltiples.

Las estructuras condicionales simples se les conoce como, “Tomas de decisión”.

Estas tomas de decisión tienen la siguiente forma:

Si <condición> entonces
Acción(es)
Fin-si

2.2 Estructura Condicional. Simple. Ejemplo

Un hombre desea saber cuánto dinero se genera por concepto de intereses sobre la cantidad que tiene en inversión en el banco. El decidirá reinvertir los intereses siempre y cuando estos excedan a $7000, y en ese caso desea saber cuánto dinero tendrá finalmente en su cuenta.

```
Inicio
        Leer p_int, cap
         int = cap * p_int
         si (int > 7000 ) entonces
                  capf = cap + int
          fin-si
          Imprimir capf
Fin
```

2.4 Estructura Condicional. Doble

Las estructuras condicionales dobles permiten elegir entre dos opciones o alternativas posibles en función del cumplimiento o no de una determinada condición. Se representa de la siguiente forma:

```
Si <condición> entonces
        Acción(es)
Si no
        Acción(es)
Fin si
```

2.5 Estructura Condicional. Doble. Ejemplo

Determinar si un alumno aprueba o reprueba un curso, sabiendo que aprobara si su promedio de tres calificaciones es mayor o igual a 70; reprueba en caso contrario.

```
Inicio
    Leer calif1, calif2, calif3
    prom = (calif1 + calif2 + calif3)/3
    Si( prom >= 70 )entonces
            Imprimir “alumno aprobado”
     Si no
            Imprimir “alumno reprobado”
    Fin-si
Fin
```

2.6 Estructura Condicional. Anidado

Las estructuras condicionales anidadas se dan, cuando si el bloque de código verdadero o el bloque de código falso, contiene otra sentencia condicional.

```
Si <condición> entonces
        Acción(es)
Si no
        Si <condición> entonces
            Acción(es)
        Si no
                Si <condición> entonces
                        Acción(es)
                Si no
                        Acción(es)
                Fin si
        Fin si
Fin si
```

2.7 Estructura Condicional. Anidado. Ejemplo

Leer 2 números; si son iguales que los multiplique, si el primero es mayor que el segundo que los reste y si no que los sume.

```
Inicio
        Leer num1, num2
        Si (num1 = num2) entonces
                resul = num1 * num2
        Si no
                Si (num1 > num2) entonces
                        resul = num1 - num2
                Si no
                        resul = num1 + num2
                fin-si
        fin-si
Fin
```

2.8 Estructura Condicional. Ejercicios

1. Determinar la cantidad de dinero que recibirá un trabajador por concepto de las horas extras trabajadas en una empresa, sabiendo que cuando las horas de trabajo exceden de 40, el resto se consideran horas extras y que estas se pagan al doble de una hora normal cuando no exceden de 8; si las horas extras exceden de 8 se pagan las primeras 8 al doble de lo que se pagan las horas normales y el resto al triple.

2. En una tienda de descuento se efectúa una promoción en la cual se hace un descuento sobre el valor de la compra total según el color de la bolita que el cliente saque al pagar en caja. Si la bolita es de color blanco no se le hará descuento alguno, si es verde se le hará un 10% de descuento, si es amarilla un 25%, si es azul un 50% y si es roja un 100%. Determinar la cantidad final que el cliente deberá pagar por su compra. se sabe que solo hay bolitas de los colores mencionados.

3. En una fábrica de computadoras se planea ofrecer a los clientes un descuento que dependerá del número de computadoras que compre. Si las computadoras son menos de cinco se les dará un 10% de descuento sobre el total de la compra; si el número de computadoras es mayor o igual a cinco pero menos de diez se le otorga un 20% de descuento; y si son 10 o más se les da un 40% de descuento. El precio de cada computadora es de $3.500.000

3.1 Estructura de repetición. While

Un bucle while permite repetir la ejecución de un grupo de instrucciones mientras se cumpla una condición (es decir, mientras la condición tenga el valor True).

```
while <condición>:
    cuerpo del bucle
```

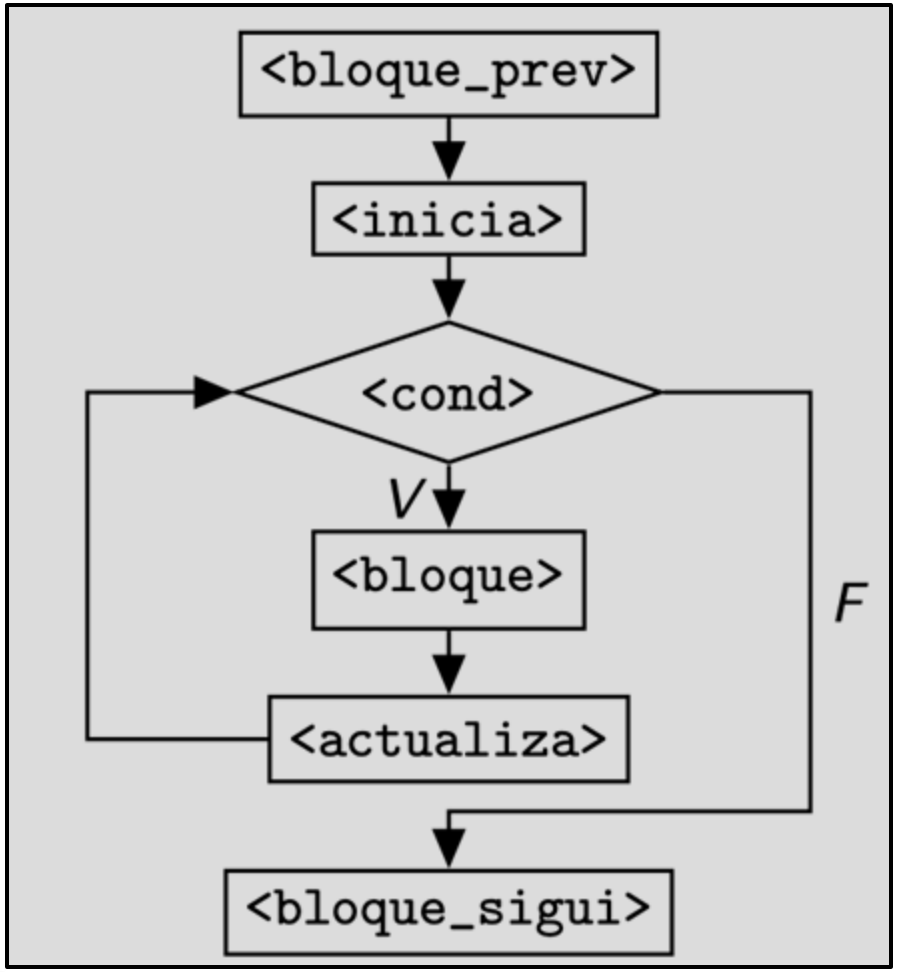
3.3 Estructura de repetición. While

- El ciclo mientras (while) permite ejecutar un bloque de instrucciones mientras que una expresión booleana dada se cumpla, es decir, mientras su evaluación dé como resultado verdadero.

- La expresión booleana se denomina condición de parada y siempre se evalúa antes de ejecutar el bloque de instrucciones; tras esto se pueden presentar dos casos:

    - Si la condición no se cumple, el bloque no se ejecuta.
    - Si la condición se cumple, el bloque se ejecuta, después de lo cual la instrucción vuelve a empezar, es decir, la condición se vuelve a evaluar.

3.4 Estructura de repetición. While



3.5 Estructura de repetición. While. Ejemplos

1. Escriba un programa que capture un número entero y que compruebe si el número es menor que 10. Si no lo está, debe volver a capturar el número repitiendo la operación hasta que el usuario escriba un valor correcto. Finalmente, debe escribir por pantalla el valor leído cuando este sea correcto.
2. Modifique el algoritmo del problema anterior para que, en vez de comprobar que el número sea menor que 10, compruebe que se encuentre en el rango (0, 20).
3. Escriba un programa que sume los números ingresados por el usuario hasta que el usuario ingrese el número 0 (detener las preguntas ante este escenario).
4. Escriba un programa que sume los números ingresados por el usuario y cuando la suma sea superior a 100 deje de pedir números y muestre el total.

3.5 Estructura de repetición. While. Ejercicios

1. Elaborar un programa que muestre cuantos datos desee un usuario, de la serie de Fibonacci.
2. Leer números enteros de teclado, hasta que el usuario ingrese el 0. Finalmente, mostrar la sumatoria de todos los números positivos ingresados.
3. Crear un programa que permita al usuario ingresar los montos de las compras de un cliente (se desconoce la cantidad de datos que cargará, la cual puede cambiar en cada ejecución), cortando el ingreso de datos cuando el usuario ingrese el monto 0.
4. Si ingresa un monto negativo, no se debe procesar y se debe pedir que ingrese un nuevo monto. Al finalizar, informar el total a pagar teniendo que cuenta que, si las ventas superan el total de $1000, se le debe aplicar un 10% de descuento.

4.1 Estructura de repetición. For

Un bucle for es un bucle que repite el bloque de instrucciones un número predeterminado de veces. El bloque de instrucciones que se repite se suele llamar cuerpo del bucle y cada repetición se suele llamar iteración.

for <elem> **in** <iterable>:
    cuerpo del bucle

4.2 Estructura de repetición. For

- Esta estructura de control tiene dos propósitos primordiales que no siempre son soportados por todo lenguaje de programación:

    1. Como una forma compacta de escribir un ciclo mientras (while).
    2. Para iterar sobre los elementos de una colección de elementos.

- Esta estructura es usualmente utilizada cuando se conocen los valores inicial y final de la variable que es utilizada en la condición de parada.

4.3 Estructura de repetición. For

- Un ciclo para (for) puede ser usado para obtener uno a uno los elementos de una colección de elementos y poder realizar con cada uno de ellos el mismo bloque de operaciones.
- Un esquema textual que en Python representa dicho ciclo (for) es el que se da en el siguiente fragmento de código.

```
<bloque_prev>
for <elemento> in <coleccion>:
    <bloque>
<bloque_sigui>
```

4.4 Estructura de repetición. For

- El fragmento <bloque_prev> es el bloque de instrucciones previas que han sido ejecutadas antes del ciclo.
- El fragmento <elemento> es la variable que se usa para ir recorriendo (iterando) sobre los elementos de la colección.
- El fragmento <coleccion> es la colección de elementos que ser´a recorrida (iterada) con el ciclo.
- El fragmento <bloque> es el bloque de instrucciones principal del ciclo que se ejecuta con cada uno de los elementos de la colección.
- El fragmento <bloque_sigui> es el bloque de instrucciones que se ejecutan después de terminar de ejecutar el ciclo.

4.5 Colección Range.

- Existen varias colecciones que se pueden iterar en Python, una de ellas es la colección (range).
- Una colección (range) es una colección de números en un intervalo, definido por valor inicial, un valor final y un valor de incremento/decremento usado a partir del valor inicial para determinar que valores quedan en el rango.
- Si no se da el valor de inicio, éste se fija en cero (0) y si no se da valor de incremento/decremento, este se fija en uno (1).

4.6 Colección Range.

Los rangos se combinan perfectamente con la instruccion (for):

- Se define una variable que se utilizará para recorrer cada número en el rango.

```
for i in range(-5, 6, 1):
    print(i)

for I in range(10,0,-1):
    print(i)

for i in range(0, 55, 5):
    print(i)
```

4.7 Estructura de repetición. For. Ejemplos

1. Crear un algoritmo que muestre los primeros 10 números de la sucesión de Fibonacci. La sucesión comienza con los números 0 y 1 y, a partir de éstos, cada elemento es la suma de los dos números anteriores en la secuencia: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55…
2. Dado un número entero positivo, mostrar su factorial. El factorial de un número se obtiene multiplicando todos los números enteros positivos que hay entre el 1 y ese número.
3. Solicitar al usuario que ingrese una frase y luego imprimir la cantidad de vocales que se encuentran en dicha frase.
4. Escribir un programa que solicite al usuario una cantidad y luego itere la cantidad de veces dada. En cada iteración, solicitar al usuario que ingrese un número. Al finalizar, mostrar la suma de todos los números ingresados.

4.7 Estructura de repetición. For. Ejercicios

Escribir un programa que permita al usuario ingresar 6 números enteros, que pueden ser positivos o negativos. Al finalizar, mostrar la sumatoria de los números negativos y el promedio de los positivos.

Nota: No olvidar que no es posible dividir por cero, por lo que es necesario evitar que el programa arroje un error si no se ingresaron números positivos.

Preguntas