Fundamentos de Programación

NTRODUCCIÓN A LOS FUNDAMENTOS DE PROGRAMACIÓN 3
Semana 03

OBJETIVO DEL LABORATORIO

Identifica la estructura de un algoritmo y sus componentes (variables).

MARCO TEÓRICO

Sabiendo la estructura de un algoritmo, podremos definir las entradas que necesitamos para posterior a ello, generar un proceso entre ellas; que finalmente mostrarán el resultado que esperamos.

Es esencial en Programación, saber identificar las entradas en un caso específico; para no cometer errores futuros con la generación de código basura.

Como fuente de información, se tiene la misma plataforma del Netacad. Tanto para realizar los ejemplos propuestos en la interfaz integrada, así como para repasar lo visto en clase.

RECURSOS

a. Hardware
- Pc Pentium IV a superior
- Conexión de red

b. Software
- Sistema Operativo Windows XP a superior
- Navegador Chrome o Firefox
- Edube Sandbox de Python desde Netacad

PROCEDIMIENTO

QUÉ ES UNA VARIABLE

1. VARIABLES EN MATEMÁTICAS

   El concepto de "variable" proviene de las Matemáticas. En Matemáticas, una variable es un símbolo que forma parte de una expresión o de una fórmula. Normalmente las variables se representan mediante letras del alfabeto latino (x, y, z, n, i, j, etc.). Dependiendo del contexto, las variables significan cosas distintas. Por ejemplo:

   - En el caso del Álgebra, una variable representa una cantidad desconocida que se relaciona con otras y que en algunos casos podemos averiguar.

     Consideremos por ejemplo la ecuación:

     ```
     x + 3 = 5
     ```

     En este caso, la variable x representa una cantidad desconocida, pero de la que se sabe que si se le suma 3 se obtiene 5. Resolviendo la ecuación, obtenemos inmediatamente que la variable x estaba representando realmente el número 2.

- En el caso del Análisis matemático, una variable no representa una cantidad determinada, sino que representa todo un conjunto de valores:

  Consideremos por ejemplo la ecuación de la recta:

  ```
  y = x + 1
  ```

  En este caso, la variable x no representa ningún valor concreto, sino que puede tomar cualquier valor numérico positivo o negativo. Para cada valor de x podemos calcular el valor correspondiente de la variable y. Si interpretamos x e y como las coordenadas de puntos en un plano y dibujamos varios puntos, podríamos ver que todos los puntos se encuentran situados en una misma recta.

2. VARIABLES EN PROGRAMACIÓN

En Programación también existe el concepto de "variable", parecido, pero no idéntico al concepto matemático.

En Programación, las variables están asociadas a variables concretas. Además, cada lenguaje de programación tiene su forma de implementar el concepto de variable, por lo que lo que se explica a continuación es válido para muchos lenguajes de programación, aunque otros lenguajes de programación permiten otras posibilidades.

- En muchos lenguajes de programación, una variable se puede entender como una especie de caja en la que se puede guardar un valor (por ejemplo, un valor numérico). Esa caja suele corresponder a una posición de memoria en la memoria del ordenador.
- Las variables se representan también mediante letras o palabras completas: x, y, a, b, nombre, apellidos, edad, etc.
- Cuando en esos lenguajes de programación escribimos la instrucción ...

  ```
  a = 2
  ```

  ... lo que estamos pidiendo al programa es que guarde el valor 2 en una "caja" y que a la caja le llame a.

  En esos lenguajes, el símbolo igualdad (=) hay que entenderlo como una asignación, no como una igualdad matemática. Al escribir una igualdad, le estamos pidiendo al programa que calcule lo que hay a la derecha de la igualdad y que lo guarde en la variable que hay a la izquierda de la igualdad.

  En esos lenguajes, no estaría permitido escribir ...

  ```
  2 = a
  ```

  ... porque 2 no es un nombre válido de variable. Tampoco estaría permitido escribir ...

  ```
  x + 3 = 5
  ```

  ... porque en el lado izquierdo no pueden aparecer operadores.

- Una vez hemos guardado un valor en una variable, podemos hacer referencia a él a lo largo del programa. Por ejemplo, el siguiente programa ...

```
a = 2
b = 3
c = a + b
```

  - guarda el valor 2 en la variable a
  - guarda el valor 3 en la variable b
  - coge los valores guardados en las variables a y b (2 y 3), los suma (2+3) y el resultado (5) lo guarda en la variable c.

3. VARIABLES EN PYTHON

En algunos lenguajes de programación, las variables se pueden entender como "cajas" en las que se guardan los datos, pero en Python las variables son "etiquetas" que permiten hacer referencia a los datos (que se guardan en unas "cajas" llamadas objetos).

Python es un lenguaje de programación orientado a objetos y su modelo de datos también está basado en objetos.

Para cada dato que aparece en un programa, Python crea un objeto que lo contiene. Cada objeto tiene:

- un identificador único (un número entero, distinto para cada objeto). El identificador permite a Python referirse al objeto sin ambigüedades.
- un tipo de datos (entero, decimal, cadena de caracteres, etc.). El tipo de datos permite saber a Python qué operaciones pueden hacerse con el dato.
- un valor (el propio dato).

Así, las variables en Python no guardan los datos, sino que son simples nombres para poder hacer referencia a esos objetos. Si escribimos la instrucción ...

```
a = 2
```

... Python:

- crea el objeto "2". Ese objeto tendrá un identificador único que se asigna en el momento de la creación y se conserva a lo largo del programa. En este caso, el objeto creado será de tipo número entero y guardará el valor 2.
- asocia el nombre a al objeto número entero 2 creado.

Así, al describir la instrucción anterior no habría que decir 'la variable a almacena el número entero 2', sino que habría que decir 'podemos llamar a al objeto número entero 2'. La variable a es como una etiqueta que nos permite hacer referencia al objeto "2", más cómoda de recordar y utilizar que el identificador del objeto.

4. DEFINIR UNA VARIABLE

Las variables en Python se crean cuando se definen por primera vez, es decir, cuando se les asigna un valor por primera vez. Para asignar un valor a una variable se utiliza el operador de igualdad (=). A la

izquierda de la igualdad se escribe el nombre de la variable y a la derecha el valor que se quiere dar a la variable.

En el ejemplo siguiente se almacena el número decimal 2,5 en una variable de nombre x (como se comenta en el apartado anterior, realmente habría que decir que se crea la etiqueta x para hacer referencia al objeto número decimal 2.5). Fíjese en que los números decimales se escriben con punto (.) y no con coma (,).

```
>>> x = 2.5
```

Para que IDLE muestre el valor de una variable, basta con escribir su nombre:

```
>>> x = 2.5
>>> x
2.5
```

Una variable puede almacenar números, texto o estructuras más complicadas (que se verán más adelante). Si se va a almacenar texto, el texto debe escribirse entre comillas simples (') o dobles ("), que son equivalentes. A las variables que almacenan texto se les suele llamar cadenas (de texto).

```
>>> nombre = "Pepito Conejo"
>>> nombre
'Pepito Conejo'
```

5. NOMBRES DE VARIABLES

Aunque no es obligatorio, se recomienda que el nombre de la variable esté relacionado con la información que se almacena en ella, para que sea más fácil entender el programa. Si el programa es trivial o mientras se está escribiendo un programa, esto no parece muy importante, pero si se consulta un programa escrito por otra persona o escrito por uno mismo, pero hace tiempo, resultará mucho más fácil entender el programa si los nombres están bien elegidos.

También se acostumbra a utilizar determinados nombres de variables en algunas ocasiones, como se verá más adelante, pero esto tampoco es obligatorio.

El nombre de una variable debe empezar por una letra o por un guión bajo (_) y puede seguir con más letras, números o guiones bajos.

```
>>> _x = 3.8
>>> _x
3.8

>>> x1 = 100
>>> x1
100



>>> fecha_de_nacimiento = "27 de octubre de 1997"
>>> fecha_de_nacimiento
'27 de octubre de 1997'
```

Los nombres de variables no pueden incluir espacios en blanco.

```
>>> fecha de nacimiento = "27 de octubre de 1997"
SyntaxError: invalid syntax
```

Los nombres de las variables pueden contener mayúsculas, pero tenga en cuenta que Python distingue entre mayúsculas y minúsculas (en inglés se dice que Python es "case-sensitive").

```
>>> nombre = "Pepito Conejo"
>>> Nombre = "Numa Nigerio"
>>> nomBre = "Fulanito Mengánez"
>>> nombre
'Pepito Conejo'
>>> Nombre
'Numa Nigerio'
>>> nomBre
'Fulanito Mengánez'
```

Cuando el nombre de una variable contiene varias palabras, se aconseja separarlas con guiones bajos para facilitar la legibilidad, aunque también se utiliza la notación camelCase, en las que las palabras no se separan pero empiezan con mayúsculas (salvo la primera palabra).

```
>>> fecha_de_nacimiento = "27 de octubre de 1997"
>>> fechaDeNacimiento = "27 de octubre de 1997"
```

6. TIPOS DE VARIABLES

En el apartado anterior hay definiciones de algunos de los tipos de variables que hay en Python: números decimales, números enteros y cadenas (una o más letras).

Aunque se definan de forma similar, para Python no es lo mismo un número entero, un número decimal o una cadena ya que, por ejemplo, dos números se pueden multiplicar, pero dos cadenas no (curiosamente, una cadena sí que se puede multiplicar por un número).

Por tanto, estas tres definiciones de variables no son equivalentes:

```
>>> fecha = 1997
>>> fecha = 1997.0
>>> fecha = "1997"
>>> fecha = [27, "octubre", 1997]
```

En el primer caso la variable *fecha* está almacenando un número entero, en el segundo *fecha* está almacenando un número decimal y en el tercero *fecha* está almacenando una cadena de cuatro letras. En el cuarto, *fecha* está almacenando una lista (un tipo de variable que puede contener varios elementos ordenados).

Este ejemplo demuestra también que se puede volver a definir una variable. Python modifica el tipo de la variable automáticamente.

7. MOSTRAR EL VALOR DE LAS VARIABLES EN IDLE

Para que IDLE muestre el valor de una variable, basta con escribir su nombre.

```
>>> a = 2
>>> a
```

```
2
```

También se puede conocer el valor de varias variables a la vez escribiéndolas entre comas (IDLE las mostrará entre paréntesis porque Python las considera como un conjunto ordenado llamado tupla), como muestra el siguiente ejemplo:

```
>>> a = b = 2
>>> c = "pepe"
>>> a
2
>>> c, b
('pepe', 2)
```

8. UTILIZAR O MODIFICAR VARIABLES YA DEFINIDAS

Una vez se ha definido una variable, se puede utilizar para hacer cálculos o para definir nuevas variables, como muestran los siguientes ejemplos:

```
>>> a = 2
>>> a + 3
5
>>> horas = 5
>>> minutos = 60 * horas
>>> segundos = 60 * minutos
>>> segundos
18000
>>> horas = 1
>>> minutos = 2
>>> segundos = 3
>>> segundos + 60 * minutos + 3600 * horas
3723
```

Se puede redefinir una variable a partir de su propio valor. Por ejemplo

```
>>> a = 10
>>> a = a + 5
>>> a
15
```

Es importante también tener presente que los nombres de las variables no tienen sentido real para Python, son simples etiquetas para referirse al contenido.

Por ejemplo, en el siguiente ejemplo se crean dos variables y se suman:

```
>>> peras = 7
>>> manzanas = 22
>>> peras + manzanas
29
```

O el siguiente ejemplo en el que Python permite realizar cualquier operación con las variables *distancia* y *tiempo*, aunque no tengan sentido físico (de hecho, sólo la división lo tiene, es la velocidad).

```
>>> distancia = 100
```

```
>>> tiempo = 4
>>> distancia + tiempo
104
>>> distancia * tiempo
400
>>> distancia / tiempo
25.0
```

Cuando se modifica una variable, el valor anterior se pierde y no se puede recuperar (salvo si se realiza la operación contraria o hay otra variable que conserva el valor anterior).

9. ASIGNACIONES AUMENTADAS

Cuando una variable se modifica a partir de su propio valor, se puede utilizar la denominada "asignación aumentada", una notación compacta que existe también en otros lenguajes de programación.

Por ejemplo:

```
>>> a = 10
>>> a += 5
>>> a
15
```

es equivalente a:

```
>>> a = 10
>>> a = a + 5
>>> a
15
```

En general:

Asignación aumentada:	es equivalente a:
a += b	a = a + b
a -= b	a = a - b
a *= b	a = a * b
a /= b	a = a / b
a **= b	a = a ** b
a //= b	a = a // b
a %= b	a = a % b

CONCLUSIONES Y RECOMENDACIONES DE LA EXPERIENCIA

- Sabiendo cómo es la estructura de un algoritmo, podremos identificar las entradas necesarias; para mostrar la salida esperada.

- Se recomienda usar el Edube SandBox para realizar nuestros ejercicios sin necesidad de instalar algún software adicional.

ACTIVIDAD VIRTUAL

1. Observa y analiza el siguiente video: https://www.youtube.com/watch?v=eoTravXOsTk, luego responde las siguientes preguntas:

- ¿Qué es un identificador o variable?
- ¿Qué criterios debemos considerar para la composición de un identificador?
- Menciona 4 ejemplos de identificadores válidos y 2 que sean inválidos, que no sean iguales a los del video