Bitácora 5 del Taller de Herramientas Computacionales

Elías Jiménez Cruz, 409085596

12/01/2019

Retomando lo visto y hecho la clase anterior, aprendimos primero un uso más elaborado de la instrucción print de Python. Para ello vimos un nuevo tipo de variable, que es la variable cadena, o string. Esta variable almacena texto, guarda una secuencia de caracteres sin algún valor lógico. La manera de declarar una cadena es poner el texto a declarar entre comillas, es decir, "Esto es una cadena.". Se puede guardar con algún nombre específico, tal y como se guarda una variable de tipo entero. Ahora bien, con el comando print, se puede mandar a imprimir una cadena sin mayores inconvenientes, sólo se escribe print "cadena". Esto es útil en programas complejos, para solicitar información o mostrar información que ayude a los usuarios a navegar por el sistema. Sin embargo, las cadenas también pueden volverse útiles en programas simples, en el momento en que se desea editar el resultado de algún cálculo hecho en el programa. En vez de sólo imprimir el número que da daterminada operación, con una cadena se puede mandar a imprimir el número con algún comentario, que lo haga entendible a aquél que lo lea. En el programa hecho en la clase anterior, con una cadena resulta sencillo mandar a imprimir la posición del objeto en el segundo cinco de una forma legible: "La posición de la pelota tras 5 segundos de haber sido lanzada es...", lo cual además le confiere cierta elegancia. Para lograr esto es necesario hacer uso de cierto comodines que se colocan en los lugares de la cadena donde queremos imprimir las variables. Se utiliza el símbolo % seguido

de una letra que indica el tipo de variable que desea imprimirse. A continuación, al final de la cadena, se insertan las variables a imprimir, en el orden en que se fueron colocando sus comodines respectivos y en el formato %(variable1, variable2, etc...). Ahora bien, respecto a estas letras, hay muchas y diversas, pero para mencionar algunas %E es para formato científico, %s para cadena, %g para imprimir la variable en su expresión más corta posible, %f para imprimir una variable en una expresión flotante, usualmente se usa de la forma %a.bf, donde a es el número mímimo de caracteres a imprimir y b el número de decimales a imprimir. Si a no está determinado, se imprime el valor de la variable sin alterar. También tenemos %e para formato científico de números pequeños o %d para número enteros. Se realizó un programa, de nombre EjemplosPrint.py para mostrar más comodines y sus usos. Ahora, como se dejó entrever, es posible imprimir variables de algún tipo determinado en un formato específico, como en el caso de las variables flotantes, aunque el mencionado archivo ya se encarga de mostrar más ejemplos. Así, finalmente retornamos al caso del programa de la caída libre, y haciendo un nuevo programa de nombre Ejemplo1Pelota_05.py se editó el resultado mediante una cadena. El código que imprime la cadena es: "print 'La posición de la pelota en el t=%g es %.2f' %(t,y)".
Finalmente pasamos a ver una función diferente de Python: Las funciones. Éstas son una manera de crear "acciones" en Python que pueden invocarse en un momento deseado, sin tener que volver a implementar el código completo. Por ejemplo, en el caso del programa de caída libre, puede implementarse una función que calcule la posición del objeto no sólo para valores fijos, sino para los valores que deseemos sin tener que escribir la fórmula en todo momento. Para esto se usa el comando def seguido del nombre que deseemos poner a la función, seguido de las variables que serán usadas para hacer nuestro cálculo entre paréntesis y separadas por comas, todo seguido por dos puntos. Entonces nosotros debemos escribir debajo nuestra operación, guardándola en una variable que será devuelta por la función con el comando return. la variable devuelta por "return" determinará el tipo de variable que será otorgado a la función, y podrá utilizarse como si de una variable normal se tratará. Así pues, en el programa de la caída libre, podemos es-

cribir su función de la siguiente manera:

```
def CaidaLibre(tiempo, VelocidadInicial):
    posicion = v0*t - 1.0/2*9.81*t**2
    return(posicion)
```

Esto permitirá introducir los valores que deseen para el tiempo y la velocidad inicial, simplemente con invocar la función por su nombre e incluir en el orden establecido los valores entre paréntesis. Si se corre el archivo una vez, incluso idle es capaz de invocar la función desde donde sea que se haya guardado el archivo.
El anterior sin duda fue un gran avance, dejándose como tarea el implementar una función de la tarea pasada. Sin embargo, la clase aún no terminaba y se comenzó a ver Latex, que es un sistema de composición de textos, que permite crear documentos de alta calidad y que es usado por muchos matemáticos para escribir sus libros o los resultados de sus investigaciones. Para ello, al igual que Python, se instaló un editor de textos con el comando apt, de nombre texstudio, y en mi caso se instaló el lenguaje de Latex bajo el nombre de texlive, pues muchas otras distribuciones de Linux ya incluyen el lenguaje por defecto. Ahora bien, creamos un documento nuevo de nombre Ejercicio01.tex, con el que aprendimos la estructura básica de todo documento de Latex, así como algunos comandos de edición. Aprendimos los comandos para comenzar el documento, para crear un título, para que el sistema reconozca caracteres de español, para poner negritas o cambiar de color el texto y para enumerar un listado, lo cual será esencial para crear las bitácoras del curso.