Taller de Herramientas Computacionales

Stephanie Escobar Sánchez

21/enero/2019

Bitácora clase 11

La clase comenzó viendo algunos comandos en python, por ejemplo, el comando *input* que muestra una cadena esperando una respuesta, lo cual es útil cuando es necesario calcular valores y que las variables las de el usuario.

1 LISTAS:

Retomamos el concepto de lista y es es posible crear una lista metiendo los elementos manualmente de la forma **L = ["elementos que queremos agregar"]**. Está vez vimos una forma automatizada de crear las listas en caso de que nuestra lista tenga muchos valores, este es con el comando **range**, utilizado de la forma:

```
for 1 in range (n):
Valor = input ("dame el valor")
L.append(valor)
```

Otro comando importante de las listas es **len**, que cuenta cuántos elementos tiene la lista. Esos fueron los promeros dos comandos importantes que vimos de las listas.

2 EJERCICIO DE CLASE:

Comenzamos construyendo problemas básicos, ya que cuando se construye algo se puede ir avanzando mejor. El ejercicio fue hacer una lista que contuviera grados F, así como los grados correspondientes en C, para ello fueron utilizados los comandos previamente mencionados. Los datos de entrada son los datos que ya tenemos dados y los traducimos en argumentos de una función. Las funciones **for in** me regresan un objeto, y **range** solo imprime el valor.
También vimos que es posible hacer funciones anidadas con paréntesis y se ejecuta primero la primera opción y después las funciones de afuera. El comando **enumerate** regresa por cada entrada de una lista el indice que le corresponde y después el valor. Y se utiliza de la forma

```
for i,c in enumerate (L1):
L1[i] = c + 5
```

lo que además suma 5. **Enumerate** nos permite crear un arreglo a partir de otro i representa al índice y c su valor Para crear la lista

```
n = 12; gradosC=[-5 + i * 0.5 for i in range(n)]
```

Finalmente en un problema la sintaxis más corta en general es la mejor y es importante saber cómo interpretar las cosas de la manera más sencilla posible.

3 GITHUB

Ya que únicamente necesitabamos archivos .tex y .py en el repositorio, fue necesario aprender cómo evitar todos los demás archivos extra que nos generan tanto Texstudio como Python. Los archivos compilados son los .pyc y Latex nos crea archivos de tipo .tex, PDF, .pyc y .gz que es una carpeta comprimida.
Github tiene la opción de ignorar archivos, y se utiliza poniendo *.log para que los ignore, para ello es necesario crear un archivo llamado .gitgignore en nuestro repositorio, así github no los subirá.