Universidad de Sonora

División de Ciencias Exactas y Naturales

Departamento de Física

Reporte de Actividad 2

Elementos de la programación Python 1

Estefania Eng Duran

Física Computacional I 2016-1

Hermosillo - 12 de febrero de 2016

¿Qué es Python?

Python es un lenguaje de programación de alto nivel y de uso general. Al momento de diseñarse se hizo hincapié en que la sintaxis favoreciera un código legible. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa, por procedimientos y funcional. Es un lenguaje interpretado, usa tecleo dinámico lo que puede ahorrar tiempo durante el desarrollo de un programa.

El interpretador puede ser utilizado interactivamente facilitando la experimentación con las distintas funciones del lenguaje, la escritura de programas de usar y tirar, o para probar un programa que se desarrolla desde cero. Esto aunado a las estructuras de datos de alto nivel lo hace un lenguaje ideal para escribir y aplicar programas en poco tiempo y en muchas plataformas.

Problema 1: Caída libre

Se deja caer una pelota desde el techo de una torre de altura h dada. Se desea saber el tiempo que tarda en tocar el suelo la pelota después de después de haber sido dejada caer. De la segunda ley de Newton se obtiene

$$t = \sqrt{\frac{2h}{g}}$$

donde $g = 9{,}81\ m/s^2$. De esta fórmula surge el siguiente propgrama que calcula la cantidad deseada.

Programa caida.py

```
h = float(input("Proporciona la altura de la torre
en metros: "))

t = math.sqrt(2*h/9.8)

print("El tiempo que tarda la pelota en caer es", t,
"segundos. Suponemos que  la friccion del aire es
despreciable")
```

Corremos el programa ingresando una altura de 50 metros y obtenemos como resultado aproximado 3.19 segundos y obtuvimos lo siguiente.

Corrida de caida.py

```
Proporciona la altura de la torre en metros: 50

('El tiempo que tarda la pelota en caer es',
3.1943828249996997, 'segundos. Suponemos que la
friccion del aire es despreciable')
```

Problema 2: Satélite en órbita

Un satélite que orbita la Tierra a una altura h tiene un periodo T en segundos. La altitud h del satélite sobre la superficie de la Tierra esta dado por la expresión

$$(R+h)^3 = (GMT^2)/(4\pi^2)$$

donde $G = 6{,}67 \times 10^{-11}\ m^3\, kg^{-1}\, s^{-2}$ es la constante de gravitación universal de Newton, $M = 5{,}97 \times 10^{24}\ kg$ es la masa de la Tierra y $R = 6371\ m$ es su radio.

Se desea saber la altura para un periodo dado que es lo que hace el siguiente programa.

Programa satelite.py

```
T = float(input("Proporciona el valor del periodo en
segundos: "))

h = (6.67 * 5.97 * 10**13 * T**2 / (4 * 3.14**2))**(1.0/3.0)
- 6371

print("La altura del satelite es", h, "metros.")
```

Al correr el progrma, obtenemos la altura para satélites con periodos de un día, 90 minutos y 45 minutos.

Corrida satelite.py

24 horas :	`Proporciona el valor del periodo en segundos:`
	`86400`
	`('La altura del satelite es',`
	`42234816.704007186,'metros.')`

90 minutos :	`Proporciona el valor del periodo en segundos:`
	`5400`
	`('La altura del satelite es',`
	`6646199.195104893,'metros.')`

45 minutos :	`Proporciona el valor del periodo en segundos:`
	`2700`
	`('La altura del satelite es',`
	`4184485.612357949,'metros.')`

Problema 3: Transformación de coordenadas

Un punto en el espacio en el sistema de coordenadas esféricas se describe por las cantidades (r, θ, ϕ) donde r es el radio, θ el angulo polar y ϕ el angulo azimutal. Las coordenadas esfericas de un punto se pueden obtener de las coordenadas cartesianas (x, y, z) mediantes las fórmulas

$$r^2 = x^2 + y^2 + z^2$$

$$\frac{z}{r} = \cos(\theta)$$

$$\frac{y}{x} = \tan(\phi)$$

El programa calcula las coordenadas esféricas a partir de las coordenadas cartesianas y las muestra en grados.

Programa esfericas.py

```
from math import pi, sqrt, acos, atan

x = float(input("Ingresa x: "))
y = float(input("Ingresa y: "))
z = float(input("Ingresa z: "))

r = sqrt(x**2 + y**2 + z**2)
theta,phi = acos(z/r),atan(y/x)
d,g = 180*theta/pi,180*phi/pi

print("r =",r,"theta (deg) =",d,"phi (deg) =",g)
```

Hacemos la prueba con un punto en coordenadas cartesianas y vemos como se transforma a coordenadas esféricas.

```
Ingresa x: 3
Ingresa y: 4
Ingresa z: 0

('r =', 5.0,
'theta (deg) =', 90.0,
'phi (deg) =', 53.13010235415598)
```

Entonces el punto (3,4,0) en coordenadas cartesianas tiene un radio $r = 5$, angulo polar $\theta = 90°$ y angulo azimutal $\phi = 53{,}1°$.

Problema 4: Par o impar

Se desea saber si un número es par o impar. Sabemos que los números pares son divisibles entre 2, es decir que su residuo es cero (2n), mientras que los impares tienen residuo 1 (2n+1). Nos apoyamos en la operación de módulo %. El operador modulo (%) trabaja sobre numeros enteros y arroja el residuo cuando el primer operando es dividido por el segundo. Ejemplo: $7 \,\%\, 3 = 1$.

Programa parimpar.py

```
n = int(input("Ingresa un número entero: "))
if n%2==0:
   print("par")
else:
   print("impar")
```

Probamos con diferentes numeros para comprobar que el programa funciona correctamente.

Corrida parimpar.py

```
Ingresa un número entero: 2
par
```

```
Ingresa un número entero: 9
impar
```

```
Ingresa un número entero: 45
impar
```

El siguiente programa compara dos números para asegurarse que uno es par y el otro impar. Si no se cumple esta condición vuelve a solicitar un número par y otro impar. Se repite la petición hasta que se cumpla, entonces te confirma los números elegidos. Nos apoyamos en el while loop statement que repetidamente ejecuta un bloque de codigo siempre y cuando la condición dada sea verdad.

Programa whilestatement.py

```
print("Ingresa dos enteros, uno par y el otro impar.")

m = int(input("Ingresa el primer entero: "))
n = int(input("Ingresa el segundo entero: "))

while (m+n)%2==0:
  print("Uno debe ser impar y el otro impar.")
  m = int(input("Ingresa el primer entero: "))
  n = int(input("Ingresa el segundo entero: "))

print("Los numeros que escogiste son",m,"y",n)
```

Probamos con diferentes pares de enteros para comprobar que el programa funciona correctamente.

Corrida whilestatement.py

```
Ingresa dos enteros, uno par y el otro impar.
Ingresa el primer entero: 5
Ingresa el segundo entero: 19
Uno debe ser impar y el otro impar.
Ingresa el primer entero: 2
Ingresa el segundo entero: 22
Uno debe ser impar y el otro impar.
Ingresa el primer entero: 77
Ingresa el segundo entero: 103
Uno debe ser impar y el otro impar.
Ingresa el primer entero: 54
Ingresa el segundo entero: 128
Uno debe ser impar y el otro impar.
Ingresa el primer entero: 11
Ingresa el segundo entero: 80
('Los numeros que escogiste son', 11, 'y', 80)
```

En efecto el programa no dejó de pedir pares de números hasta que estos fueron un par y un impar.

Problema 5: Series

Los Números de Fibonacci es una sucesión de números enteros aparecen en toda la naturaleza. Este programa calcula la secuencia de Fibonacci menores a 1000.

Programa fibonacci.py

```
f1,f2 = 1,1

while f2<1000:

    print(f2)

    f1,f2 = f2,f1+f2
```

El resultado obtenido al correr esta programa es:

Corrida fibonacci.py

```
1  2  3  5  8  13  21  34  55  89  144  233  377  610  987
```

Modificamos el programa anterior para calcular la sucesión de números de Catalán,

$$C_{n+1} = \frac{2(2n+1)}{n+2} C_n$$

El programa que los calcula lo mostramos a continuación.

Programa catalan.py

```
C,n = 1,0

while C<=1000000:

    print(C)

    C,n = 2*(2*n+1)*C/(n+2),n+1
```

El resultado obtenido es

Corrida catalan.py

```
1   1   2   5   14   42 132 429 1430 4862 16796 58786 20801 2742900
```

Conclusión

Con esta actividad trabajamos con distintos programas escritos en Python para hacer física computacional. En vez de hacer programas desde cero, tomamos programas sencillos como modelo para hacer otros. Continuamos con el uso de la herramienta LaTeXpara hacer reportes.

Referencias

[1] PyMan 0.9.31, *Introduction to Python for science.*

[2] Mark Newman, *Computational Physics with Python*, Capítulo 2.