PH2150 Scientific Computing Skills

Andrew Casey

May 28, 2019

In this problem sheet we will start to operate on the variable type of *lists*. Then we will create our first user defined functions

1 PS2 Ex1: Lists

- Create a list containing the first twenty elements of the periodic table (using full name, not just symbol), Nb do not use the word **list** as the variable name for your list as it is already in use as the function *list*.
- Create a second list containing the next ten elements of the periodic table.
- What is the difference between adding (Concatenating) the lists together or using the append function to join the lists.
- Combine your two lists to make a third list containing 30 elements, Use the len() function to confirm the new list has 30 elements.
- Print the 23rd element in your list.

We will use this list in later exercises.

2 PS2 Ex2: Lists and control structures

- Write a program to create and print a new list containing the elements from your list (30 element list from Ex1) that begin with the letter "s"
- Write a program to create and print a new list containing the elements in your list (from Ex1) that consist of only 4 characters.

3 PS2 Ex3: More Functions

Write a *function* to compute the area of an arbitrary triangle. Nb. this means a user defined function in python i.e *def functionname*. An arbitrary triangle can be described by the coordinates of its three vertices: $(x_1, y_1), (x_2, y_2), (x_3, y_3)$. The area of the triangle is given by the formula:

$$A = \frac{1}{2}[x_2y_3 \ - \ x_3y_2 \ - \ x_1y_3 \ + \ x_3y_1 \ + \ x_1y_2 \ - \ x_2y_1] \tag{1}$$

Write a **function area(vertices)** that returns the area of a triangle whose vertices are specified by the argument vertices, which is a nested list of the vertex coordinates. For example, vertices can be [[0,0], [1,0], [0,2]] if the three corners of the triangle have coordinates (0, 0), (1, 0) and (0, 2). Test the area function on a triangle with known area. Save the program as file: *area_triangle.py*. Add a docstring to your function to explain to the user what the arguments of the function need to be and what it is going to return.

4 PS2 Ex4: Functions operating on Lists

Computing the length of a path. Some object is moving along a path in the plane. At n points of time we have recorded the corresponding (x, y) positions of the object:

$$(x_0, y_0), (x_1, y_2), .., (x_{n-1}, y_{n-1}) \tag{2}$$

The total length L of the path from (x_0, y_0) to (x_{n-1}, y_{n-1}) is the sum of all the individual line segments $((x_{i-1}, y_{i-1})to(x_i, y_i)$, $i = 1, ..., n-1$:

$$L = \sum_{i=1}^{n-1} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \tag{3}$$

Make a function *pathlength(x, y)* for computing L according to the formula. The arguments x and y hold all the $x_0, ..., x_{n-1}$ and $y_0, ..., y_{n-1}$ coordinates, respectively. Test the function on a triangular path with the four points (1, 1), (2, 1), (1, 2), and (1, 1). Name the program file: pathlength.py.

5 PS2 Ex5: Approximate pi

The value of pi equals the circumference of a circle with radius 1/2. Suppose we approximate the circumference by a polygon through $N+1$ points on the circle. The length of this polygon can be found using the *pathlength* function from Exercise PS2 Ex4. Compute $N + 1$ points (x_i, y_i) along a circle with radius 1/2 according to the formula:

$$x_i = \frac{1}{2}\cos(2\pi i/N), \quad y_i = \frac{1}{2}\sin(2\pi i/N), \quad i = 0, ..., N. \tag{4}$$

Call the pathlength function and write out the error in the approximation of pi for $N = 2^k; k = 2; 3; ...; 10$: Name the program file: piapprox.py.