

Numpy

Create Empty Array of size n*m	<code>np.empty((n,m), np.int_)</code>
Create random array of size n*m in range [a b]	<code>np.random.randint(a, b, (n, m))</code>
Create array with all element equals to zeros	<code>np.zeros((n,m), np.int_)</code>
Create array with all element equals to Inf	<code>np.full((1,n), np.inf)</code>
Merge two array as a new array	<code>np.concatenate((a,b))</code>
Copy new array with new address and same value as a	<code>np.copy(a)</code>
<code>location = np.zeros((n, 2), np.int_)</code>	

List

Appends an element to the end of the list	<code>list.append()</code>
Create array with all element equals to None	<code>[None]*n</code>
Delete item in an array by index	<code>del a[index]</code>
<pre>a = ["apple", "banana", "cherry"] b = ["Ford", "BMW", "Volvo"] for i in b: a.append(i)</pre>	

ortool_cp_sat

declare the model	<code>model = cp_model.CpModel()</code>
declare variable with boolean value	<code>model.NewBoolVar(f'x[{i}]{j}')</code>
<pre>x = {} for work in range(number_worker): for task in range(number_task): x[work,task] = model.NewBoolVar(f'x[{work}]{task}')</pre>	

folium

`folium.map(location, tiles, zoom_start)`

Numba

Decorator	<code>@jit, @njit</code>
speed math	<code>fastmath = True</code>
cache function	<code>cache= True</code>
Parallel numba	<code>nogil = True, parallel= True, for i in prange(n):</code>
njit for empty list	<code>@njit</code>
<pre>@njit def create_S() S = List()</pre>	

Matplotlib.pyplot

Draw points given long, lat	<code>plt.plot(X, Y, 'ro')</code>
Draw points vertically	<code>plt.plot(list(zip(temp)), 'ro')</code>
draw line	<code>plt.plot([x1, x2], [y1, y2], 'ro)</code>
change size of a figure	<code>figure(figsize=(width, heigh), dpi=resolution)</code>
show value on figure	<code>plt.text(x, y, value)</code>
<pre>plt.plot([3, 2], [5,6], 'r-', marker='o', markerfacecolor='b', markeredgcolor = 'b', markersize =5, linewidth = 0.5, alpha=0.5) from matplotlib.pyplot import figure figure(figsize=(8, 6), dpi=80) plt.text(5,5,'Total demand = {} and number of sub tours = {}'.format(weight.sum(),len(shortest_path)-1))</pre>	

class

`def __init__(self,....)`

Ortools

Restrict Vehicle	<code>routing.VehicleVar(index).SetValues([-1, 2,3,4])</code>
different limit constraint	<code>routing.AddDimensionWithVehicleCapacity(demand_ _callback_index, 0, # null capacity slack data['vehicle_c- apacities'], # vehicle maximum capacities True, # start cumul to zero 'Capacity')</code>
find vehicle	<code>VehicleVar(i)</code>
first node after depot	<code>start_var = routing.NextVar(routing.Start(vehicle_nbr))</code>
	<code>SetVehicleFixedCost</code>
	<code>AddDimensionWithVehicleCapacity</code>
	<code>SetVehicleCost</code>
	<code>https://acrogenesis.com/or-tools/documentation/user- _manual/manual/vrp/partial_routes.html</code>
	<code>ActiveVar(i) == (NextVar(i) != i).</code>
	<code>SetCumulVarSoftLowerBound(index_end, 2, 10000)</code>