Question	Score:	Out of:	Comments:
1	5	5	The reaction equation is correctly identified and displayed, the code is well organized
2	4	5	The ODE function created allows the computation of analytical solutions into the parameters and the analytical solution is correctly found. Use theory to explain the code's functioning.

3	6	10	You could include the ODE code in the report as the question asks for coding style, solving of ODE45.
			The code does not solve the ode using ode45.
			Should be:
			`#Using solver to solve ODE`
			`sol=solve_ivp(dAdt,[0,1000],[c0], 'RK45', t_eval=t)`
			
			`#Concentrations over time`
			`Numerical_t_conc=sol.y`
			
			Visuals have all correct elements and are easy to interpret. The report analysis is also well done and draws plausible conclusions using qualitative and quantitative data.

4	10	10	Correct code for Euler's 1st order algorithm, well organized code and efficient. The function is programmed in a general way as expected from the rubric. Visualization with all correct elements and is well presented. Good use of graph data in the report analysis.

5	9	10	Very good explanation of error analysis in report and detailed explanation of the theory. Could include a graph to show the convergence rate.
			Example:
			`a8[1].plot([0,1,2,3,4,5],np.ones(6), c='k', linewidth=2, alpha=0.7)`
			`a8[1].plot([1,2,3,4],rate, 'x', c=TUeRed, markersize=10)`
			`a8[1].set_ylim(0,1.5)`
			`a8[1].set_xlim(0.5,4.5)`
			`a8[1].set_xticks([1,2,3,4])`
			`a8[1].set_xticklabels(['40','80','160', '320'])`
			`a8[1].set_title('(b) Rate of Convergence at different Step Numbers')`
			`a8[1].set_xlabel('Step Number')`
			`a8[1].set_ylabel('Rate of Convergence')`
			`a8[1].grid()`

6	6	10	Great explanation of the steps for the ODE, however the analysis graphical is very limited. Code is simple, efficient and well structured. Visuals are insightful and all elements are present. Explain also how the python functions you use are relevant to answer the question with your code. The analysis should include an explanation for the trends such as that since the reaction

			is exothermic, as A reacts into B, it releases heat and thus the greatest increase in temperature correlates with the decrease in concentration.
7	9	10	Good comparison between question 6 and 7. Code is correct, pretty and efficient. Visuals are well done. The code works for an arbitrary number of equations as required. Explain also how the Python solver used are relevant for answering question.

8	6	10	More methods should be use to compare higher order solver as well as include a condition for an unrecognized method.
			For example:
			`dt = (tspan[1] - tspan[0])/number_of_points`
			`t = np.linspace(tspan[0], tspan[1], number_of_points+1)`
			`y = np.zeros((number_of_points+1, len(y0)))  # len(y0) because you would need an initial condition for each derivative.`
			`for i in range(len(y0)): #initial conditions as a loop to ensure universability.`
			`y[0,i] = y0[i]`
			`if method == 'midpoint':`
			`for i in range(number_of_points):`
			`k1 = fun(t[i], y[i,:])`
			`k2 = fun(t[i] + dt*0.5, y[i,:] + 0.5*dt*k1)`
			`y[i+1,:] = y[i,:] + dt * k2`
			`elif method == 'euler':`
			`for i in range(number_of_points):`
			`y[i+1,:] = y[i,:] + dt * fun(t[i], y[i,:])`
			`elif method == 'rk2':`
			`for i in range(number_of_points):`
			`k1 = fun(t[i], y[i,:])`
			`k2 = fun(t[i] + dt, y[i] + dt*k1)`
			`y[i+1,:] = y[i] + dt*0.5*(k1+k2)`
			`elif method == 'rk4':`
			`for i in range(number_of_points):`
			`k1 = fun(t[i], y[i,:])`
			`k2 = fun(t[i] + dt*0.5, y[i,:] + 0.5*dt*k1)`
			`k3 = fun(t[i] + dt*0.5, y[i,:] + 0.5*dt*k2)`
			`k4 = fun(t[i] +dt, y[i,:] + dt*k3)`
			`y[i+1,:] = y[i] + dt*((1/6)*k1 + (1/3)*(k2+k3) + (1/6)*k4)`
			`else:`
			`return 'Unknown method specified. Check documentation for supported methods' # In case an unknown method is specified`

			`return t, y` Good testing of the RK2 method. Code for RK2 is correct and well organized.
9	8	10	Correct error analysis visuals, good report analysis and great addition of table and graph to use as numerical quantitative data. Could explain the code as well in the report. Comparing with other higher order methods allow a deeper error analysis.
10	7	10	ODE function is solved with own solver, well organized code and very nice visuals. Could explain the chaos theory mentioned and include as well the conversion method for x and y in the report. Very little report analysis, should explain the graph with numerical and qualitative data.
11	7	10	The code does not compute the energy budget for different solvers and different amounts of steps. It however does compute the energy budget correctly for the RK2 method only for 50 steps. The theory of energy conservation is well explained. Great visuals with all elements included. Graphical analysis could draw more conclusions and educated recommendations. Further exploration is missing.

TIPS:

1. Split each task in a separate python file to allow an easier running.
2. Explain more the functions used in the code during the report explanation.
3. Try to implement more graphs and visuals whenever possible to support arguments made in the analysis;

TOPS:

1. very organized code with comments guiding through
2. great understanding of course theory
3. Visuals are insightful and include all the required elements. Very easy to understand.