COMPUTATIONAL PRACTICUM assignment

Differential Equations

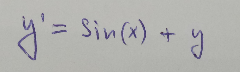
**Artur Akhmetshin**

**Group: BS17-05**

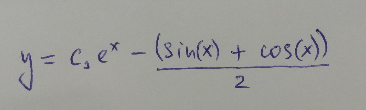
**y’ = sin(x) + y**

**Exact solution of IVP(Initial Value Problem)**

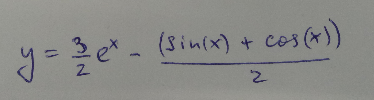
there x0 = 0 and y0 = 1



Given Differential Equation:



Solution of DE:

  
Solution of IVP:

There isn’t any point of discontinuity in solution of given differential equation.

System contains 3 classes: Equation, Numeric methods and supporting class Plotting

**UML diagram for Equation and Numeric methods**

|  |
| --- |
| **Equation** |
| **+x0 : Float**  **+y0 : Float**  **+X : Float**  Equation  For numeric methods  **+n : Integer**  **+h : Integer** |
| **+get\_derivative(x : Float, y : Float)**  **+\_\_init\_\_(x0 : Float, y0 : Float, X : Float, n : Integer)**  **+local\_errors(ff : Equation)**  **+max\_error(ff : Equation)**  **+exact\_solution()** |

|  |
| --- |
| **Numeric methods** |
|  |
| **+euler\_method(f : Equation)**  **+euler\_method\_improved(f : Equation)**  **+runge\_kutta\_method(f : Equation)**  **+total\_approximation\_errors(start\_n : Integer, end\_n : Integer, x0 : Float, y0 : Float, X : Float)** |

Implementation of numeric methods for equation

**Explanation of Equation class attributes and methods**

**Equation class is used to operate with given y’ = sin(x) + y differential equation, change initial values, grid size to solve IVP inside class.**

**Attributes:**

**+x0 : Float Starting point of x-axis of IVP problem segment**

**+y0 : Float Value of function in x0 point y(x0) = y0**

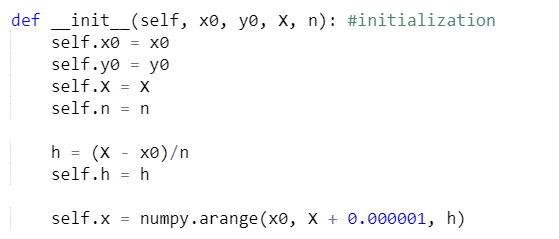
**+X : Float Ending point of x-axis of IVP problem segment**

**+n : Integer Grid size**

**+h : Integer Value of one grid step(depends on x0, X0 and n)**

**Methods:**

**+\_\_init\_\_(x0 : Float, y0 : Float, X : Float, n : Integer)**

****

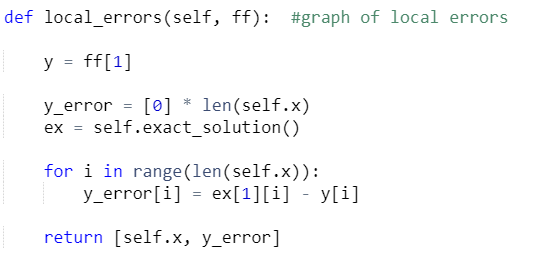
**Method initialize initial values, scope and grid size for solving IVP problem, these values are assigned to Equation instance**

**+get\_derivative(x : Float, y : Float)**

**C:\Users\Arthur\Downloads\init.png**

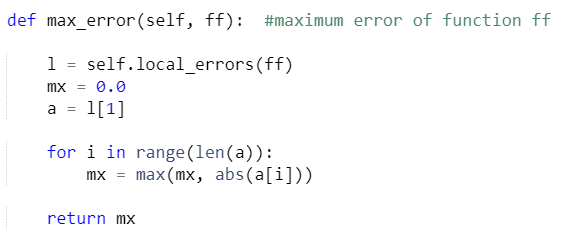
**Method returns local derivative of function in the point (x, y) using following function y’ = sin(x) + y**

**+local\_errors(ff : Equation)**

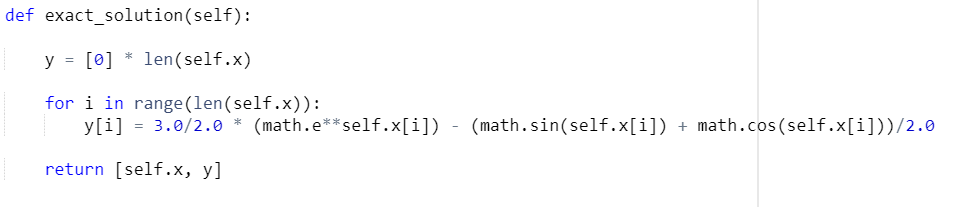
****

**Method calculates function of local errors of function ff with respect to exact solution of given differential equation**

**+max\_error(ff : Equation)**

**  
Method calculate value of the maximum error of function ff with respect to exact solution of given differential equation**

**+exact\_solution()**

****

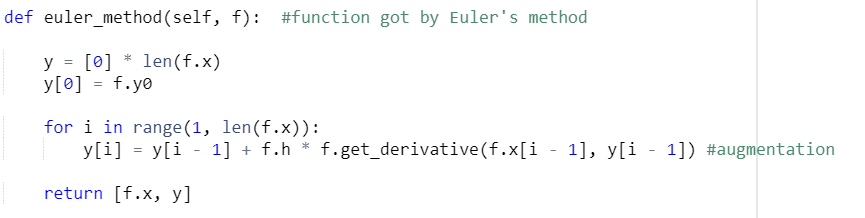
**Method calculate values of exact solution for given differential equation with determined scope and grid size in \_\_init\_\_**

**Explanation of Numeric\_methods class attributes and methods**

**Numeric\_methods class is used to operate solve IVP problem using Euler’s, Euler’ Improved and Runge-Kutta methods for given Equation e that is accepted by the class.**

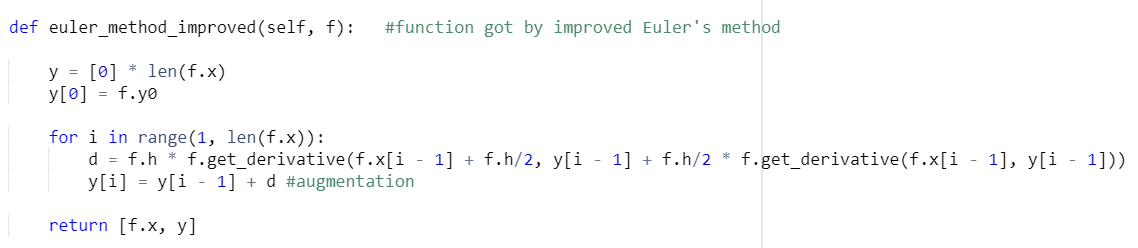
**Numeric\_methods has no any attribute.**

**+euler\_method(f : Equation)**

****

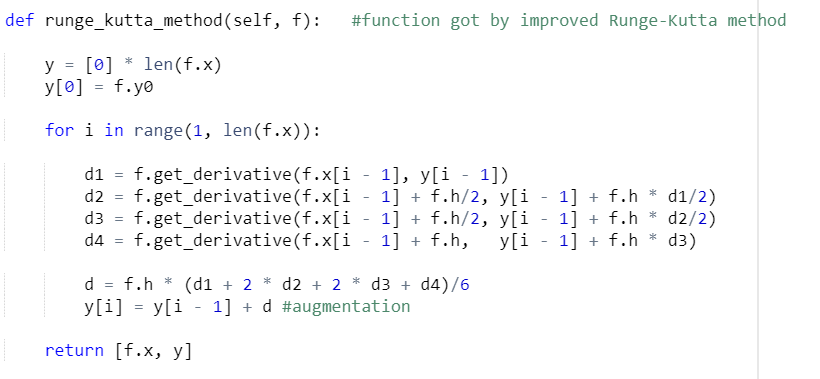
**Method calculate values of function of given differential equation ff on scope [x; X] with grid step n. It uses Euler’s method for calculations.**

**+euler\_method\_improved(f : Equation)**

****

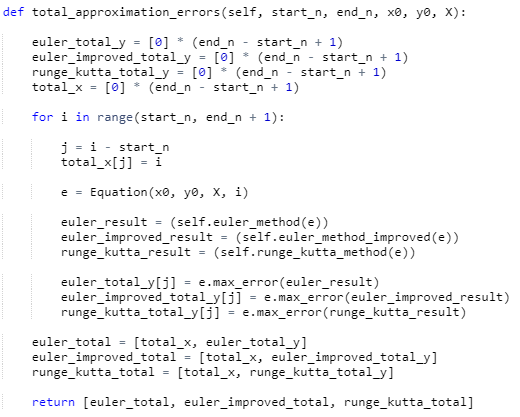
**Method calculate values of function of given differential equation ff on scope [x; X] with grid step n. It uses Improved Euler’s method for calculations.**

**+runge\_kutta\_method(f : Equation)**

****

**Method calculate values of function of given differential equation ff on scope [x; X] with grid step n. It uses Runge-Kutta method for calculations(calculating local derivatives, getting local result and augment it.**

**+total\_approximation\_errors(start\_n : Integer, end\_n : Integer, x0 : Float, y0 : Float, X : Float)**

****

**Method analyze the total approximation error depending on the number of grid cells, method accepts strat\_n – starting point of grid step and end\_n – finishing point of grid step and return functions of total errors of Euler’s, Improved Euler’s and Runge-Kutta methods for given differential equation y’ = sin(x) + y. It was implemented by using methods inside Numeric\_methods class for all integer grid steps values on [start\_n; end\_n] and calculating maximum error of taken functions.**

**Graph of solutions of differential equation y’ = sin(x) + y using Euler’s method, Improved Euler’s method, Runge-Kutta method and graph of exact solution.**

**Graphs of local errors, difference between exact solution of differential equation y’ = sin(x) + y and solutions calculated using Euler’s method, Improved Euler’s method, Runge-Kutta method.**

**Graph of total approximation errors of solutions of differential equation y’= sin(x) + y calculated using Euler’s method, Improved Euler’s method, Runge-Kutta method, depends on number of grid cells (x-axis).**