Question 1
Given
$y'(x) = -3x^2y^2$

Analytical solution:
$y'(x) = -3x^2y^2$
$\Rightarrow \frac{dy}{dx} = -3x^2y^2$
$\Rightarrow \frac{1}{y^2}dy = -3x^2dx$
$\Rightarrow \int \frac{1}{y^2}dy = \int -3x^2dx$
$\Rightarrow -\frac{1}{y} = -x^3 + C$
$\Rightarrow \frac{1}{y} = x^3 - C$
$\Rightarrow y = \frac{1}{x^3-C}$

Given Initial condition
$y(0) = 2$
$\Rightarrow \frac{1}{0^3-C} = 2$
$\Rightarrow -C = \frac{1}{2}$
$\Rightarrow C = -\frac{1}{2}$

Then,
$y = \frac{1}{x^3-C}$
$\Rightarrow y = \frac{1}{x^3--\frac{1}{2}}$
$\Rightarrow y = \frac{2}{2x^3+1}$

Euler's Modified Method (EMM)

Algorithm

Predictor

$y_p = y_i + h \times f(x_i, y_i)$

Corrector 1

$y_{c1} = y_i + \frac{h}{2}[f(x_i, y_i) + f(x_i + h, y_P)]$

Corrector 2

$y_{i+1} = y_i + \frac{h}{2}[f(x_i, y_i) + f(x_i + h, y_{c1})]$

Where

$f(x, y) = -3x^2y^2$

Algorithm Steps

1. Compute initial slope at beginning of interval

   $k_1 = f(x_n, y_n)$

2. First prediction

   $y^{(1)} = y_n + hk_1$

   Next value

   $y_n^{pred} = y_n + hk_1$

3. Correction 1

   $k_2 = f(x_{n+1}, y^{(1)})$

   $y^{(2)} = y_n + \frac{1}{2}h(k_1 + k_2)$

   Slope @ predicted point

   $k_2 = f(t_n + h, y_{n+1}^{pred})$

   Average slope for next value

   $y_{n+1} = y_n + \frac{1}{2}h(k_1 + k_2)$

4. Correction 2

   $k_3 = f(x_{n+1}, y^{(2)})$

5. Final value

   $y_{n+1} = y_n + \frac{1}{2}h(k_1 + k_3)$

Parameters

- Step sizes: $h = 0.2$ and $h = 0.1$
- Function: $f(t, y)$

```python
import numpy as np
import pandas as pd
import argparse

def get_equations(key: str):
    if key.lower() == "assignment1":
        def f(x, y):
            return -3 * x**2 * y**2
        def y_exact(x):
            return 2 / (2 * x**3 + 1)
        return f, y_exact
    else:
        raise ValueError(f"Unknown equation keyword: {key}")

def run_modified_euler(f, y_exact, h: float, y0: float = 2.0, x_end: float = 1.0) -> pd.DataFrame:
    x_vals = np.arange(0, x_end + h, h)
    y_vals = [y0]
    errors = [0]
    f_xy_vals = [f(0, y0)]
    y_p_vals = [np.nan]
    f_new_vals = [np.nan]
    y_new_vals = [y0]

    for i in range(1, len(x_vals)):
        x_i = x_vals[i - 1]
        y_i = y_vals[-1]

        f_xy = f(x_i, y_i)
        y_p = y_i + h * f_xy
        y_c1 = y_i + (h / 2) * (f_xy + f(x_i + h, y_p))
        y_next = y_i + (h / 2) * (f_xy + f(x_i + h, y_c1))
        f_new = f(x_i + h, y_c1)
        true_val = y_exact(x_vals[i])
        error = abs(true_val - y_next)

        y_vals.append(y_next)
        errors.append(error)
        f_xy_vals.append(f_xy)
        y_p_vals.append(y_p)
        f_new_vals.append(f_new)
        y_new_vals.append(y_next)

    return pd.DataFrame({
        'x': x_vals,
        'y': y_vals,
        'Exact y': y_exact(x_vals),
        'Error': errors,
        'f(x,y)': f_xy_vals,
        'yp': y_p_vals,
        'fnew': f_new_vals,
        'ynew': y_new_vals
    })

if __name__ == "__main__":
    parser = argparse.ArgumentParser(description="Modified Euler Method with 2 corrections")
    parser.add_argument('--equation', type=str, required=True,
                        help="Choose a predefined equation, e.g., 'assignment1'")
    parser.add_argument('--step-size', type=float, required=True,
                        help="Step size (e.g. 0.1)")
    parser.add_argument('--y0', type=float, default=2.0, help="Initial y value (default: 2.0)")
    parser.add_argument('--x-end', type=float, default=1.0, help="Final x value (default: 1.0)")
    args = parser.parse_args()
    f, y_exact = get_equations(args.equation)
    df = run_modified_euler(f, y_exact, args.step_size, args.y0, args.x_end)
    print(df.to_string(index=False))
```

```
python EMM.py --equation assignment1 --step-size 0.1
```

```
  x        y  Exact y    Error    f(x,y)       yp      fnew     ynew
0.0 2.000000 2.000000 0.000000  0.000000      NaN       NaN 2.000000
0.1 1.994036 1.996008 0.001972 -0.000000 2.000000 -0.119281 1.994036
0.2 1.964916 1.968504 0.003588 -0.119285 1.982107 -0.463111 1.964916
0.3 1.893422 1.897533 0.004111 -0.463307 1.918585 -0.966568 1.893422
0.4 1.770042 1.773050 0.003007 -0.967963 1.796626 -1.499641 1.770042
0.5 1.599271 1.600000 0.000729 -1.503864 1.619656 -1.911551 1.599271
0.6 1.398064 1.396648 0.001416 -1.918252 1.407446 -2.105897 1.398064
0.7 1.188615 1.186240 0.002375 -2.110949 1.186969 -2.078039 1.188615
0.8 0.990227 0.988142 0.002085 -2.076823 0.980932 -1.890925 0.990227
0.9 0.814805 0.813670 0.001136 -1.882656 0.801962 -1.625783 0.814805
1.0 0.666788 0.666667 0.000122 -1.613295 0.653476 -1.347045 0.666788
```

```
python EMM.py --equation assignment1 --step-size 0.1
```

```
  x        y  Exact y    Error    f(x,y)       yp      fnew     ynew
0.0 2.000000 2.000000 0.000000  0.000000      NaN       NaN 2.000000
0.2 1.954276 1.968504 0.014228 -0.000000 2.000000 -0.457236 1.954276
0.4 1.762801 1.773050 0.010249 -0.458304 1.862616 -1.456453 1.762801
0.6 1.407367 1.396648 0.010718 -1.491584 1.464484 -2.062758 1.407367
0.8 0.997893 0.988142 0.009750 -2.139135 0.979540 -1.955605 0.997893
1.0 0.662607 0.666667 0.004059 -1.911916 0.615509 -1.440936 0.662607
```

Question 2

Given

$\frac{dy}{dx} = 3x + 2y + xy$

$y(0) = -1$

$y(x) = y(0) + y'(0)x + \frac{y''(0)}{2!}x^2 + \frac{y^{(3)}(0)}{3!}x^3 + \cdots + \frac{y^{(6)}(0)}{6!}x^6$

Derivatives at $x = 0$

$y' = f(x, y) = 3x + 2y + xy$

$y' = f(x, y) = 3x + (2 + x)y$	$y'(0) = 2(-1) = \ -2$
$y'' = f(x, y) = 3 + y + (2 + x)y'$	$y''(0) = 3 - 1 + 2(-2) \ - 2$
$y''' = f(x, y) = 2y' + 2y + (2 + x)y''$	$y'''(0) = 2(-2) + 2(-2) = \ -8$
$y^4 = f(x, y) = 3y'' + 2y + (2 + x)y'''$	$y^4(0) = 3(-2) + 2(-8) = -22$
$y^5 = f(x, y) = 4y''' + 2y + (2 + x)y^4$	$y^5(0) = 4(-8) + 2(-22) = -76$
$y^6 = f(x, y) = 5y^4(2 + x)y^5$	$y^6(0) = 5(-22) + 2(-76) = -262$

Compute at $x = 0.1$

$y(x) = y(0) + y'(0)x + \frac{y''(0)}{2!}x^2 + \frac{y^{(3)}(0)}{3!}x^3 + \cdots + \frac{y^{(6)}(0)}{6!}x^6$

$\Rightarrow y(0.1) = -1 + (-2)(0.1) + \frac{-2}{2!}(0.1)^2 + \frac{-8}{3!}(0.1)^3 + \frac{-22}{4!}(0.1)^4 + \frac{-76}{5!}(0.1)^5 + \frac{-262}{6!}$
$(0.1)^6$

$\Rightarrow y(0.1) \approx -1.21143$

Question 3

Given

$u'' - (t + 1)\,uv + v' = cost$

$v'' = u' + uv$

$u(0) = 2$

$u'(0) = 1$

$v(0) = 3$

$v'(0) = -1$

Convert to First-Order System

$x_0 = u$

$x_1 = u'$

$x_2 = v$

$x_3 = v'$

Then,

$x_0{}' = x_1$

$x_2{}' = x_3$

$x_1' = (t + 1)x_0x_2 - x_3 + cost$

$x_3' = x_1 + x_0x_2$

And,

$x_0(0) = 2$

$x_1(0) = 3$

$x_2(0) = 1$

$x_3(0) = -1$

2nd Order Runge-Kutta Method (General Form)

Given

$$h = 0.2$$
$$a = \frac{2}{3}$$
$$b = \frac{1}{3}$$
$$\alpha = \frac{3}{2}$$
$$\beta = \frac{3}{2}$$

Then,

$$k_1 = h \times f(t_n, y_n)$$
$$k_2 = h \times f(t_n + \frac{3}{2}h, y_n + \frac{3}{2}k_1)$$

$$y_{n+1} = y_n + \alpha k_1 + \beta k_2$$

Applying Algorithm

Initial values

$h = 0.2$

$t_n = 0$

$x_{0_n} = 2$

$x_{2_n} = 1$

$x_{1_n} = 3$

$x_{3_n} = -1$

Compute k1 values

$kx_0 0 = h \times U(t_n, x0_n, x1_n, x2_n, x3_n)$

$\Rightarrow kx_{0_2} = 0.2 \times U(0, 2, 3, 1, -1) = 0.2$

$kx_1 = h * V(t_n, x0_n, x1_n, x2n, x3_n)$

$\Rightarrow kx_{1_2} = 0.2 * U(0, 2, 3, 1, -1) = 0.2$

$kx_2 = h * W(t_n, x0_n, x1_n, x2_n, x3_n)$

$\Rightarrow kx_{2_2} = 0.2 * U(0, 2, 3, 1, -1) = 0.2 \times 8 = 1.6$

$kx_3 = h * X(t_n, x0_n, x1_n, x2_n, x3_n)$

$\Rightarrow kx_{3_2} = 0.2 * U(0, 2, 3, 1, -1) = 0.2 \times 7 = 1.4$

Intermediate step values for k2

$t_{half} = t_n + \left(\frac{3}{2}\right) \times h = 0 + 0.3 = 0.3$

$x_{0_{half}} = x_{0_n} + \left(\frac{3}{2}\right) \times k = 1 + \frac{3}{2}(0.2) = 2.3$

$x_{1_{half}} = x_{1_n} + \left(\frac{3}{2}\right) \times kx_1 = 1 + \frac{3}{2} \quad \left(\frac{1}{6}\right) = 3.4$

$x_{2_{half}} = x_{2_n} + \left(\frac{3}{2}\right) \times kx_2 = 3 + \frac{3}{2}(-0.2) = 2.7$

$x_{3_{half}} = x_{3_n} + \left(\frac{3}{2}\right) \times kx_3 = -1 + \frac{3}{2}(1.4) = 1.1$

Compute k2 values

$kx_{0_2} = h \times U(t_{half}, x_{0_{half}}, x_{1_{half}}, x_{2_{half}}, x_{3_{half}})$

$\Rightarrow 0.2 \times 3.4 = 0.68$

$kx_{1_2} = h \times V(t_{half}, x_{0_{half}}, x_{1_{half}}, x_{2_{half}}, x_{3_{half}})$

$\Rightarrow (0.3 + 1)(2.3)(2.7) - 1.1 + \cos(0.3) = 8.073 - 1.1 + 0.9553 = 1.5857$

$kx_{2_2} = h \times W(t_{half}, x_{0_{half}}, x_{1_{half}}, x_{2_{half}}, x_{3_{half}})$

$\Rightarrow 0.2 \times 1.1 = 0.22$

$kx_{3_2} = h \times X(t_{half}, x_{0_{half}}, x_{1_{half}}, x_{2_{half}}, x_{3_{half}})$

$\Rightarrow 0.2 \times (3.4 + 2.3 \times 2.7) = 1.922$

Final values using weighted combination, at t=0.2

$x_{0_2} = x_{0_n} + \left(\frac{2}{3}\right) * kx_0 + \left(\frac{1}{3}\right) * kx_{0_2}$

$\Rightarrow 2 + 32(0.2) + 31(0.68) = 2 + 0.1333 + 0.2267 = 2.36$

$x_{1_2} = x_{1_n} + \left(\frac{2}{3}\right) * kx_1 + \left(\frac{1}{3}\right) * kx_{1_2}$

$\Rightarrow 1 + 32(1.6) + 31(1.5857) = 1 + 1.0667 + 0.5286 = 2.5953$

$x_{2_2} = x_{2_n} + \left(\frac{2}{3}\right) * kx_2 + \left(\frac{1}{3}\right) * kx_{2_2}$

$\Rightarrow 3 + 32(-0.2) + 31(0.22) = 3 - 0.1333 + 0.0733 = 2.94$

$x_{3_2} = x_{3_n} + \left(\frac{2}{3}\right) * kx_3 + \left(\frac{1}{3}\right) * kx_{3_2}$

$\Rightarrow -1 + 32(1.4) + 31(1.922) = -1 + 0.9333 + 0.6407 = 0.574$