Q1:

Newton's Method for solving $W_{i+1} = hf(t_{i+1}, W_{i+1})$

and we know that $y' = f(t, y)$, $a \le t \le b$, $y(a) = \alpha$

$g(W_{i+1}) = 0$, $g(W_{i+1}) = W_{i+1} - W_i - hf(t_{i+1}, W_{i+1})$

and then $W_{i+1}^{(k+1)} = W_{i+1}^{(k)} - \dfrac{g(W_{i+1}^{(k)})}{g'(W_{i+1}^{(k)})}$

$g'(W_{i+1}) = 1 - h\dfrac{\partial f}{\partial y}(t_{i+1}, W_{i+1})$

And we can know that $W_{i+1}^{(0)} = W_i$

$$W_{i+1}^{(k+1)} = W_{i+1}^{(k)} - \frac{W_{i+1}^{(k)} - W_i - hf(t_{i+1}, W_{i+1}^{(k)})}{1 - h\frac{\partial f}{\partial y}(t_{i+1}, W_{i+1}^{(k)})}$$

Q2:

```
function [t, w] = backeuler(f, dfdy, a, b, alpha, N, maxiter, tol)
  h = (b - a) / N;            % Step size
  t = linspace(a, b, N+1);    % Time vector
  w = zeros(1, N+1);          % Initialize w to store y values
  w(1) = alpha;               % Initial condition

  % Iterate over each time step
  for i = 1:N
      fprintf('Time step %d, t = %f\n', i, t(i+1));
      fprintf('Iter     w(i+1)           |delta|\n');
      fprintf('--------------------------------\n');

      w_next = w(i); % Initial guess for Newton's method
      for j = 1:maxiter
          g = w_next - w(i) - h * f(t(i+1), w_next);
          g_prime = 1 - h * dfdy(t(i+1), w_next);
          delta = -g / g_prime;
          w_next = w_next + delta;  % Update w_next using Newton's method

          % Print the value of Newton updates
          fprintf('%4d %12.8f %12.8e\n', j, w_next, abs(delta));

          if abs(delta) < tol  % Check for convergence
              fprintf('Convergence achieved at iteration %d.\n', j);
              fprintf('--------------------------------\n');
              break;
          end
      end
      % Check if maximum iterations were exceeded without convergence
      if j == maxiter && abs(delta) >= tol
          error('Maximum iterations reached without convergence at t = %f', t(i+1));
      end
      w(i+1) = w_next; % Assign converged value to solution
  end
end
```

Q3: (a)

Q3

(a) We know that $y' = y^2(1-y)$, $y' = \lambda y$

$\lambda = \frac{\partial f}{\partial y} = 2y - 3y^2 \approx -1$, and we know that $h\lambda = -2.7853$.

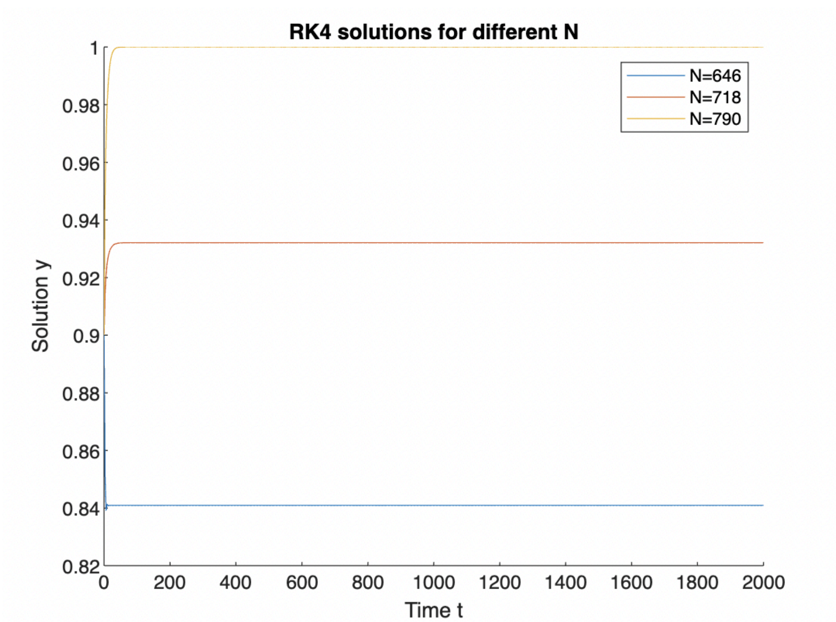
$y(t) \approx 1$, $\lambda \approx -1$, $h = \frac{-2.7853}{\lambda} \approx 2.7853$

Since $0 \le t \le 2000$, We estimate N as

$N = \frac{2000}{h} = \frac{2000}{2.7853} \approx 718$.

(b)

```
% Function definition
  f = @(t, y) y^2 * (1 - y);

  % Parameters
  a = 0; b = 2000; y0 = 0.9;
  N = round(718 * [0.9, 1, 1.1]);  % 10% below, exact, and 10% above

  % Solve and plot
  figure;
  hold on;
  for n = N
     [t, y] = rk4(f, a, b, y0, n);
     plot(t, y, 'DisplayName', sprintf('N=%d', n));
     fprintf('y(2000) with N = %d: %f\n', n, y(end));
  end
  legend show;
  xlabel('Time t');
  ylabel('Solution y');
  title('RK4 solutions for different N');
y(2000) with N = 646: 0.840919
y(2000) with N = 718: 0.932113
y(2000) with N = 790: 1.000000
```



(c)

(c)

The Backward Euler Method formula is

$y_{n+1} = y_n + hf(t_{n+1}, y_{n+1})$

and we can know that $y' = \lambda y$

$y_{n+1} = y_n + \lambda y_{n+1}$

$y_{n+1}(1-h\lambda) = y_n$

$y_{n+1} = \frac{y_n}{1-h\lambda}$

To determine stability: $\left|\frac{1}{1-h\lambda}\right|$

For stability => $Re(\lambda) < 0$, "$1-h\lambda$" is positive

Hence $\left|\frac{1}{1-h\lambda}\right| \leq 1$

So we can know that Backward Euler method is stable for all λ with negative real part and it's proving its A-stability

And we can see the plot. The numerical solution is closest to the expected value $y(2000) \approx 1$ when $N = 790$, indicating that the solution is stable and accurate.

(d)

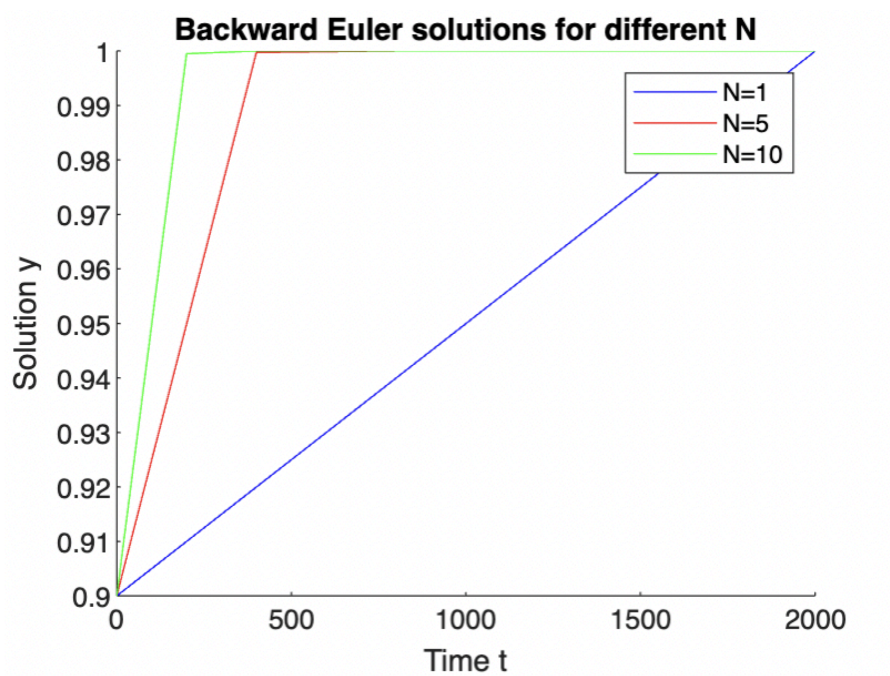
```
% Define the differential equation and its derivative
f = @(t, y) y^2 * (1 - y);
dfdy = @(t, y) 2*y * (1 - y) - y^2;

a = 0;
b = 2000;
alpha = 0.9;

% Set parameters
maxiter = 20; % Maximum number of iterations
tol = 1e-12;     % Tolerance

% Solve and plot results for different step counts
N_values = [1, 5, 10];
colors = ['b', 'r', 'g'];

figure;
hold on;
for j = 1:length(N_values)
   N = N_values(j);
   [t, w] = backeuler(f, dfdy, a, b, alpha, N, maxiter, tol);
```

```
    plot(t, w, colors(j), 'DisplayName', ['N=' num2str(N)]);
  end
  xlabel('Time t')
  ylabel('Solution y')
  title('Backward Euler solutions for different N')
  legend show
  hold off;
```

```
Time step 1, t = 2000.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.02846947 1.28469469e-01
  2   1.00144803 2.70214410e-02
  3   0.99995449 1.49354135e-03
  4   0.99995002 4.46627009e-06
  5   0.99995002 3.98803134e-11
  6   0.99995002 7.88715553e-18
Convergence achieved at iteration 6.
--------------------------------
Time step 1, t = 400.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.02806324 1.28063241e-01
  2   1.00122574 2.68375012e-02
  3   0.99975482 1.47091521e-03
  4   0.99975050 4.32538431e-06
  5   0.99975050 3.73479696e-11
  6   0.99975050 4.40305288e-17
Convergence achieved at iteration 6.
--------------------------------
Time step 2, t = 800.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   0.99999950 2.49002201e-04
  2   0.99999938 1.23664308e-07
  3   0.99999938 3.05021665e-14
Convergence achieved at iteration 3.
--------------------------------
Time step 3, t = 1200.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.00000000 6.20646271e-07
  2   1.00000000 7.68530805e-13
Convergence achieved at iteration 2.
--------------------------------
Time step 4, t = 1600.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.00000000 1.54774437e-09
  2   1.00000000 1.16283010e-17
Convergence achieved at iteration 2.
--------------------------------
Time step 5, t = 2000.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.00000000 3.85970004e-12
  2   1.00000000 9.69022590e-18
Convergence achieved at iteration 2.
--------------------------------
```

```
Time step 1, t = 200.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.02755906 1.27559055e-01
  2   1.00094931 2.66097474e-02
  3   0.99950615 1.44315776e-03
  4   0.99950199 4.15576182e-06
  5   0.99950199 3.44114912e-11
  6   0.99950199 1.74335599e-17
Convergence achieved at iteration 6.
--------------------------------
Time step 2, t = 400.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   0.99999801 4.96017592e-04
  2   0.99999752 4.89378363e-07
  3   0.99999752 4.76576418e-13
Convergence achieved at iteration 3.
--------------------------------
Time step 3, t = 600.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   0.99999999 2.46533881e-06
  2   0.99999999 1.20952488e-11
  3   0.99999999 3.50426186e-17
Convergence achieved at iteration 3.
--------------------------------
Time step 4, t = 800.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.00000000 1.22653076e-08
  2   1.00000000 2.59059524e-16
Convergence achieved at iteration 2.
--------------------------------
Time step 5, t = 1000.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.00000000 6.10213925e-11
  2   1.00000000 1.76751926e-17
Convergence achieved at iteration 2.
--------------------------------
Time step 6, t = 1200.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.00000000 3.03571430e-13
Convergence achieved at iteration 1.
--------------------------------
Time step 7, t = 1400.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.00000000 1.54657934e-15
Convergence achieved at iteration 1.
--------------------------------
Time step 8, t = 1600.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.00000000 0.00000000e+00
Convergence achieved at iteration 1.
--------------------------------
```

```
Time step 9, t = 1800.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.00000000 0.00000000e+00
Convergence achieved at iteration 1.
--------------------------------
Time step 10, t = 2000.000000
Iter    w(i+1)          |delta|
--------------------------------
  1   1.00000000 0.00000000e+00
Convergence achieved at iteration 1.
--------------------------------
```



Yes, according to the picture we know that the method shows stability for N=5 and N=10, but for N=1 it is not optimally stable over a wide range of time intervals [0,2000], so increasing N improves stability and accuracy.