```
0;

function x = solvetridiagonal(b, d1, d2, d3)
  % n is the length of the center diagonal
  n = length(d2);

  % Scan down to reduce the lower diagonal to zero and the pivots to 1.
  for i = 1:n
    % Rescale the row so the pivot is 1.
    a = d2(i);
    d2(i) /= a;
    if i < n
      d3(i) /= a;
    end
    b(i) /= a;

    % Cancel out all non zero elements in the column except for the pivot.
    if i < n
      factor = -d1(i);
      d1(i) += factor;
      d2(i+1) += factor * d3(i);
      b(i+1) += factor * b(i);
    end
  end

  % Scan back upwards and reduce the upper diagonal to zero and thus solving for x.
  for i = flip(2:n)
    factor = -d3(i-1);
    d3(i-1) += factor;
    b(i-1) += factor * b(i);
  end

  % Finally set x = b
  x = b;
end

% 2.a

% The input matrix is n x n. b is in R^n
n = 10000;
% construct the 3 diagaonals
d1 = - (2:n);
d2 = 3 * (1:n);
d3 = - (1:n-1);
% pick b to be something
b = 1:n;

x = solvetridiagonal(b, d1, d2, d3);

% 2.b

% Define the function g from the textbook.
function gt = g(t)
  gt = (pi/2)^2*sin(pi/2 * t);
end

% n is number of steps
n = 100;
% h is step size
h = 1/n;

% create the tridiagonal matrix representing our problem
d1 = repmat(-1/h^2, 1, n-1);
d2 = repmat(2/h^2, 1, n);
d3 = repmat(-1/h^2, 1, n-1);

% create the b values
b = g((1:n) * h);

% solve for v of Dv=b
v = solvetridiagonal(b, d1, d2, d3);

% check against matlab
```

```
d = full(gallery('tridiag', n, -1/h^2, 2/h^2, -1/h^2));
vright = (d\(b'))';
% likely will vary slightly, but not greatly
assert(norm(v - vright, 2) < 1e-13);


% compute actual answer
u = sin(pi/2 * (1:n) * h);

% compute norm

norm(v'-u', inf)
norm(vright'-u', inf)
```