Problems 1 through 6 are worth 10 points each. Problem 7 is worth 15 points for a total of 75. *All work must be shown for full credit.* Please begin each problem on a new page and label what problem you are working on. No outside resources are permitted.

1

Consider the following MATLAB code. You may assume that `a` and `b` are positive integers.

```
%beginning of function
function out = mystery(a,b)
    if a < b %want a to be greater than or equal to b
        temp = b; b = a; a = temp;
    end
    while mod(a,b) != 0 %run until a is divisible by b
        temp = b;
        b = mod(a,temp);
        a = temp;
    end
    out = b;
end
%end of function
```

a) One of the lines above has a syntax error that will not allow the function to be run. Identify the error and briefly explain how to correct it.

b) If `c = mystery(32,12)`, what is the value of `c`? Explain how you came to your answer.

2

a) Using the composite trapezoid rule, numerically estimate the integral

$$\int_0^4 (x-1)^2 dx$$

using uniformly spaced nodes at each integer in the domain of integration.

b) Compute the error from using the scheme in part a. Then, give two *different* ways to more accurately estimate this integral numerically.

3

a) Define what is meant by *cubic spline* and, for these, *natural* and *not-a-knot* conditions.

b) Determine the parameters a, b, c, d, and e so that the following function $s(x)$ is a *natural* cubic spline on the interval $[0, 2]$ with knots at $x = 0, 1$, and 2.

$$s(x) = \begin{cases} a + b(x-1) + c(x-1)^2 + d(x-1)^3 & 0 \le x \le 1 \\ (x-1)^3 + ex^2 - 1 & 1 \le x \le 2 \end{cases}$$

—Exam continued on back—

4

Suppose $c > 0$ and we want to numerically approximate the square root of c using Newton's method. In other words, we wish to numerically find the (positive) root of $f(x) = x^2 - c$ using the scheme:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

a) Prove that the error $e_n = x_n - \sqrt{c}$ satisfies the recurrence relation

$$e_{n+1} = \frac{1}{2} \frac{e_n^2}{e_n + \sqrt{c}}$$

b) Using the relation from part a, determine the order of convergence assuming the initial guess is sufficiently close to the root.

5

a) Write a MATLAB function `lagbasis(nodes,j,x)` that has inputs

- a nonempty vector of x-coordinates `nodes` (representing interpolation nodes)
- a positive integer `j` less than or equal to `length(nodes)` (representing an index for the above vector)
- a real number `x` that is not necessarily an element of `node`

and returns the value of the jth *Lagrange basis function* evaluated at x, i.e. calculates $\ell_j(x)$.

b) Write MATLAB code that would make a plot that contains a curve representing the Lagrange polynomial interpolating the points (-1,1), (0, 0), and (2,4). You may call your function from the previous part. **Bonus**: add code to label the axes of the plot (any name you want) and make the Lagrange polynomial curve red.

6

a) Derive the fourth order accurate centered difference approximation for the first derivative of a function $f(x)$, assuming uniform grid spacing h.

b) The third order accurate expression for the first derivative of $f(x)$ given by forward differences is

$$f'(x_0) = \frac{-\frac{11}{6}f(x_0) + 3f(x_1) - \frac{3}{2}f(x_2) + \frac{1}{3}f(x_3)}{h}$$

Here, $x_j = x_0 + jh$, where j is the spacing between adjacent nodes. What is the third order accurate expression for the first derivative of $f(x)$ at $x = x_0$ given by *backward* differences?

7

Clearly indicate if each statement is true or false. No work needs to be shown for this section.

a) The best rootfinding algorithm is always the one of highest order.
b) Two shortcomings of polynomial interpolation include Gibbs's phenomenon and aliasing error.
c) In the Gaussian quadrature approximation of $\int_a^b f(x)dx$, the sum of the weights w_i always equals $b-a$.
d) The error due to polynomial interpolation can be minimized by using Chebyshev nodes.
e) The midpoint rule and the trapezoid rule are the same order of accuracy.

—End of exam—