1. The table below shows that the degree of precision of the given quadrature formula is 3:

<table>
<thead>
<tr>
<th>$f(x)$</th>
<th>$\int_{-1}^{1} f(x) , dx$</th>
<th>$f(-\sqrt{3}/3) + f(\sqrt{3}/3)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$x$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$x^2$</td>
<td>$2/3$</td>
<td>$2/3$</td>
</tr>
<tr>
<td>$x^3$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$x^4$</td>
<td>$2/5$</td>
<td>$2/9$</td>
</tr>
</tbody>
</table>
2. (a) \( J_1 = \int_{0.1}^{1} \sqrt{1 + x} \, dx \approx 1.11649159 \)

| \( n \) | open/closed | \( I \) | \( |J_1 - I| \) | bound          |
|-------|-------------|------|-------------|---------------|
| 1     | closed      | 1.10836008 | 8.13 \times 10^{-3} | 1.24 \times 10^{-1} |
| 2     | closed      | 1.11644734 | 4.43 \times 10^{-5} | 1.63 \times 10^{-2} |
| 3     | closed      | 1.11647157 | 2.00 \times 10^{-5} | 1.58 \times 10^{-3} |
| 4     | closed      | 1.11649123 | 3.66 \times 10^{-7} | 2.55 \times 10^{-4} |
| 0     | open        | 1.12049096 | 4.00 \times 10^{-3} | 1.97 \times 10^{-2} |
| 1     | open        | 1.11917540 | 2.68 \times 10^{-3} | 5.85 \times 10^{-3} |
| 2     | open        | 1.11652963 | 3.80 \times 10^{-5} | 3.87 \times 10^{-4} |
| 3     | open        | 1.11651825 | 2.67 \times 10^{-5} | 1.27 \times 10^{-4} |

(b) \( J_2 = \int_{0}^{\pi/2} \sin^2(x) \, dx \approx 0.78539816 \)

| \( n \) | open/closed | \( I \) | \( |J_1 - I| \) | bound          |
|-------|-------------|------|-------------|---------------|
| 1     | closed      | 0.78539816 | \varepsilon/2 | 7.75          |
| 2     | closed      | 0.78539816 | 3\varepsilon/2 | 2.39          |
| 3     | closed      | 0.78539816 | 0            | 3.15 \times 10^{-1} |
| 4     | closed      | 0.78539816 | 3\varepsilon/2 | 4.61 \times 10^{-2} |
| 0     | open        | 0.78539816 | \varepsilon   | 9.69 \times 10^{-1} |
| 1     | open        | 0.78539816 | 0            | 2.87 \times 10^{-1} |
| 2     | open        | 0.78539816 | \varepsilon   | 7.47 \times 10^{-2} |
| 3     | open        | 0.78539816 | \varepsilon/2 | 2.45 \times 10^{-2} |

...very curious.

(c) \( J_3 = \int_{1}^{10} \frac{1}{x} \, dx \approx 2.30258509 \)

| \( n \) | open/closed | \( I \) | \( |J_1 - I| \) | bound          |
|-------|-------------|------|-------------|---------------|
| 1     | closed      | 4.95000000 | 2.65        | 1.46 \times 10^{3} |
| 2     | closed      | 2.74090909 | 4.38 \times 10^{-1} | 4.43 \times 10^{4} |
| 3     | closed      | 2.56339286 | 2.61 \times 10^{-1} | 5.83 \times 10^{3} |
| 4     | closed      | 2.38570043 | 8.31 \times 10^{-2} | 2.10 \times 10^{5} |
| 0     | open        | 1.63636364 | 6.66 \times 10^{-1} | 1.82 \times 10^{2} |
| 1     | open        | 1.76785714 | 5.35 \times 10^{-1} | 5.40 \times 10^{1} |
| 2     | open        | 2.07489285 | 2.28 \times 10^{-1} | 1.38 \times 10^{3} |
| 3     | open        | 2.11637856 | 1.86 \times 10^{-1} | 4.53 \times 10^{2} |

The interval is wide, so the error bounds aren’t much good.
(d) \( J_4 = \int_0^1 x^{1/3} \, dx = 0.75 \)

| n | open/closed | \( I \) | \(|J_4 - I|\) | bound |
|---|-------------|--------|-------------|--------|
| 1 | closed      | 0.50000000 | 2.50 \times 10^{-1} | \( \infty \) |
| 2 | closed      | 0.69580035  | 5.42 \times 10^{-2} | \( \infty \) |
| 3 | closed      | 0.71260315  | 3.74 \times 10^{-2} | \( \infty \) |
| 4 | closed      | 0.73063414  | 1.94 \times 10^{-2} | \( \infty \) |
| 0 | open        | 0.79370053  | 4.37 \times 10^{-2} | \( \infty \) |
| 1 | open        | 0.78347087  | 3.35 \times 10^{-2} | \( \infty \) |
| 2 | open        | 0.76111371  | 1.11 \times 10^{-2} | \( \infty \) |
| 3 | open        | 0.75935723  | 9.36 \times 10^{-3} | \( \infty \) |

The error bounds are infinite because the derivatives of \( x^{1/3} \) are unbounded at 0.
3. Estimate \( \int_0^2 \frac{1}{x^4} \, dx \) to within \( 10^{-5} \).

On the interval \([0, 2]\), \( f''(\xi) \) is bounded by \( 1/32 \) and \( f^{(4)}(\xi) \) is bounded by \( 3/128 \). The needed values for \( h \) then follow from Theorems (4.4-4.6).

<table>
<thead>
<tr>
<th></th>
<th>open/closed</th>
<th>required ( h )</th>
<th>approximation</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>closed</td>
<td>.0438</td>
<td>.40547058</td>
<td>( 5.47 \times 10^{-6} )</td>
</tr>
<tr>
<td>2</td>
<td>closed</td>
<td>.443</td>
<td>.40546637</td>
<td>( 1.27 \times 10^{-6} )</td>
</tr>
<tr>
<td>0</td>
<td>open</td>
<td>.0310</td>
<td>.40545979</td>
<td>( 5.31 \times 10^{-6} )</td>
</tr>
</tbody>
</table>

According to these error bounds, then, using Simpson’s Rule would allow us to accurately estimate the integral with many fewer function evaluations than with the Trapezoid or Midpoint rules.
4. (a) Plugging in \( f(x) = e^{\lambda x} \) gives the formula
\[
\frac{e^{\lambda} - 1}{\lambda} = \frac{1}{2} \left( e^{\lambda} + 1 \right) + \sum_{m=1}^{\infty} b_m \lambda^{2m-1} \left( e^{\lambda} - 1 \right).
\]
Multiply both sides by \( \lambda (e^{\lambda} - 1)^{-1} \) to yield
\[
-\frac{\lambda}{e^{\lambda} - 1} = -1 + \frac{\lambda}{2} + \sum_{m=1}^{\infty} b_m \lambda^{2m}.
\]
(1)
Therefore \( b_m \) is the Taylor coefficient of the function
\[
\frac{-\lambda}{e^{\lambda} - 1}
\]
corresponding to \( \lambda^{2m} \). Let’s set this up in the form
\[
\left( e^{\lambda} - 1 \right) \left[ -1 + \frac{\lambda}{2} + \sum_{m=1}^{\infty} b_m \lambda^{2m} \right] = -\lambda
\]
and define a new power series \( \sum_{j} \alpha_j \lambda^j \) where \( \alpha_0 = -1, \alpha_1 = 1/2, \) and \( b_m = \alpha_{2m} \). Then divide the power series for \( e^{\lambda} - 1 \) by \( \lambda \) to get the relation
\[
\left( \sum_{k=0}^{\infty} \frac{\lambda^k}{(k+1)!} \right) \left( \sum_{j=0}^{\infty} \alpha_j \lambda^j \right) = -1.
\]
We may rearrange this sum in the form
\[
\sum_{s=0}^{\infty} \sum_{t=0}^{s} \alpha_t \lambda^{s-t} \frac{1}{(s-t+1)!} = -1,
\]
and equating terms gives
\[
\sum_{t=0}^{s} \frac{\alpha_t}{(s-t+1)!} = \begin{cases} 
-1 & s = 0 \\
0 & \text{otherwise}
\end{cases}
\]
Equation (1) implies that \( \alpha_j = 0 \) for all odd \( j \) greater than 1, so we can eliminate those indices, separate out \( \alpha_0 \) and \( \alpha_1 \) from the rest of the sum, and then substitute \( \alpha_{2j} = b_j \) to get
\[
-\frac{1}{(2m+1)!} + \frac{1}{2(2m)!} + \sum_{k=1}^{m} \frac{b_k}{(2(m-k)+1)!} = 0.
\]
Subtracting the final term of the sum from both sides gives the desired recurrence:
\[
b_m = \frac{1}{(2m+1)!} - \frac{1}{2(2m)!} - \sum_{k=1}^{m-1} \frac{b_k}{(2(m-k)+1)!}
\]
(2)
(b) If we program relation (2) in Matlab then we will start losing precision after the first few numbers and won’t get exact rational numbers. In any case, the exact values of the first ten numbers \( b_1, \ldots, b_{10} \) are
\[
\begin{align*}
    b_1 &= -\frac{1}{12}, \\
    b_2 &= \frac{1}{720}, \\
    b_3 &= -\frac{1}{30240}, \\
    b_4 &= \frac{1}{1209600}, \\
    b_5 &= -\frac{1}{47900160}, \\
    b_6 &= \frac{1}{130767436800}, \\
    b_7 &= -\frac{1}{747242496000}, \\
    b_8 &= \frac{3617}{174611}, \\
    b_9 &= -\frac{43867}{5109094217170944000}, \\
    b_{10} &= \frac{802857662698291200000}{174611}.
\end{align*}
\]
5. Before tackling the problem, let us re-examine the Euler–Maclaurin expansion. Note that by a change
of variable \( x \mapsto j + x \), the same formula holds on any interval of length one:

\[
\int_j^{j+1} f(x) \, dx = \frac{1}{2} (f(j+1) + f(j)) + \sum_{m=1}^{\infty} b_m \left( f^{(2m-1)}(j+1) - f^{(2m-1)}(j) \right).
\]

Also note that we can write

\[
\int_0^n f(x) \, dx = \sum_{j=0}^{n-1} \int_j^{j+1} f(x) \, dx.
\]

Summing up the corresponding sides of the Euler–Maclaurin formulae for each \( j \) and noting the can-
cellation, we get

\[
\int_0^n f(x) \, dx = -\frac{1}{2} (f(0) + f(n)) + \sum_{j=0}^{n} f(j) + \sum_{m=1}^{\infty} b_m \left( f^{(2m-1)}(n) - f^{(2m-1)}(0) \right),
\]

valid for arbitrary \( f \).

(a) Apply the formula (3) to the function \( f(x) = x^k \); the term \( \sum_{j=0}^{n} f(j) \) on the right hand side is
then exactly what we are after. It suffices to show that the other terms are polynomials in \( n \) of
degree at most \( k + 1 \). This is clearly true of the integral on the left hand side, since it evaluates
to \( (k + 1)^{-1} n^{k+1} \). The first term on the right hand side gives

\[-\frac{1}{2} (f(0) + f(n)) = -\frac{n^k}{2}.
\]

Finally, the sum involving \( b_m \) is also polynomial in \( n \) of degree at most \( k - 1 \), since \( f(x) = x^k \) is
always differentiated at least one time.

(b) This is a direct application of the formula (3), but is otherwise quite tedious. For a list of the
first 10 polynomials, see https://en.wikipedia.org/wiki/Faulhaber’s_formula.

(c) We can find the formula for \( P_{k+1}(n) \) by using Lagrange interpolation on the nodes 0, 1, 2, \ldots, \( k+1 \):

\[
P_{k+1}(n) = \sum_{j=0}^{k+1} P_{k+1}(j) L_j(n)
\]

\[
= \sum_{j=0}^{k+1} P_{k+1}(j) \prod_{i \neq j}^{n} \frac{n - j}{i - j}
\]

\[
= \sum_{j=0}^{k+1} \sum_{s=1}^{j} s^k \omega(n) \prod_{i \neq j} \frac{1}{i - j}
\]

\[
= \omega(n) \sum_{s=0}^{k+1} s^k \sum_{j=s}^{k+1} \frac{\lambda_j}{n - j},
\]

where \( \omega(n) = \prod_{i=0}^{k+1} n - i \) and \( \lambda_j = \prod_{i \neq j} \frac{1}{i - j} \). Evaluating this double sum for \( k = 1, 2, \ldots, 10 \)
gives the first 10 polynomials.

An alternative approach that avoids using Lagrange interpolation, but gives a nice recurrence...
relation for the polynomials:

\[ P_{k+1}(n) + (n + 1)^k = P_{k+1}(n + 1) \]

\[ = \sum_{i=1}^{n+1} i^k \]

\[ = \sum_{j=0}^{n} (j + 1)^k \]

\[ = \sum_{j=0}^{n} \sum_{i=0}^{k} \binom{k}{i} j^i \]

\[ = \sum_{i=0}^{k} \binom{k}{i} \sum_{j=0}^{n} j^i \]

\[ = \sum_{i=0}^{k} \binom{k}{i} P_{i+1}(n) \]

\[ = P_{k+1}(n) + kP_k(n) + \sum_{i=0}^{k-2} \binom{k}{i} P_{i+1}(n). \]

Rearranging gives the recurrence formula

\[ P_k(n) = \frac{1}{k} \left[ (n + 1)^k - \sum_{i=0}^{k-2} \binom{k}{i} P_{i+1}(n) \right] \]
6. (a) For the derivation of the recurrence see the paper “Generation of Finite Difference Formulas on Arbitrarily Spaced Grids” by Bengt Fornberg (1988). A PDF of the paper is posted on the class website, with Lecture 13. The paper’s derivation presents the case \( x_0 = 0 \), but for the case \( x_0 = a \) we can define \( g(x) = f(x + a) \) and estimate \( g^{(m)}(0) \) as before, with everything remaining the same except for the substitution \( \alpha_k = x_k - a \).

(b) Pseudocode for the algorithm is also given in the Fornberg paper. A Matlab implementation is given below, where the indices are changed to account for the fact that Matlab indexing begins at 1:

```matlab
function d = fornberg(M,x0,alphas)
N = length(alphas) - 1;
deltas(1,1,1) = 1; %Delta(M,N,K) for the order of indexing.
c1 = 1;
for n = 1:N
    c2 = 1;
    for v = 0:(n-1)
        c3 = alphas(n+1)-alphas(v+1);
        c2 = c2*c3;
        if n <= M, deltas(n+1,n,v+1) = 0; end
        for m = 0:min(n,M)
            if m == 0, D = 0; else, D = m*deltas(m,n,v+1); end
            D2 = (alphas(n+1)-x0)*deltas(m+1,n,v+1);
            deltas(m+1,n+1,v+1) = (D2-D)/c3;
        end
    end
    c1 = c2;
end
d = deltas(end,end,:);
d = reshape(d,1,[]);
end
```

This code returns the vector of entries \( \delta_{m,n,k} \), \( 0 \leq k \leq n \) such that \( f^{(m)}(a) \approx \sum_{k=0}^{n} \delta_{m,n,k} f(x_k) \).

(c) Below is code that (up to the b-values listed in problem (4)) prints the first five rows of the table.

```matlab
rows = 5;
bs = [VECTOR OF B-VALUES];
T = ones(rows+1,2*rows+2); T(:,1) = 1/2;
for i = 1:rows
    n = 2*i + 1;
    for j = 1:i
        m = 2*j-1;
        d = fornberg(m,1:n);
        xs = zeros(1,2*rows+2);
        b = bs(j);
        xs(1:n) = d;
        T(i+1,:) = T(i+1,:) - b*xs;
    end
end
```

It’s a bit much for Matlab to try to find the exact rational numbers (and too tedious to bother
doing it by hand), but the first few rows of the table are printed below:

<table>
<thead>
<tr>
<th>$n$</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>4</td>
<td>0.3750</td>
<td>1.1667</td>
<td>0.9583</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>6</td>
<td>0.3299</td>
<td>1.3208</td>
<td>0.7667</td>
<td>1.1014</td>
<td>0.9812</td>
<td>1.0000</td>
</tr>
<tr>
<td>8</td>
<td>0.3042</td>
<td>1.4604</td>
<td>0.4535</td>
<td>1.4714</td>
<td>0.7394</td>
<td>1.0825</td>
</tr>
<tr>
<td>10</td>
<td>0.2870</td>
<td>1.5890</td>
<td>0.0360</td>
<td>2.2409</td>
<td>−0.1406</td>
<td>1.7209</td>
</tr>
<tr>
<td>12</td>
<td>0.2743</td>
<td>1.7093</td>
<td>−0.4749</td>
<td>3.5218</td>
<td>−2.2396</td>
<td>4.0688</td>
</tr>
</tbody>
</table>

Looking at the later rows of the table, the largest of the coefficients appear to be growing without bound.