1. (5 marks)
Prove that, for the midpoint integration rule for calculating
\[ I = \int_{a}^{b} f(x) \, dx, \]
the truncation error \( T \) can be expressed as
\[ T = f''(\xi) \frac{(b-a)^3}{24}, \]
with \( \xi \in (a, b) \). (Hint: use a Taylor series expansion about
\( c = (a + b)/2 \), and the Weighted Mean Value Theorem for Integrals).

2. (10 marks)
The two-point Gaussian integration rule for calculating
\[ I_1 = \int_{-1}^{1} f(x) \, dx \]
is given by \( \hat{I}_1 = f(-1/\sqrt{3}) + f(1/\sqrt{3}) \). Its degree of precision \( m \) equals 3.

(a) Find the Gaussian integration rule of precision \( m = 3 \) for approximating
\[ I_a = \int_{a}^{b} f(x) \, dx. \]
Express the result in the form \( \hat{I}_a = w_0 f(x_0) + w_1 f(x_1) \) (find \( w_0, w_1, x_0, x_1 \)). (Hint: you
can use the rule for \( I_1 \), after transformation of the integration variable.)

(b) Approximate \( I = \int_{-4}^{4} (x^3 + x^2 + 1) \, dx \) using the Gaussian integration rule derived in (a)
(show your work). Compare with the exact result.

(c) Approximate \( I = \int_{2}^{2} \exp(-x^2) \, dx \) using the Gaussian integration rule derived in (a)
(show your work). An approximation for \( I \) that is correct up to six significant digits
is given by \( \hat{I} = 1.76416 \). (The methods of Question 3 below will give more accurate approximations than the two-point Gaussian rule!)

3. (10 marks)
Download the script \texttt{myIntegral.m} from the course homepage. The purpose of the script is
to approximate \( I = \int_{-2}^{2} \exp(-x^2) \, dx \) using the composite Trapezoid and Simpson integration rules. Unfortunately, the functions \texttt{Trapezoid} and \texttt{Simpson} are missing! You are asked to
write Matlab implementations for the composite Trapezoid and Simpson rules that are to be
used with the script you downloaded. The first line of your Matlab m-files \texttt{Trapezoid.m} and
\texttt{Simpson.m} should read
\begin{verbatim}
function result=Trapezoid(a,b,n,F)
end
\end{verbatim}
and
\begin{verbatim}
function result=Simpson(a,b,n,F).
end
\end{verbatim}
respectively, with \([a,b]\) the integration interval, \( n \) the number of subintervals, and \( F \) the
function to be integrated. (Remember, the Trapezoid rule uses two points in each of the
\( n \) subintervals (namely, the border points of the intervals), and the Simpson rule uses three
points in each of the \( n \) subintervals (with one additional midpoint in each interval)). (Note: in
the script, \( F \) is defined as an ‘anonymous function’. See Matlab’s help for more information.)
You can test your functions **Trapezoid** and **Simpson** by executing the script and verifying whether you obtain an accurate approximation of $I$. See the Note at the end of this assignment about how to hand in your code.

4. (5 marks)
Write a script to approximate $I = \frac{2}{\sqrt{\pi}} \int_0^1 \exp(-x^2) \, dx$ using the composite Trapezoid and Simpson routines of Question 3, for $n = 1, 2, 4, 8, 16$. A highly accurate approximation is given by $\hat{I} = 0.84270079294971$. For both methods, make a table of the truncation errors $T_n$ for varying $n$, and the ratios $r_n = T_{n/2}/T_n$ between errors for successive $n$s. Do you observe the theoretically expected ratios $r_n$ and orders of accuracy for the two methods?

![Figure 1: Automatic integration.](image)

5. (10 marks)
Write a Matlab function that implements automatic integration using the composite Simpson rule. The first line of your Matlab m-file `Automatic.m` should read

```matlab
function [result,errorEstimate,levelsUsed] = Automatic(a,b,nInitial,F,tolerance,maxLevels).
```

Here $[a, b]$ is the integration interval, $nInitial$ is the number of intervals on the initial, coarsest level, $F$ is the function to be integrated, $tolerance$ is a specified absolute error tolerance, and $maxLevels$ is the maximum number of levels that you allow. $nInitial$ needs to be a multiple of 2, because you need to estimate the error of the first approximation on $nInitial$ intervals by calculating an approximation on $nInitial/2$ intervals. A maximum number of levels $maxLevels$ is specified for cases in which the tolerance required cannot be met for reasonable $n$. `Automatic.m` needs to invoke `Simpson.m` from Question 3 at appropriate places. After successful completion, `result` is the resulting approximation of the integral, `errorEstimate`
is the error on the finest level used as estimated by the automatic refinement procedure, and levelsUsed is the number of levels that were used to assure that the estimated error is smaller than the specified error tolerance.

Write a script that calls function Automatic to approximate the integral

\[ I = \int_0^5 (\exp(-x^2) \sin(12x) + 1) \, dx \]

(see Figure 1.) A highly accurate approximation of \( I \) is given by \( \hat I = 5.08454268897537 \). Use \( nInitial = 2 \), \( tolerance = 10^{-3} \), and \( maxLevels = 10 \). How many levels do you need to reach the tolerance? (What is \( n \) on the finest level?) What is the resulting approximation and the estimated error? How does the estimated error compare to the real error? Repeat for \( tolerance = 10^{-6} \). See the Note at the end of this assignment about how to hand in your code. (Tip: In case you do not get Automatic.m to work properly, you can use Simpson.m from Question 3 to approximate \( I \) on successively refined grids by hand or using a script. You can also calculate the error estimates that way.)

6. (optional, if you want to try some recursive programming; no extra credit)
Write a recursive Matlab function that implements adaptive integration. Suggestion for the first line of your Matlab m-file Adaptive.m:

function [result,errorEstimate,finestLevel] = Adaptive(xLeft,xRight,n,F,tol,level).

Now xLeft, xRight, n, tol and level will be different for each recursive invocation of the function on different levels. On the top, coarsest level, xLeft = a, xRight = b, n = nInitial, tol = tolerance, and level = maxLevels. After successful completion, result is the resulting
approximation of the integral, \textit{errorEstimate} is the error estimated by the adaptive refinement procedure (which is the sum of the errors in the recursive subintervals), and \textit{finestLevel} is the label of the finest level that is used in the calculation (the coarsest, initial level has label \textit{maxLevels}, and the finest level allowed has label 1). Plot the function evaluation points used as in Figure 2, indicating where the adaptive method needs more points. Test the adaptive method for the same problem as in Question 5, and compare the results and efficiency of the adaptive and automatic methods. Consider possible optimizations (for example, do not recalculate $\hat{I}(2h)$ on level $h$, but pass it in as an argument of the function).

Note: For Questions 3 and 5, place your versions of Trapezoid.m, Simpson.m and Automatic.m in a single directory (nothing else in this directory ) with the name your\_email\_your\_student\_id. Then zip up this directory (using zip -r your\_email\_your\_student\_id your\_email\_your\_student\_id if on a unix machine). Mail the file your\_email\_your\_student\_id.zip to j39lee@math.uwaterloo.ca by 1:30 pm on the due date. Your code will be subject to black box testing.