Problem sheet 4

Please submit the question marked * to your tutor.

E4.1

Write a program to implement both the trapezoidal rule and Simpson's rule for an arbitrary function f on an interval $[a, b]$.

In [1]:
```
def trapezium(a,b,f):
    # fill in
    # I = ...
    return I

def simpson(a,b,f):
    # fill in
    # I = ...
    return I
```

Test your program on the function $f(x) = x$ and make sure that both rules the trapezoidal rule and Simpson's rule produce the exact value for $\int_a^b x\, dx$ for various a and b. Then test your program for $f(x) = x^3$. What do you observe?

In [ ]:
```
f = lambda x:x
print(trapezium(0,1,f),simpson(0,1,f))

f = lambda x:x**3
print(trapezium(0,1,f), simpson(0,1,f))
```

E4.2 *

Write a program to implement the composite trapezoidal rule on a uniform mesh for approximating $\int_a^b f(x)\, dx$. The program should allow the user to input a, b, J and f.

Run your program for $f(x) = \exp(x)$, $a = 0,\ b = 1$, and $J = 4, 8, 16, 32, 64$. Find the exact value of the integral and compute the error in your approximations. Determine experimentally the rate of convergence as $J \to \infty$.

Explain your results using the theory from lectures.

Repeat for $f(x) = \sqrt{x}$ (the theory for the rate of convergence is significantly more difficult).

In [31]:
```
import numpy as np
def composite_trapezium(a,b,J,f):
    # fill in
    # I = ...
    return I
```

E4.3.

Write a function to implement the composite Simpson's rule for approximating $\int_a^b f(x)\, dx$ on a uniform mesh. The program should allow the user to pass in a, b, J and f.

Run your program for $f(x) = \exp(x)$, $a = 0$, $b = 1$, for $J = 8, 16, 32, 64, 128$. Find the exact value of the integral and compute the error in your approximations. Determine experimentally the rate of convergence as $J \to \infty$.

Repeat for $f(x) = x^3$ and $f(x) = \sqrt{x}$. Explain your results in each case
using the theory from lectures.

In [1]:
```
def composite_simpson(a,b,J,f):
    # fill in
    # I = ...
    return I
```

E4.4.

Consider the integral $\int_1^4 \exp(\frac{1}{2}x^2)\, dx$. Use any method you like (e.g. you could try quad from scipy.integrate) to find the value of this integral correct to 10-decimal places. Approximate this integral using the composite Simpson's rule on a uniform mesh with J subintervals for $J = 8, 16, 32, 64, 128$. Find the error for each J and estimate the rate of convergence as J increases.

In [3]:
```
from scipy.integrate import quad
f = lambda x: np.exp(x**2/2)
# look up how quad works ...
```

E4.5 *

Let $Q_{1,J}(f)$ be the composite trapezoidal rule applied over $[0, 1]$ on the mesh $y_j = jh$, $j = 0, \ldots, J$, where $h = 1/J$ and $J \in \mathbb{N}$. This is used to approximate $I(f) = \int_0^1 f(x)\, dx$ as described in lectures.

We show in lectures that $I(f) - Q_{1,J}(f) = -\frac{1}{12}h^2 f''(\zeta)$, for some $\zeta \in [0,\ 1]$.

A more precise result, based on the *Euler-Maclaurin* formula, shows that, provided f is sufficiently smooth, there exist $C_2,\ C_4,\ C_6,\ \ldots$, **independent** of h, such that

$$I(f) - Q_{1,J}(f) = C_2 h^2 + C_4 h^4 + C_6 h^6 + \cdots.$$

(a) Write down the corresponding expansion for $I(f) - Q_{1,(2J)}(f)$. \item By eliminating C_2 between the two expansions, find $\theta \in \mathbb{R}$ such that

$$I(f) - (\theta Q_{1,J}(f) + (1-\theta) Q_{1,2J}(f)) = \tilde{C}_4 h^4 + \cdots,$$

for some $\tilde{C}_4 \in \mathbb{R}$, which you need not determine.

(b) Find coefficients $a,\ b,\ c \in \mathbb{R}$ such that

$$\theta Q_{1,J}(f) + (1-\theta) Q_{1,2J}(f) = \sum_{j=1}^{J} h\left(a f(y_{j-1}) + b f((y_{j-1} + y_j)/2) + c f(y_j)\right).$$

What is this method? Could you use this idea to generate more accurate rules?

In [ ]:
```
```