Section 2, Exercise 1

Write a program to calculate the area of the trapezoid bounded by $x_1 = 1.5$, $x_2 = 2$, the x-axis, and a straight line drawn between $f(x_1)$ and $f(x_2)$, where $f(x) = x^2$.

Section 2, Exercise 1

Write a program to calculate the area of the trapezoid bounded by $x_1 = 1.5$, $x_2 = 2$, the x-axis, and a straight line drawn between $f(x_1)$ and $f(x_2)$, where $f(x) = x^2$.

```
def f(x):
    return x**2

x1=1.5
x2=2.0
dx=x2-x1
area = 0.5*(f(x1)+f(x2))*dx
print(area)

```

1.5625

Section 2, Exercise 2

Suppose you have two data points (x_1, y_1) and (x_2, y_2), and you want to estimate what the value of y would be for some value of x between x_1 and x_2 using *linear interpolation*. What does that mean? How does this relate to a numerical integral using the trapezoidal rule?

Section 2, Exercise 2

Suppose you have two data points (x_1, y_1) and (x_2, y_2), and you want to estimate what the value of y would be for some value of x between x_1 and x_2 using *linear interpolation*. What does that mean? How does this relate to a numerical integral using the trapezoidal rule?

An interpolation is an educated guess about the y value between two known points. A linear interpolation assumes that the function is a smooth line between the two known points.

With a linear interpolation, the extracted y value for an x halfway in between the two known points is the average of the y value at the two known points. The trapezoidal rule, by doing an average, is making a linear interpolation between the two give points.

Section 2, Exercise 3

Can you think of other ways of representing the vector field with a plot, other than the arrow plot? You have to depict both the magnitude and direction, which you could do separately or on the same plot.

Section 2, Exercise 3

Can you think of other ways of representing the vector field with a plot, other than the arrow plot? You have to depict both the magnitude and direction, which you could do separately or on the same plot.

A density plot, which represents density by shading or color, could be used to represent the magnitude of the vector.
A similar color-based plot could be used to show the direction, provided that the scale only represents numbers from 0 to 2*pi.

Keep the arrows to represent direction, but make all the arrows the same size. Use the color of the area to represent the magnitude.

Section 2, Exercise 4

Suppose you have a list of numbers, and you want to determine which number is closest to some value. Make up a list of numbers, and write a program that finds the one closest to 5 and prints it out.

Section 2, Exercise 4

Suppose you have a list of numbers, and you want to determine which number is closest to some value. Make up a list of numbers, and write a program that finds the one closest to 5 and prints it out.

```python
#here's my list:
a = [5.001, 5.000002, 4.739, 4.999999999999999, 5.28, 4.5, 6.2]
print(a)

#here we search through the list
diff=10
closest=10
for i in a:
    if abs(i-5)<diff:
        diff=abs(i-5)
        closest=i
print("closest value to 5 in this list is: ",closest)
```

```
>>>
= RESTART: C:\Users\lisaw\OneDrive\Documents\Teaching\4350\2023Fall\Lecture Note
s\Section2-NumericalCalculus\exercises\Exercise4.py
[5.001, 5.000002, 4.739, 4.999999999999999, 5.28, 4.5, 6.2]
closest value to 5 in this list is:  4.999999999999999
>>> |
```

Section 2, Exercise 5

Write a program which uses the forward difference to approximate the derivative of $f(x) = x^5$, evaluated at $x = 0.5$, using $h = 1e - 5$. Also find the true value of the integral and compare the values.

Section 2, Exercise 5

Write a program which uses the forward difference to approximate the derivative of $f(x) = x^5$, evaluated at $x = 0.5$, using $h = 1e-5$. Also find the true value of the integral and compare the values.

```
>>>
= RESTART: C:\Users\lisaw\OneDrive\Documents\Teaching\4350\2023Fall\Lecture Note
s\Section2-NumericalCalculus\exercises\Exercise5.py
Numerical approximation =  0.3125125
True value =  0.3125
Difference =  1.25e-05
Note that the approximation error is 0.5*h*f''(x) =  1.25e-05
The rounding error is much smaller, ~ 6.25e-13
```