Euler Method for approximating a function: When dealing with differential equations, we do not always need to resort to a numerical solution in order to calculate an answer that is close enough to the solution. One such way of doing this is through Taylor expansion where we take the initial value and then for time step (t+dt) calculate the derivative of the function at that point t. With a smaller dt we can approximate the function with greater accuracy.

Pseudocode: In order to determine what is necessary for our calculation we should first comment in a description of each part of our code and what we want it to do. An example might be:

```
#Declare variables and arrays

#Perform these subroutines:
#Initialize variables- set the initial values of our variables
#perform calculation
```

```
Calculate Nu at i = 1, skip the initial value, calculate using euler method, t_steps=t_i+dt, repeat for n-1 steps
```

```
#store the results in a variable
```

How to test your program:

Determine if the output looks reasonable. Does the result match what you had expected? If you show this to someone you should be able to explain the reasoning behind why the output looks the way it does.

Does the program agree with exact results that are available? If there are known results with certain parameters, check with your program to see if you obtain the same result with that same parameter.

Check that your program works for different step sizes. Since dt is an independent variable in relation to the values in your function it should not become altered in any significant way if it is changed.

Notes about approximations:

You should always test your function to different approximations to understand what happens as the accuracy increases, you should also realize that not one method solves all differential equation problems

The guidelines to programming:

Program structure: use subroutines to break up the task and make the program easy to understand when reading. Use subroutines to perform jobs that take multiple lines of code or that are required repeatedly

Use descriptive names: Variables and subroutines should have names that describe their intended purpose.

Use comments

Do not attempt to write everything compact. Sacrifice time and speed for clarity, break code into more variables, etc.

Make the graphical output clear. Labels, parameter values, what should be plotted and how.