Lily Davoren, Aidan Corpus
Lab 1 Report

TASK 2 PROGRAMMING STRATEGIES:

To accomplish task (a), we defined the Maclaurin series for estimating sin(x) by iterating through the x-values and then iterating through the four approximation orders within that loop for each x-value. Within this loop, we calculated the Maclaurin series estimate of sin(x) at a range of x-values and stored the estimates for each of the four orders for an x-value in a list. For plotting, we defined 100 x-values between 0 and 2pi to be substituted into our Maclaurin function. When we plot the four approximation order estimates for each of these x-values, we plot them all on the same graph. Refer to Appendix A to see the code used to complete this task.

To accomplish task (b), we defined a function that calculates the true relative error of the Maclaurin series. This function calls the Maclaurin function defined in task (a) for 'n' approximation orders, takes the Maclaurin estimate of the x-values given in the table, and finds the true relative error of each estimate. We then placed these error values for each order for each x-value into a dataframe to produce a table of values. Refer to the code in Appendix B.

To accomplish task (c), we wrote a new set of code that calculates the true error for Maclaurin estimates of sin(0.2) up to order n by defining the equation of the Maclaurin series and and then calculating the true value for a certain number of n-values. For plotting these error values on a logarithmic scale, if the error value is greater than zero, then we can keep the error value in our list of errors; however, if the error value is less than zero, to avoid having a case of log(0) in which the values would not appear on the plot because they are undefined, we replace the error value with a very small number close to zero. Finally, we plot the true error against the order at both a linear scale and a logarithmic scale. Refer to code in Appendix C.

Lily Davoren, Aidan Corpus

Appendix A

```
x_list = []

# Define a Maclaurin series function for estimating sin(x)
def maclaurin(x, n):
    x_list = []
    for j in x:
        sum = 0.0
        for i in range(0,n):
            exp = (2*i) + 1
            num = j**exp / math.factorial(exp)
            if i % 2 == 0:
                sum += num
            elif i % 2 != 0:
                sum += -num
        x_list.append(sum)
    return x_list

# Generate x values for plotting
x = np.linspace(0, 2*math.pi, 100)

plt.xlim(0, 2*math.pi)
plt.ylim(-3, 3)

# Calculate Maclaurin series for different orders and the true sin(x) values
n_1 = maclaurin(x, 1)
n_3 = maclaurin(x, 2)
n_5 = maclaurin(x, 3)
n_7 = maclaurin(x, 4)
control = np.sin(x)

# Plot sin(x) and Maclaurin series approximations
plt.plot(x, control, label = 'sin(x)')
plt.plot(x, n_1, label = 'O(1)')
plt.plot(x, n_3, label = 'O(3)')
plt.plot(x, n_5, label = 'O(5)')
plt.plot(x, n_7, label = 'O(7)')

plt.grid()
plt.legend()
plt.show()
```

Appendix B

```
# Define a list of x values
x = [0, 0.01, 0.1, 0.2, math.pi/2]
x_list = []

# Define a Maclaurin series function for estimating sin(x)
def maclaurin(x, n):
    x_list = []
    for j in x:
        sum = 0.0
        for i in range(0,n):
            exp = (2*i) + 1
            num = j**exp / math.factorial(exp)
            if i % 2 == 0:
                sum += num
            elif i % 2 != 0:
                sum += -num
        x_list.append(sum)
    return x_list

# Define a function to calculate the true relative error of the Maclaurin series
def true_rel_error(x, n):
    big_list = []
    list1 = []
    for i in range(1, n):
        list1 = maclaurin(x, i)
        for j in range(0, len(list1)):
            list1[j] = list1[j] - np.sin(x[j]) / np.sin(x[j])
        big_list.append(list1)
    return big_list

# Calculate the true relative errors for different x values and orders of n
true = true_rel_error(x, 5)
true1 = true[0]
true2 = true[1]
true3 = true[2]
true4 = true[3]

names = ['x=0', 'x=0.01', 'x=0.1', 'x=0.2', 'x=pi/2']
```

Lily Davoren, Aidan Corpus
Lab 1 Report

```
# Create a DataFrame to display true relative errors in a table
data = {
    'n': names,
    "1": true1,
    '3': true2,
    '5': true3,
    '7': true4
}

df = pd.DataFrame(data)

df = df.T

pd.options.display.float_format = '{:.5f}'.format

print(df)
```

Appendix C

```
# Define the true value of sin(0.2)
true = np.sin(0.2)

# Create an array of n values from 1 to 20
n_values = np.arange(1, 21)

# Initialize an empty list to store the true errors
errors = []

# Calculate the true error for Maclaurin estimates of sin(0.2) up to order n
for i in n_values:
    sign = (-1) ** i
    odd = 2 * i + 1
    approx = sign * (0.2 ** odd) / np.math.factorial(odd)
    errors.append((approx - true))

# Add a small number to each error to avoid log(0) and for visual clarity
log = [err if err > 0 else 1e-16 for err in errors]

# Plot the true errors
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 5))

ax1.plot(n_values, errors, marker='o', linestyle='-')
ax1.set_xlabel('n')
ax1.set_ylabel('True Error')
ax1.set_title('True Error vs. n (Linear)')
ax1.grid()

ax2.plot(n_values, log, marker='o', linestyle='-')
ax2.set_xlabel('n')
ax2.set_ylabel('True Error')
ax2.set_title('True Error vs. n (Log Scale)')
ax2.set_yscale('log')
ax2.set_ylim(1e-18, 1e-14)
ax2.grid()

plt.tight_layout()
plt.show()
```