Before you turn this problem in, make sure everything runs as expected. First, **restart the kernel** (in the menubar, select Kernel→Restart) and then **run all cells** (in the menubar, select Cell→Run All).

Make sure you fill in any place that says `YOUR CODE HERE` or "YOUR ANSWER HERE", as well as your name and collaborators below:

In [1]:
```
NAME = "Sharjil Mohsin"
COLLABORATORS = "N/A"
```

Minor Assignment 1

ENGINEER 1D04
Dr. Ashgar Bokhari
McMaster University, Fall 2017

Background

You might recall from Calculus that

$$lim_{n\to\infty}\left(1+\frac{1}{n}\right)^n = e,$$

the numerical constant that equals $2.71828182845904523536\ldots$ to 20 decimal places. e is irrational (like $\sqrt{2}$) and transcendental (like π) and has many truly wonderful properties.

Overview

The purpose of this assignment is to write a Python program that computes the steps of the convergence of $(1 + 1/n)^n$ to e as n increases in size. Design, implement, and test a program that satisfies the requirements below.

Requirements

1. The program contains a function that does the following:

   a. Finds the nth term of the convergence, where n is an integer chosen by the user. i.e if the user chooses 2, the output of the function should be 2.25, as this is what the convergence evaluates to when n = 2 (shown here $(1 + 1/2)^2$). The first few terms of the convergence are displayed here so you can check your answers:

   $$(1 + 1/1)^1, (1 + 1/2,)^2, (1 + 1/3)^3, \dots, (1 + 1/n)^n$$

   b. Calculates the following sigma notation.

   $$\sum_{i=0}^{n} 5i^2$$

   HINT: A `for` loop would come in handy here.

   c. Calculates the following product notation.

   $$\prod_{i=1}^{n} 3i^2$$

   HINT: This will look very similar to part B.

2. Your name, MacID, student number, and the date are given in comments at the top of your code before the rest of your program.

3. Your program MUST have valid Python syntax and it must run without errors. Ensure that your program runs properly by running it before you submit.

4. You must sign out with a TA or IAI after you have submitted your lab at the submission station. Failure to do so could result in a mark of zero.

Design and Implementation Instructions

- HINT: Use the design of this summation notation if you get stuck, it is similar to how your final code should look:

$$\sum_{i=0}^{10} 2i$$

```
total = 0
for i in range(11):
   total = total + (2 * i)
return total
```

1. When coding $(1 + 1/n)^n$, remember to use brackets for proper grouping (BEDMAS)

In [2]:
```
# DO NOT PUT AN INPUT STATEMENT IN THIS CELL

###################---minor1(n) - (e: 4, sigma: 3, product: 3 Marks)---###################
def minor1(n):
    #Place your code between here...

    e = (1+(1/n))**n        #change this value to your answer for e_convergence

    sigma = 0
    for i in range(n+1):
        sigma = sigma + 5*(i**2)
    #change this value to your answer for sigma notation

    product = 1
    for i in range(1, n+1):
        product = product * 3*(i**2)
    #change this value to your answer for product notation

    #...and here
    # YOUR CODE HERE
    return e, sigma, product
# Change the value 5 to another number to test your code
result = minor1(5)
print("e_convergence =", result[0])
print("sigma =", result[1])
print("product =", result[2])
```

```
e_convergence = 2.4883199999999994
sigma = 275
product = 3499200
```

In [3]:
```
###########---TESTING CELL---#############

#Run this cell to see your autograded mark

'''
Sample Input:
5

Sample Output:
e_convergence = 2.4883199999999994
sigma = 275
product = 3499200
'''
from autograder import autograde
checkMinor1Mark = autograde()
checkMinor1Mark.marker([minor1], "minor1")
```

```
Automarked Score = 10.0 / 10.0
(This mark is an estimate of your grade and may not accurately represent the mark you receive)
Use your testing plan to ensure your code is working as intended

BEFORE LEAVING THE LAB:
1. Restart your kernel and run all cells to check that your code is working as intended
        Kernel --> Restart & Run All
2. Make sure to answer the design questions and testing plan
3. Save your work
        File --> Save and Checkpoint
4. Submit your assignment
        On Jhub: Assignments --> Downloaded Assignments --> assign#_section# --> Submit
5. Sign out with your IAI or one of your TAs
```

Design Questions

1. What does your program do if the user chooses an integer less than 1? Ideally, what could your program do in this case?
2. Do you think your program provides a good way to compute an approximation to e?
3. Is it ok for the sigma notation to start at 1 instead of 0? Why or why not?
4. Is it ok for the product notation to start at 0 instead of 1? Why or why not?

Enter your answers to the above questions in to the markdown cell below.

1. If the user chooses an integer less than 1, the program will fail to run and instead show an error to the user saying that the program cannot interpret the number that the user input as it cannot interpret a float object as an integer. Any number less than 1 or any number with decimals are float objects, and since the program can only take in integer values, it will show an error whenever a decimal or a number less than 1 is set as 'n'.
2. Yes, I believe that the program is a great way to compute an approximate the value of e because as we can see from the background section of this lab, we get closer to the value of e as we approach infinity. Since we cannot use infinity as it will crash the program, putting in a really high number into the program will give us a number that is really close to the actual number of e (for example, setting n = 1000 will give the output 2.7169239322355936, which is pretty close to the value of e). Therefore, this program is effective for approximating the value of e.
3. Yes, it is okay for the sigma notation to start at 1 instead of 0. This is because when we use the sigma notation for this program, we see that when n = 0, the value of the function is 0. Since there is nothing to be added, there is no difference to the final result of the sigma notation it starts at 1 instead of 0 in the case of this program.
4. No, it is not okay to start at 0 instead of 1 in our case. This is because if the product notation starts at 0, the final result from the program will always show 0 as the 0 will multiply any number that is in the notation, which will result in the answer being 0 every time.

Testing Plan

This is the first lab where you will practice implementing a test plan for your code. We will be doing so in all future labs, as good software development involves making sure your program does what you want.

Produce a test plan in the Markdown cell below, in the following form:

```
Test i
Input: [n]
Expected Output: [A, B, C]
Actual Output: [Aa, Ba, Ca]
Result: Pass/Fail
```

where i is the test number (i.e Test 1, Test 2, etc) and n is the input for the program. A, B, C are the outputs the program is expected to produce, for the values of e, sigma and product respectively. Find these values with your calculator. Aa, Ba, Ca are the outputs the program actually produces for the values of e, sigma and product respectively.

Example:

```
Test 1
Input: [2]
Expected Output: [2.4883199999999994, 275, 3499200]
Actual Output: [2.4883199999999994, 275, 3499200]
Result: Pass (Expected and actual output match! A Fail would be if the expected output differs from the actual output.)
```

Please include at least three tests in the Markdown cell below. Use your test cases to build your confidence in the correctness of your code before your final submission. It can also help find bugs you did not know about before!

```
Test 1
Input: [5]
Expected Output: [2.4883199999999994, 275, 3499200]
Actual Output: [2.4883199999999994, 275, 3499200]
Result: Pass

Test 2
Input: [1]
Expected Output: [2, 5, 3]
Actual Output: [2.0, 5, 3]
Result: Pass

Test 3
Input: [2]
Expected Output: [2.25, 25, 36]
Actual Output: [2.25, 25, 36]
Result: Pass
```

Preparation for Major 1

This lab has provided an opportunity to practice translating from mathematical notation to Python syntax. This is an important skill for an Engineer using Python. This skill will be tested in Major 1.

To help you practice this skill further please implement the following mathematical expressions in Python. The expected value is given so that you can check your answers. Hint: For this exercise, you need to include `import math` at the beginning of your script.

The expressions in Python syntax and the results obtained should be added to the cell below. This part of the lab is optional, but recommended to prepare for your first Major lab.

Expression	Expected Output
$0.5sin(\frac{\pi}{2}) + 1.6cos(\frac{3}{4}\pi)$	-0.631370849898
$e^{-4.5^2} + ln(0.87)$	-0.139262065728
$1.453x10^{-12}\sqrt{2.5x10^6}$	$2.29739472011e^{-9}$
$tan^{-1}(0.8)$	0.674740942224
$\frac{4}{3}\pi(0.74)^3$	1.69739832194

In [ ]:

In [4]:
```
#Ignore this cell
```

In [5]:
```
#Ignore this cell
```

In [6]:
```
#Ignore this cell
```

In [7]:
```
#Ignore this cell
```

In [8]:
```
#Ignore this cell
```

In [9]:
```
#Ignore this cell
```

In [10]:
```
#Ignore this cell
```

In [11]:
```
#Ignore this cell
```

In [12]:
```
#Ignore this cell
```

In [13]:
```
#Ignore this cell
```