MATH 152 Lab 8

Put team members' names and section number here.

```
import sympy as sp
from sympy.plotting import (plot,plot_parametric)
import matplotlib.pyplot as plt
import numpy as np
```

Question 1

1a

```
# Enter your code here
n = sp.symbols('n', positive=True, integer=True)

a_n = sp.factorial(2*n) / (2**n * sp.factorial(n)**2)
ratio = sp.simplify(sp.Abs(a_n.subs(n, n+1) / a_n))
limit = sp.limit(ratio, n, sp.oo)

print("a_n =", a_n)
print("Ratio a_{n+1}/a_n =", ratio)
print("L =", limit)
```

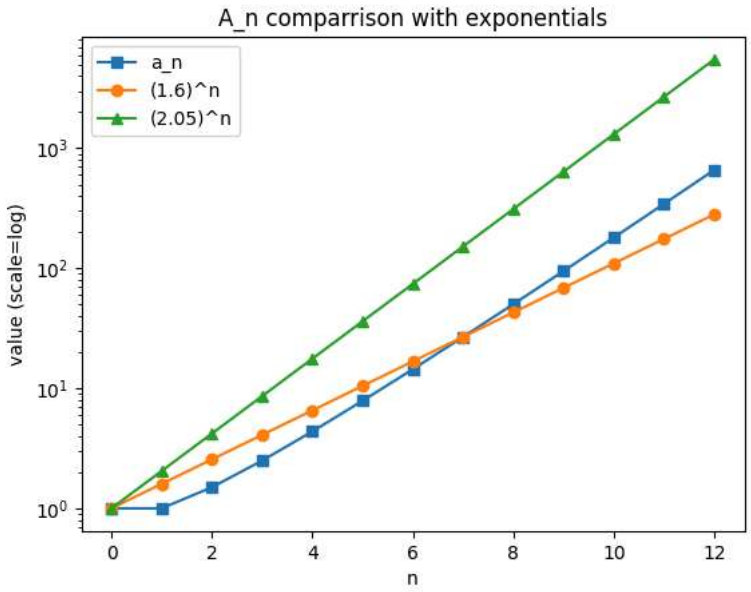
```
a_n = factorial(2*n)/(2**n*factorial(n)**2)
Ratio a_{n+1}/a_n = (2*n + 1)/(n + 1)
L = 2
```

1b

```
# Enter your code here
n_array = np.arange(0, 13, 1)

a_values = [sp.factorial(2*k) / (2**k * sp.factorial(k)**2) for k in n_array]
b_values = [1.6**k for k in n_array]
c_values = [2.05**k for k in n_array]

plt.figure()
plt.semilogy(n_array, a_values, 's-', label='a_n')
plt.semilogy(n_array, b_values, 'o-', label='(1.6)^n')
plt.semilogy(n_array, c_values, '^-', label='(2.05)^n')
plt.xlabel('n')
plt.ylabel('value (scale=log)')
plt.title('A_n comparrison with exponentials')
plt.legend()
plt.show()
```



1c

```
# Enter your code here
M = sp.symbols('M', positive=True)
t_n = sp.factorial(2*n) / (M**n * sp.factorial(n)**2)
ratio_M = sp.simplify(t_n.subs(n, n+1) / t_n)
Limit_M = sp.limit(ratio_M, n, sp.oo)
M_inconclusive = sp.solve(sp.Eq(Limit_M, 1), M)[0]

print("Ratio t_{n+1}/t_n =", ratio_M)
print("Limit L(M) =", Limit_M)
print("Ratio test inconclusive at M =", M_inconclusive)
```

```
Ratio t_{n+1}/t_n = 2*(2*n + 1)/(M*(n + 1))
Limit L(M) = 4/M
Ratio test inconclusive at M = 4
```

1d

```
# Enter your code here
M_value = 4
term1 = sp.factorial(2*n) / (M_value**n * sp.factorial(n)**2)
term2 = term1 / n

asymp1 = sp.limit(sp.sqrt(n) * term1, n, sp.oo)
asymp2 = sp.limit(n**(sp.Rational(3,2)) * term2, n, sp.oo)

print("S1 behaves like 1/sqrt(n):", asymp1, "=> diverges")
print("S2 behaves like 1/n^(3/2):", asymp2, "=> converges")
```

```
S1 behaves like 1/sqrt(n): 1/sqrt(pi) => diverges
S2 behaves like 1/n^(3/2): 1/sqrt(pi) => converges
```

Question 2

2a

```
# Enter your code here
a_n_ram = (2*sp.sqrt(2)) * sp.factorial(4*n) * (1103 + 26390*n) / (9801 * sp.factorial(n)**4 * 396**(4*n))
ratio_ram = sp.simplify(sp.Abs(a_n_ram.subs(n, n+1) / a_n_ram))
L_ram = sp.limit(ratio_ram, n, sp.oo)

print("Ratio =", ratio_ram)
```

```
print("Limit as n→∞ =", L_ram)
```

```
Ratio = (2*n + 1)*(4*n + 1)*(4*n + 3)*(26390*n + 27493)/(3073907232*(n + 1)**3*(26390*n + 1103))
Limit as n→∞ = 1/96059601
```

2b

```
# Enter your code here
print("Numeric value of 1/396^4 =", float(1 / (396**4)))
print("Limit < 1, series converges absolutely by Ratio Test.")
```

```
Numeric value of 1/396^4 = 4.066485764395378e-11
Limit < 1, series converges absolutely by Ratio Test.
```

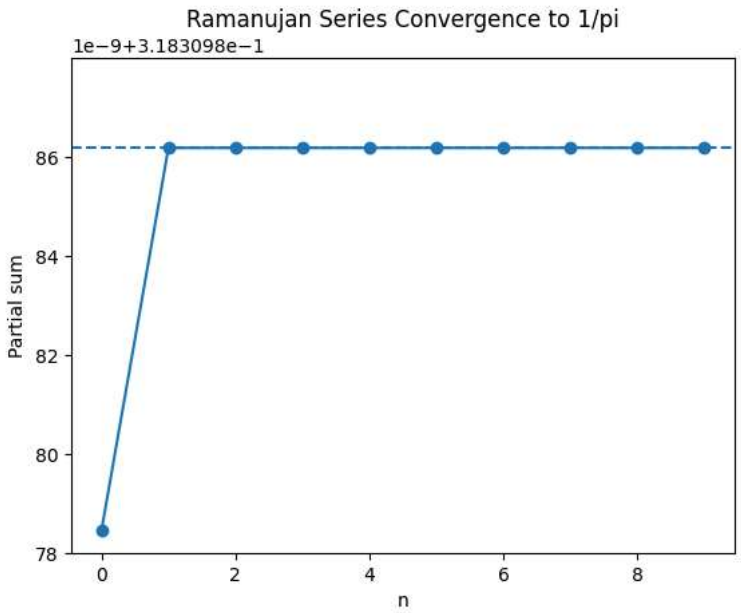
2c

```
# Enter your code here
def a_function(k):
    return (2*sp.sqrt(2)) * sp.factorial(4*k) * (1103 + 26390*k) / (9801 * (sp.factorial(k)**4) * 396**(4*k))

partial_sums = []
s = 0
for k in range(10):
    s += a_function(k)
    partial_sums.append(float(s))

plt.figure()
plt.plot(range(10), partial_sums, marker='o')
plt.xlabel('n')
plt.ylabel('Partial sum')
plt.title('Ramanujan Series Convergence to 1/pi')
plt.ylim([0.318309878, 0.318309888])
plt.axhline(1/np.pi, linestyle='--')
plt.show()

print("1/pi =", 1/np.pi)
print("Last partial sum ≈", partial_sums[-1])
```




```
1/pi = 0.3183098861837907
Last partial sum ≈ 0.3183098861837907
```

Question 3

3a

```
# Enter your code here
x = sp.symbols('x', real=True)
lhs = 1/(1 - x)
rhs = -1/x * (1/(1 - 1/x))
print("Simplified difference =", sp.simplify(lhs - rhs))
```

```
Simplified difference = 0
```

3b

There is no submission for this part.

3c

```
# Enter your code here
k = sp.symbols('k', positive=True, integer=True)
series_at_5 = sp.summation(-5**(-k), (k, 1, sp.oo))
lhs_val = 1/(1 - 5)
print("Series sum at x=5:", series_at_5)
print("Check vs 1/(1-5):", lhs_val)
```

```
Series sum at x=5: -1/4
Check vs 1/(1-5): -0.25
```