```python
from sympy import *
from sympy import log as ln
from sympy import symbols
from sympy import diff
from sympy import Function
from sympy import plot

t = Symbol('t')
f = 1 / ln(t)
f_2nd_derv = diff(f, t, 2)
f_4th_derv = diff(f, t, 4)

print(f"Function, f(t): {f}\nSecond Derv, f''(t): {f_2nd_derv}\nFourth Derv, f''''(t): {f_4th_derv}")

f_2 = lambdify(t, f_2nd_derv)
f_4 = lambdify(t, f_4th_derv)

print(f"Second Derv, f''(2): {f_2(2)}\nFourth Derv, f''''(2): {f_4(2)}")

Function('f_2nd_derv')
Function('f_4th_derv')
plt = plot(f, f_2nd_derv, f_4th_derv, (t, 2, 500), title="Derv. and Monotonicity", legend= True, xlabel='t', ylabel='f(t) & derv. of f(t)')
plt.show

```

Q2_Ans:

2(a):

We know that in Riemann sum technique, we select the numbers ξ_i that minimize $f(x)$ on every sub-interval $[x_{i-1}, x_i]$

$$\sum_{i=1}^{n} \min_{[x_{i-1},x_i]} f(\xi_i)\,(x_i - x_{i-1}) \leq \int_a^b f(x)dx \leq \sum_{i=1}^{n} \max_{[x_{i-1},x_i]} f(\xi_i)\,(x_i - x_{i-1})$$

Also, we define Reimann Left (L) and Right (R) as:

$$L(f, x_i) = \sum_{i=1}^{n} f(x_{i-1})\,(x_i - x_{i-1})$$

$$R(f, x_i) = \sum_{i=1}^{n} f(x_i)\,(x_i - x_{i-1})$$

Hence, for a monotonically decreasing function, we minimalize the $R(f, x_i)$ and take the maximum of $L(f, x_i)$ to minimize the $f(x)$ on every sub-interval $[x_{i-1}, x_i]$.

Therefore,

$$\min\{L(f, x_i), R(f, x_i)\} \leq \int_a^b f(x)dx \leq \max\{L(f, x_i), R(f, x_i)\}$$

Reduces to

$$R(f, x_i) \leq \int_a^b f(x)dx \leq L(f, x_i)$$

2(b):

$$L(f, x_i) = \sum_{i=1}^{n} f(x_{i-1})\,(x_i - x_{i-1})$$

$$R(f, x_i) = \sum_{i=1}^{n} f(x_i)\,(x_i - x_{i-1})$$

So,

$$L - R = \sum_{i=1}^{n} [f(x_{i-1})\,(x_i - x_{i-1}) - f(x_i)\,(x_i - x_{i-1})]$$

$$= \sum_{i=1}^{n} (x_i - x_{i-1})[f(x_{i-1}) - f(x_i)]$$

We know,

$$h = \max_{1<i\leq n} (x_i - x_{i-1}) = \frac{b-a}{n}$$

and

$$x_i = b \;\;\&\;\; x_{i-1} = a$$

Therefore,

$$L - R = \frac{b-a}{n}[f(a) - f(b)]$$

```python
import numpy as np
from sympy import *
from sympy import log as ln
import sympy



def Reimann_Int(a, b, N):
    h = (b - a) / (N - 1)
    x = np.linspace(a, b, N)
    f = 1 / np.log(x)

    I_riemannL = h * sum(f[:N-1])

    #I_riemannR = h * sum(f[1::])

    return print(f"Reimann left Int:{I_riemannL}")

t = Symbol('t')
f = 1 / ln(t)

from numpy import integrate

Li_200_ = integrate(f, (t, 2, 200))

print(f"Li(200) using standard function:{Li_200_}")

for i in range (0, 2000):
    i += 1
    if (Li_200_ - Reimann_Int(2, 200, i)) == 1e4:
        print(f"No. of steps(value of N) needed to produce 3 decimal place accuracy:{i}")
    else:
        continue


```

```python
import numpy as np

def Reimann_Int_mid(a, b, N):
    h = (b - a) / (N - 1)
    x = np.linspace(a, b, N)
    f = 1 / np.log(x)

    I_mid = h * sum(1 / np.log((x[:N-1] \
        + x[1:])/2))

    return print(f"Reimann left Int:{I_mid}")
```

```python
import numpy as np


def simp_intg(a, b, N):
    h = (b - a) / (N - 1)
    x = np.linspace(a, b, N)
    f = np.exp^(-x)
    I_simp = (h/3) * (f[0] + 2*sum(f[:N-2:2]) \
            + 4*sum(f[1:N - 1:2]) + f[N-1])

    return print(f"Simpson integration:{I_simp}")
```

```python
from sympy import *
from sympy import exp
from sympy import symbols
from sympy import diff
from sympy import Function
from sympy import plot

t = Symbol('t')
f = 1 / exp(-1 * t**2)
f_4th_derv = diff(f, t, 4)

print(f"Fourth Derv, f''''(t): {f_4th_derv}")

Function('f_4th_derv')
plt = plot(f_4th_derv, (t, 0, 5), title="Fourth derv. Graph", legend= True, xlabel='t', ylabel='Fourth derv. of f(t)')
plt.show
```

Q8_Ans:

$$w_1 f(x_1) + \ w_2 f(x_2) = \int_{-1}^{1} f(x)dx$$

To get the nodes and weight according to the conditions imposed:

$$w_1 + w_2 = \int_{-1}^{1} f(1)dx = 2$$

$$w_1 x_1 + w_2 x_2 = \int_{-1}^{1} x dx = 0$$

$$w_1 {x_1}^2 + w_2 {x_2}^2 = \int_{-1}^{1} x^2 dx = \frac{2}{3}$$

$$w_1 {x_1}^3 + w_2 {x_2}^3 = \int_{-1}^{1} x^3 dx = 0$$

Therefore, solving the above equation we can easily get the following values:

$$w_1 = w_2 = 1 \ \& \ x_1 = \frac{-1}{\sqrt{3}},\ x_2 = \frac{1}{\sqrt{3}}$$

We can say that this yield:

$$\int_{-1}^{1} f(x)dx = f\left(\frac{-1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right)$$

Therefore we can infer that it has degree of precision equal to 3 since it integrates exactly all polynomials of degree ≤ 3.