In [3]:

```python
import sympy as sp

x = sp.Symbol('x')

C = 5*x**3 - 10*x**2 + 4*x + 3

C_derivative = sp.diff(C, x)
print("Gradient (First Derivative):", C_derivative)

critical_points = sp.solve(C_derivative, x)
print("Critical Points:", critical_points)

C_second_derivative = sp.diff(C_derivative, x)
print("Second Derivative:", C_second_derivative)

for point in critical_points:
    second_derivative_value = C_second_derivative.subs(x, point)
    if second_derivative_value > 0:
        print(f"x = {point} is a Minimum (Second derivative is positive)")
    elif second_derivative_value < 0:
        print(f"x = {point} is a Maximum (Second derivative is negative)")
    else:
        print(f"x = {point} is an Inflection Point (Second derivative is zero)")


print("\nDecision-Making Interpretation:")
if critical_points:
    min_x = min(critical_points, key=lambda p: C_second_derivative.subs(x, p) if C_second_derivative.subs(x, p)
    print(f"The optimal number of AI startups to fund for minimizing cost is approximately x = {min_x}.")
else:
    print("No minimum cost point found in the given function.")
```

```
Gradient (First Derivative): 15*x**2 - 20*x + 4
Critical Points: [2/3 - 2*sqrt(10)/15, 2*sqrt(10)/15 + 2/3]
Second Derivative: 30*x - 20
x = 2/3 - 2*sqrt(10)/15 is a Maximum (Second derivative is negative)
x = 2*sqrt(10)/15 + 2/3 is a Minimum (Second derivative is positive)

Decision-Making Interpretation:
The optimal number of AI startups to fund for minimizing cost is approximately x = 2*sqrt(10)/15 + 2/3.
```