```cpp
// COS30008, Problem Set 1/2, 2024

#include "Polynomial.h"

#include <cmath>

double Polynomial::operator()(double aX) const
{
    double result = 0.0;

    for (size_t i = 0; i <= fDegree; i++)
    {
        // raise x to the power of i, then multiply with coefficient at that degree
        result += pow(aX, i) * fCoeffs[i];
    }

    return result;
}

Polynomial Polynomial::getDerivative() const
{
    Polynomial derivative;

    if (fDegree > 0)
    {
        derivative.fDegree = fDegree - 1;

        for (size_t i = 0; i < fDegree; i++)
        {
            // (d/dx) a_i * x^i = i * a_i * x^(i-1)
            derivative.fCoeffs[i] = (i + 1) * fCoeffs[i + 1];
        }
    }

    return derivative;
}

Polynomial Polynomial::getIndefiniteIntegral() const
{
    Polynomial antiDer;
    antiDer.fDegree = fDegree + 1;

    for (size_t i = 0; i <= fDegree; i++)
    {
        // ∫ a_i * x^k (dx) = a_i / (k+1) * x^(k+1)
        antiDer.fCoeffs[i + 1] = fCoeffs[i] / (i + 1);
    }

```

```
49      return antiDer;
50  }
51
52  double Polynomial::getDefiniteIntegral(double aXLow, double aXHigh) const
53  {
54      Polynomial antiDer = getIndefiniteIntegral();
55      return antiDer(aXHigh) - antiDer(aXLow);
56  }
```