Suppose we want to find a real root of the equation \( f(i) = 0 \) and there appears to be no apparent solution. Newton’s method is an algorithm that finds successive values for \( i \) such that \( f(i) \to 0 \).

Newton’s method proceeds as follows.

1. Estimate the root \( r \) for the equation \( f(i) = 0 \). Our first estimate is \( i_1 \).

2. Find \( i_2 \), the next estimate of the root where the tangent to the curve at \( (i_1, f(i_1)) \) crosses the \( x \)-axis. The equation for the tangent line is

\[
y - f(i_1) = f'(i_1)(i - i_1)
\]

3. The line crosses the \( x \)-axis at a point with coordinates \( i = i_2, y = 0 \). Substituting the foregoing values into equation (1) yields

\[
0 - f(i_1) = f'(i_1)(i_2 - i_1)
\]

4. Solve equation (2) for \( i_2 \).

\[
i_2 = i_1 - \frac{f(i_1)}{f'(i_1)}
\]

5. Equation (3) can be generalized to produce a sequence \( i_1, i_2, \cdots, i_n, i_{n+1} \) such that when \( |i_{n+1} - i_n| < \varepsilon, \ f(i_{n+1}) \approx 0 \).

\[
i_{n+1} = i_n - \frac{f(i_n)}{f'(i_n)}
\]
Example: $x^2 = 5$
1. $f(x) = x^2 - 5$
2. $f'(x) = 2x$
3. $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$
4. $x_{n+1} = x_n - \frac{x^2 - 2}{2x}$

```cpp
#include <iostream>
#include <fstream>
using namespace std;
#define eps 1e-6

//Function g implements f'(x)=2x
double g(double x){return 2*x;}

//Function f implements f(x)=x**2-5
double f(double x){return x*x-5.0;

//Function Newton employs Newton's method to find a root of the equation f(x)=x**2-5
double Newton(double x)
{ double x1,x2;
  for (; ;)
  {
    x2=x1-f(x1)/g(x1);
    if (fabs(x2-x1)<eps) return x2;
    x1=x2;
  }
}

int main()
{ double x;
  for (; ;)
  {
    cout << "Enter an estimate for x<>0. ";
    cin >> x;
    if (fabs(x)<eps) {
      cout << "The value you entered is too close to zero. ";
      continue;
    }
    break;
  }
  cout << endl;
  cout << "x=" << Newton(x) << ";";
  cout << endl;
  return 0;
}
```

**Figure 1.** Program p01 that implements Newton’s method to find roots for the equation $f(x) = x^2 - 5$
References: