```cpp
#include <iostream>
#include <math.h>

#define E 0.000000001
#define KROK 0.1
#define NMAX 1000

using namespace std;
int fx; // numer wybranej funkcji
int iter; // ilość iteracji

// Obliczanie wartości f(x)
long double f(long double x) {
    switch (fx) {
        case 1: return x * x + sin(x) - 1;
            break;
        case 2: return (x - 1)*(x - 2)*(x - 3);
            break;
        case 3: return x * x * x * x * x - 6 * x + log(x);
            break;
        case 4: return (x * x - 1) / (x - 3);
            break;
        default: return -1;
            break;
    }

}
// Obliczanie wartości pochodnej. "st == 1" oznacza pierwszą pochodną, "st == 2" drugą

long double pochodna(long double x, int st) {
    switch (fx) {
        case 1:
        {
            if (st == 1)
                return 2 * x + cos(x);
            if (st == 2)
                return 2 - sin(x);
            break;
        }
        case 2:
        {
            if (st == 1)
                return 3 * x * x - 12 * x + 11;
            if (st == 2)
                return 6 * x - 12;
            break;
        }
        case 3:
        {
            if (st == 1)
                return 5 * x * x * x * x + 1 / x - 6;
            if (st == 2)
                return (20 * x * x * x * x * x - 10) / (x * x);
            break;
        }
        case 4:
        {
            if (st == 1)
                return (x * x - 6 * x + 1) / ((x - 3)*(x - 3));
            if (st == 2)
                return 16 / ((x - 3)*(x - 3)*(x - 3));
```

```
62                 break;
63             }
64             default: return -1;
65                 break;
66         }
67 }
68 // Szukanie miejsca zerowego metodą bisekcji
69 void bisekcja(long double a, long double b) {
70         if (iter + 1 >= NMAX) {
71             cout << "Bisekcja: MAX ITER ERROR!\n";
72             return;
73         }
74         iter++;
75 
76         long double x1 = (a + b) / 2.0;
77 
78         if (fabs(a - b) < E) {
79             if (f(a) * f(b) < 0 && fabs(f((a + b) / 2.0)) < E) {
80                 cout << "Miejsce zerowe w " << (a + b) / 2.0 << " po " << iter
    << " iteracjach." << endl;
81                 iter = 0;
82             }
83             return;
84         } else {
85             if (f(a) * f(x1) < 0) {
86                 bisekcja(a, x1);
87             }
88             if (f(b) * f(x1) < 0) {
89                 bisekcja(x1, b);
90             }
91         }
92 }
93 
94 int main() {
95         cout.precision(15);
96         long double a, b;
97         cout << "Wybierz funkcję: \n"
98                 << "[1] x^2+sin(x)-1\n"
99                 << "[2] (x-1)(x-2)(x-3)\n"
100                << "[3] x^5-6x+ln(x)\n";
101        cin >> fx;
102        cout << "Podaj a i b:\n";
103        cin >> a >> b;
104        cout << "BISEKCJA:\n";
105        for (double i = a; i <= b; i += KROK) {
106            if (i + KROK <= b)
107                bisekcja(i, i + KROK);
108            else if (fabs(i - b) > E)
109                bisekcja(i, b);
110        }
111        iter = 0;
112        cout << "STYCZNYCH:\n";
113        long double x0 = 0;
114        for (double i = a; i <= b; i += KROK) {
115            if (f(i) * pochodna(i, 2) > 0) {
116                x0 = i;
117            } else
118                x0 = i + KROK;
119            for (int ii = 1; ii < NMAX; ii++) {
120                iter++;
121 
122                if (pochodna(x0, 1) != pochodna(x0, 1))
```

```
123                     break;
124                 x0 = x0 - (f(x0) / pochodna(x0, 1));
125                 if (fabs(f(x0)) < E)
126                     break;
127                 if (iter + 1 >= NMAX) {
128                     cout << "Stycznych: MAX ITER ERROR!\n";
129                     return 0;
130                 }
131 
132             }
133             if (fabs(f(x0)) < E && (x0 <= i + KROK && x0 >= i))
134                 cout << "Miejsce zerowe: " << x0 << " po " << iter << " iteracj
    ach." << endl;
135             iter = 0;
136         }
137         return 0;
138 }
139 
140 
```