EXERCISE 1

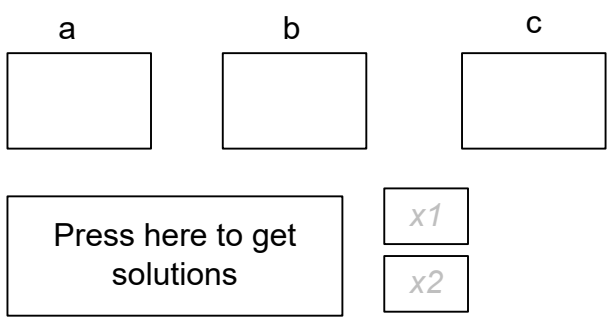
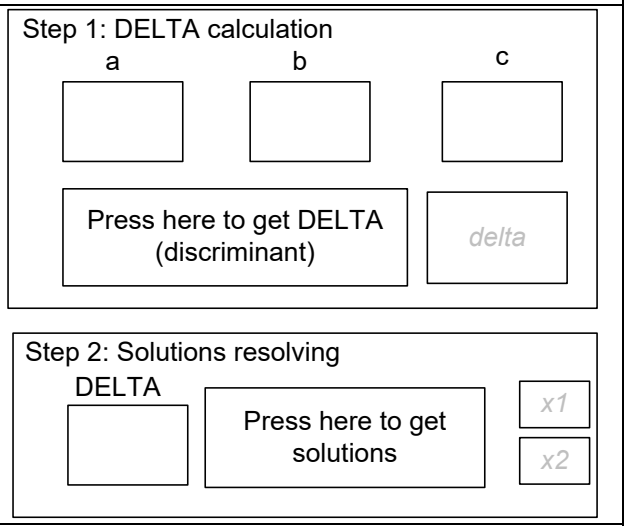
NAME: **STUDENT CODE:**

I. VERIFICATION AND VALIDATION

Description: The purpose is to help users to solve a 2-degree equation (ax^2+bx+c).

Spec: Given input of *a*, *b*, and *c*; the system returns the outputs of x_1 and x_2 (extreme cases are temporarily not considered)

Two systems are developed as follows.

<table>
<tr><th>SYSTEM 1</th><th>SYSTEM 2</th></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td>Code:
x1 = (-b +sqrt(DELTA))/2a
x2 = -b -sqrt(DELTA/2a)</td><td>Code:
DELTA = (b*b-4*a*c)
x1 = (-b +sqrt(DELTA))/2a
x2 = (-b -sqrt(DELTA))/2a</td></tr>
</table>

What are the problems of those two systems? Write down your answer here.

II. TEST-CASES

Description: Some input values

a) How many test-cases we need for the following function *f*1. What are they?

```
int f1(int x) {
    if (x > 10)
        return 2 * x;
    else
        return -x;
}
```

b) Check if your test-cases can detect error if *f*1 is implemented as follows

```
int f1(int x) {
  if (x > 10)
    return 2 * x;
  else if (x > 0)
    return -x;
  else
    return 2 * x;
```

```
}
```

In this case, how many test-cases we need to test this function? What are they?

c) How many test-cases we need to test this function? What are they?

```
int f2(int x) {
  if (x < 10)
    return 2 * x;
  else if (x < 2)
    return -x;
  else
    return 2 * x;
}
```

In this case, how many test-cases we need to test this function? What are they?

d) How many test-cases we need to test this function? What are they?

```
int f3(int x) {
  if (log(x * x * cos(x)) < 3 * x)
    return 2 * x;
  else
    return 2 * x;
}
```

e) Check if your test-cases can detect error if *findMax* is implemented as follows

```
int findMax(int num1, int num2, int num3) {
    int max = 0;
    if ((num1 > num2) && (num1 > num3))
        max = num1;
    if ((num2 > num1) && (num2 > num3))
        max = num2;
    if ((num3 > num1) && (num3 > num2))
        max = num3;
    return max;
}
```

In this case, how many test-cases we need to test this function? What are they?

III. PRATICE 1

- Mô tả bài toán, các input / output có thể có của bài toán
- Xây dựng các test cases kiểm tra tính đúng đắn chương trình
- Viết đoạn mã tự động kiểm tra chương trình cho bên dưới đúng hay sai?

```
#include <iostream>
#include <cmath>

using namespace std;

int solveQuartic(double a, double b, double c, double x[]) {
    if (a == 0 && b == 0 && c == 0) {
        return -1;
    }
```

```
    if (a == 0 && b == 0) {
        return 0;
    }

    if (a == 0) {
        double y = -c / b;
        if (y < 0) return 0;
        x[0] = sqrt(y);
        x[1] = -sqrt(y);
        return 2;
    }

    double delta = b * b - 4 * a * c;
    if (delta < 0) return 0;

    double y1 = (-b + sqrt(delta)) / (2 * a);
    double y2 = (-b - sqrt(delta)) / (2 * a);

    int count = 0;
    if (y1 >= 0) {
        x[count++] = sqrt(y1);
        x[count++] = -sqrt(y1);
    }
    if (y2 >= 0 && y2 != y1) {
        x[count++] = sqrt(y2);
        x[count++] = -sqrt(y2);
    }

    return count;
}

int main() {
    double a, b, c;
    cin >> a >> b >> c;

    double x[4];
    int n = solveQuartic(a, b, c, x);

    if (n == -1) {
        cout << "Infinite solutions." << endl;
    } else if (n == 0) {
        cout << "No solution." << endl;
    } else {
        cout << "The equation has " << n << "real solution(s): ";
        for (int i = 0; i < n; i++) {
            cout << x[i] << " ";
        }
```

```
        cout << endl;
    }

    return 0;
}
```

---o0o---

(End)