```
#include <iostream>
#include <stdlib.h>
#include <cmath>
#include <vector>
using namespace std;

double fn(double x) {
	double y = exp(2 * x) * sin(3 * x);
	return y;
}

double fn2(double x) {
	double y = pow(cos(x), 2);
	return y;
}

typedef double (* func)(double args);

double compSimpson(double a, double b, int n, func fn) {
	double h = (b - a) / n;
	double xi0 = fn(a) + fn(b);
	double xi1 = 0;
	double xi2 = 0;

	for (int i = 0; i < n; i++) {
		double x = a + (i * h);
		if (i % 2 == 0) {
			xi2 = xi2 + fn(x);
		} else {
			xi1 = xi1 + fn(x);
		}
	}
	double xi = h * (xi0 + 2 * xi2 + 4 * xi1) / 3;
	return xi;
}

double compTrap(double a, double b, int n, func fn) {
	double h = (b - a) / n;
	double xi1 = fn(a);
	double xi2 = 0;
	double xi3 = fn(b);

	for (int i = 0; i < n; i++) {
		double x = a + (i * h);
			xi2 = xi2 + fn(x);
	}
```

```
        double xi = h * (xi1 + 2 * (xi2) + xi3) / 2;
        return xi;
}


void Romberg(double a, double b, int n, double R[10][10], func fn) {
        int i, j, k;
        double h = b - a;
        double sum;

        R[0][0] = (h / 2) * (fn(a) + fn(b));

        for (i = 1; i < n; i++) {
                h = h / 2;
                sum = 0;
                for (k = 1; k <= pow(2, i) - 1; k += 2) {
                        sum += fn(a + k * h);
                }

                R[i][0] = R[i-1][0] / 2 + sum * h;
                for (j = 1; j <= i; j++) {
                        R[i][j] = R[i][j-1] + (R[i][j-1] - R[i-1][j-1] / (pow(4.0, j) - 1));
                }
        }

        for (i = 0; i < n; i++) {
                for (j = i; j >= 0; j--) {
                        cout << R[i][j] << "\t";
                }
                cout << endl;
        }
 }

int main() {
        double simp10 = compSimpson(0, 2, 10, fn);
        double simp100 = compSimpson(0, 2, 100, fn);
        double simp1000 = compSimpson(0, 2, 1000, fn);
        cout << "Simpson n = 10: " << simp10 << endl;
        cout << "Simpson n = 100: " << simp100 << endl;
        cout << "Simpson n = 1000: " << simp1000 << endl;

        double trap10 = compTrap(0, 2, 10, fn);
        double trap100 = compTrap(0, 2, 100, fn);
        double trap1000 = compTrap(0, 2, 1000, fn);
        cout << "Trap n = 10: " << trap10 << endl;
        cout << "Trap n = 100: " << trap100 << endl;
```

```
        cout << "Trap n = 1000: " << trap1000 << endl;

        double R[10][10];
        Romberg(-1, 1, 10, R, fn2);

}
```

OUTPUT:

```
Simpson n = 10: -14.2022
Simpson n = 100: -14.214
Simpson n = 1000: -14.214
Trap n = 10: -13.8041
Trap n = 100: -14.2099
Trap n = 1000: -14.2139
```

Romberg:

```
0.583853
2.38924 1.29193
4.64389 2.40159 1.41611
9.27902 4.67637 2.41824 1.44514
18.553  9.29472 4.68447 2.42284 1.45228
37.1037 18.5609 9.29868 4.68652 2.42402 1.45406
74.2062 37.1076 18.5629 9.29967 4.68703 2.42432 1.4545
148.412 74.2081 37.1086 18.5634 9.29992 4.68716 2.42439 1.45461
296.823 148.413 74.2086 37.1088 18.5635 9.29998 4.68719 2.42441 1.45464
593.646 296.824 148.413 74.2087 37.1089 18.5635 9.3     4.6872  2.42441 1.45465
```