Problems 1 and 2

By now, it should be clear how I handle main.cpp, assign5.hpp, and in assign5.cpp void assign5::main(). Since these problems are small and simple enough, I kept all other code in assign5.cpp for this assignment. Common code:

#include <functional>

#include <iostream>

#include <vector>

#include <math.h>

double Trapm(const std::vector<double> f, double h) {

int n = f.size();

double sum = f[0];

for (int i = 1; i < n - 1; ++i) {

sum += 2 \* f[i];

}

sum += f[n];

return h \* sum / 2;

}

double Trapm(const std::vector<double> x, const std::vector<double> y) {

int n = x.size();

double sum = 0;

for (int i = 1; i < n; ++i) {

sum += (y[i] + y[i - 1]) \* (x[i] - x[i - 1]) / 2;

}

return sum;

}

double Trapm(std::function<double(int)> f, double a, double b, int n) {

double h = (b - a) / n;

double sum = f(a);

for (int i = 1; i < n - 1; ++i) {

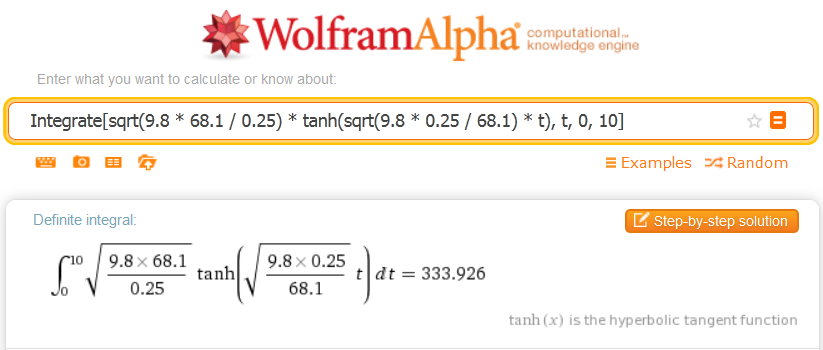
sum += 2 \* f(a + h \* i);

}

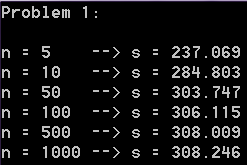
sum += f(b);

return h \* sum / 2;

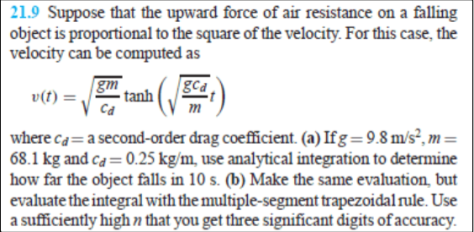
}



Part a: since we can use regular calculators I figured wolfram alpha would be fine.



Part b



Problem 1

Code for problem 1:

void Prob1() {

const double a = 0, b = 10,

g = 9.8, m = 68.1, c = 0.25;

auto v = [&g, &m, &c](double t) {

return sqrt(g \* m / c) \*

tanh(sqrt(g \* c / m) \* t);

};

std::cout

<< "Problem 1:" << std::endl << std::endl

<< "n = 5 --> s = " << Trapm(v, a, b, 5) << std::endl

<< "n = 10 --> s = " << Trapm(v, a, b, 10) << std::endl

<< "n = 50 --> s = " << Trapm(v, a, b, 50) << std::endl

<< "n = 100 --> s = " << Trapm(v, a, b, 100) << std::endl

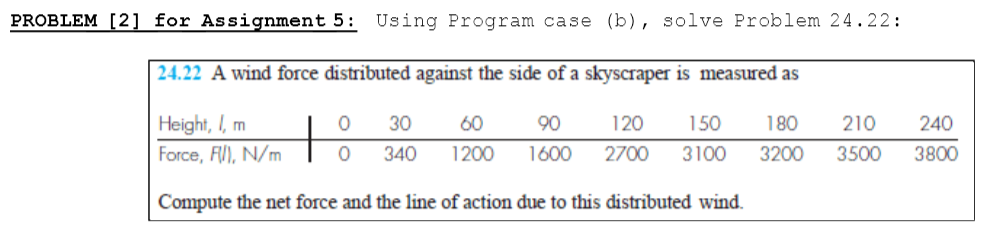
<< "n = 500 --> s = " << Trapm(v, a, b, 500) << std::endl

<< "n = 1000 --> s = " << Trapm(v, a, b, 1000) << std::endl

<< std::endl;

}

Problem 2



Code for problem 2:

void Prob2() {

const std::vector<double> x {0, 30, 90, 120, 150, 180, 210, 240};

const std::vector<double> y {0, 340, 1200, 1600, 2700, 3100, 3200, 3500, 3800};

std::cout << "Problem 2: " << Trapm(x, y) << std::endl << std::endl;

}