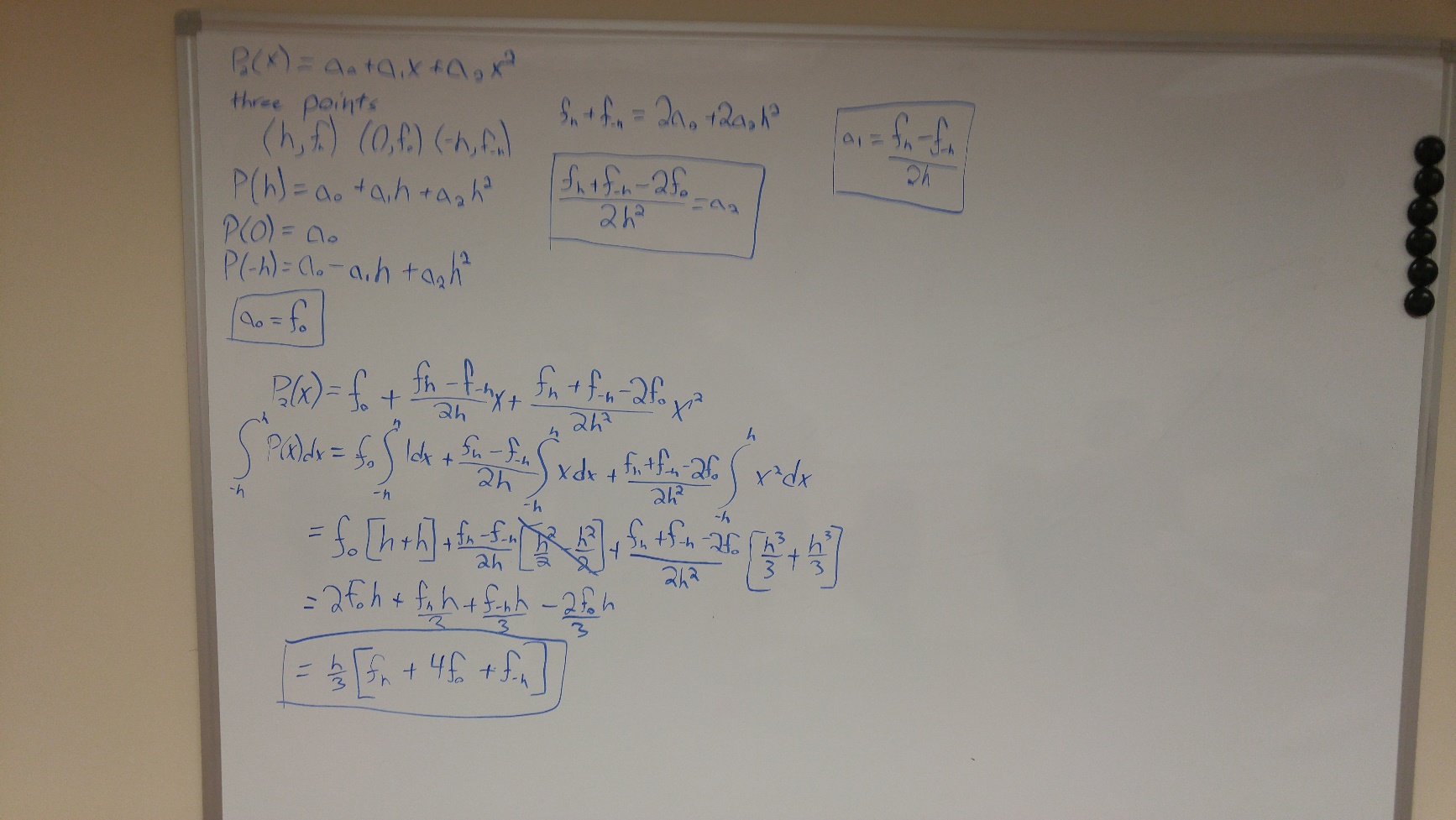
1. 

monte\_carlo\_method\_for(10000000)

function answer = monte\_carlo\_method\_for(how\_accurate)

itterator = 0;

successes = 0;

while (itterator < how\_accurate)

x = rand();

y = rand();

if (x^2 + y^2 <= 1)

successes = successes + 1;

end

itterator = itterator + 1;

end

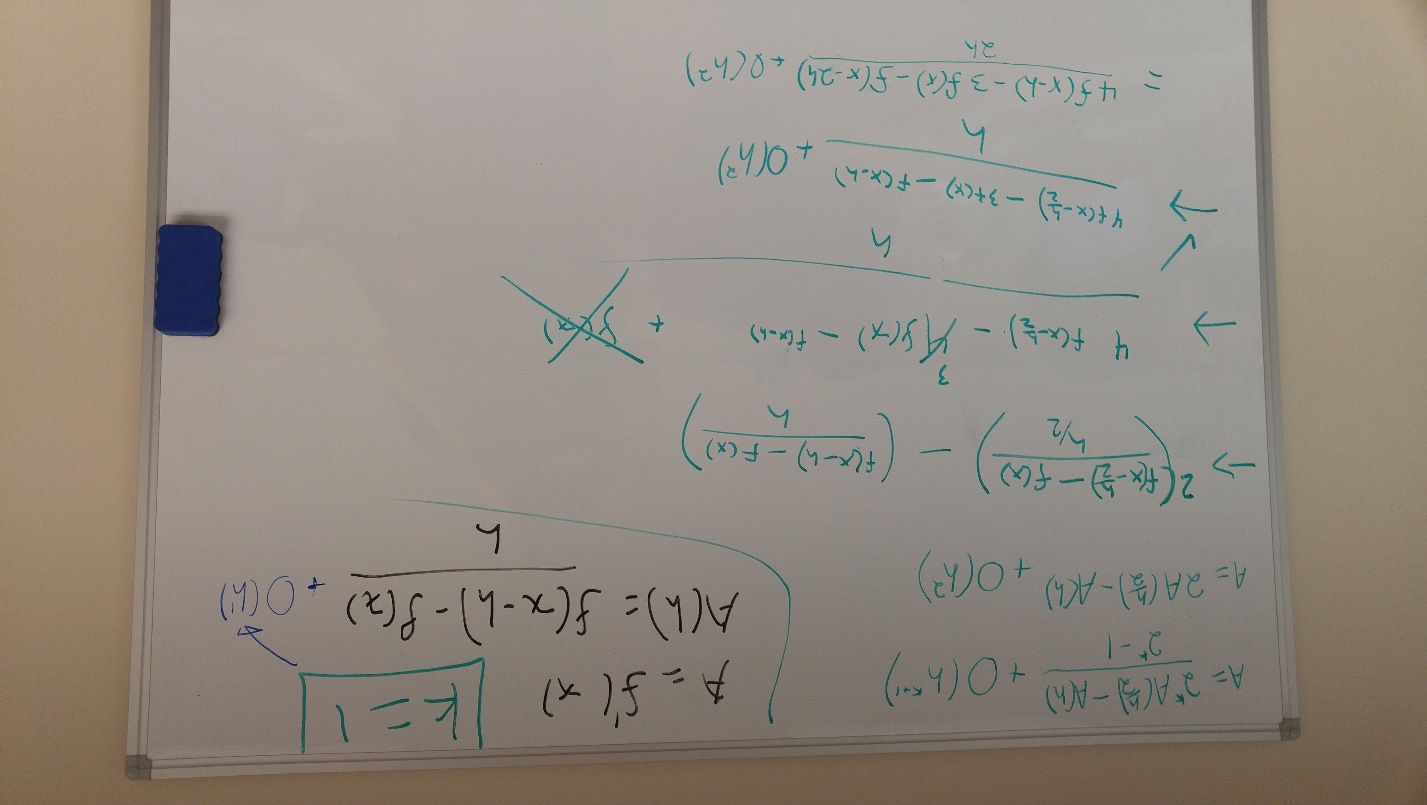
answer = successes / how\_accurate;

%must times by four because of four quadrants

answer = answer \* 4;

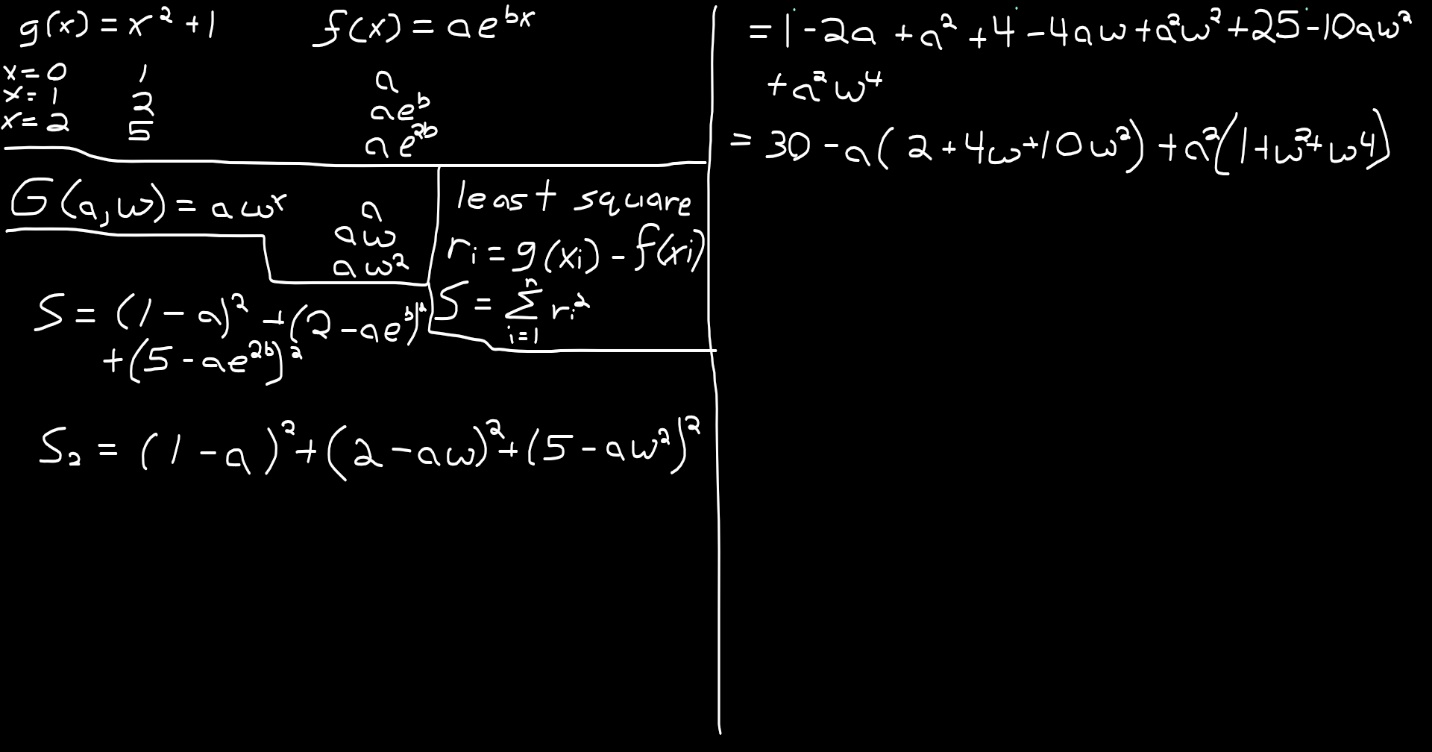
end

1. i) k = 1 because the forward derivative has an error of O(h^1)

ii) 

(Please note I worked with Colton on this problem and our solutions may be the same)

iii) This reminds me of the One-sided Derivative at Boundary Point formula on week 3 slide 12

1. 

equation = @(a,b) (1 - a).^2 + (2 - a \* exp(b)).^2 + (5-a\*exp(2\*b)).^2;

answer = fminsearch(@(u) equation(u(1),u(2)) , [1,1])

x = 0:0.1:2;

function1 = x.^2 + 1;

function2 = 0.8838\*exp(0.8644\*x);

%please note the numbers used were found with the above code

% a = 0.8838, b = 0.8644

plot(x,function1,x,function2)

iv) This is a linear partial derivative if we take w at 0 for da/dx and a at 0 for dw/dx respectively. We get G’(da/dx,0) = 2da/dx + 2a\*da/dx and G’(0,dw/dx) = 0.