Math 3300 Programming Assignment 7

Instructions: Work on the following 2 programs and submit your source code to me via Blackboard. Send me 2 cpp files.

1. Newton's method is an algorithm that can be used to solve an equation of the form $f(x) = 0$ using a sequence of approximations. Here's how it works: You make a reasonable guess for a solution, x_0.

   To get the next term in the sequence, we evaluate $x_1 = x_0 - f(x_0)/f'(x_0)$.

   To get the next term in the sequence, we evaluate $x_2 = x_1 - f(x_1)/f'(x_1)$.

   In general, the $(i+1)^{st}$ approximation is given by:

   $$x_{i+1} = x_i - f(x_i)/f'(x_i)$$

   where $f'(x)$ is the derivative of f.

   You are to write a program that attempts to find solutions to a *polynomial* equation of the above form (this is not always possible). To do this, you will ask the user to enter the degree of the polynomial (i.e. the highest power of x). A polynomial is not a standard C++ data type. To represent a polynomial, you will instead store the coefficients of the polynomial in an array. You will create an array of the appropriate size, and have the user enter the coefficients of the polynomial into that array.

   For example: If the user wishes to enter a degree 3 polynomial, create an array of size 4 to store the coefficients in. So $2 - 5x^2 + 4x^3$ might be represented as an array: $\{2, 0, -5, 4\}$ (the x^1 coefficient is 0 in this example).

   You should create a **function** which calculates the value of the polynomial at a specific x value (the inputs for the function should be the array representing the polynomial, its size, and the x-value to evaluate at), and another **function** which calculates the value of the derivative at a specific x value (same inputs as the previous function).

   You should continue looping until the difference between successive x-values in the sequence become very small. This is how you tell if the method is working. Sometimes Newton's method fails, however, depending on your choice of the initial guess. So you shouldn't just loop until you get an answer, or you might get an infinite loop (maybe stop the calculation after looping a certain number of times like 1000 or whenever 2 consecutive x numbers gets smaller than say .001).

Try to enter the following examples in my sample program:

Solve $2x^4 - 3x^2 + 4x - 5 = 0$ with an initial guess of $x = 1$. You should get a solution $x = 1.23354$. Here's the first couple steps of the algorithm:

First, the derivative is $f'(x) = 8x^3 - 6x + 4$.

$x_0 = 1. f(x_0) = -2, f'(x_0) = 6$

and so:

$x_1 = x_0 - f(x_0)/f'(x_0) = 1 - (-2)/6 = 4/3 = 1.33333$

Now

$x_1 = 1.33333, f(1.33333) = 107/81, f'(1.33333) = 404/27$

and so:

$x_2 = x_1 - f(x_1)/f'(x_1) = 1.24505$

and:

$x_3 = 1.24505 - f(1.24505)/f'(1.24505) = 1.23371$

and:

$x_4 = 1.23371 - f(1.23371)/f'(1.23371) = 1.23354.$

etc.

We see that eventually the difference between sequential terms gets very small. For example, here $|x_4 - x_3| = .00017$ (smaller than .001).

If you try to solve $x^5 - x + 1 = 0$ with an initial guess of $x = 0$, you'll see the sequence does not converge to a solution.

2. You will write a program which will factor a polynomial of the type $ax^2 + bx + c$ or indicate that the polynomial will not factor (a, b, and c are integers).

   Examples:

   (a) $x^2 + 3x + 2 = (x + 1)(x + 2)$
   (b) $x^2 - 7x + 10 = (x - 5)(x - 2)$
   (c) $2x^2 + 7x + 5 = (2x + 5)(x + 1)$
   (d) $4x^2 - 9 = (2x + 3)(2x - 3)$
   (e) $2x^2 + 3x = x(2x + 3)$
   (f) $2x^2 + 6x + 8 = 2(x^2 + 3x + 4)$ (this is as far as the polynomial can be factored)
   (g) $x^2 + 3x + 4$ doesn't factor

   You will do this by writing separate functions to handle different parts of this problem.

   I illustrate the way that I would like you to approach this problem below:

If you want to factor $ax^2 + bx + c$, you can first try to factor out the greatest common divisor of the numbers a, b, and c. After factoring out this number, you try to find any pairs of factors of the product $a \cdot c$ which add together to equal b (Note: After factoring out the GCD, the numbers a, b, and c may have changed). If there are no such pairs of factors, the polynomial cannot be factored.

Here's an example: Let's factor $12x^2 + 2x - 24$. The GCD of these 3 numbers is 2, after factoring that out we have: $2(6x^2 + x - 12)$. We now have that $a = 6, b = 1, c = -12$.

We calculate $a \cdot c = 6 \cdot (-12) = -72$. We then need to find any pairs of factors of -72 which add together to equal $b = 1$. You can see that will be 9 and -8. Once we find those two factors, we factor by splitting the x term into these factors:

$$6x^2 + x - 12 = 6x^2 + 9x - 8x - 12$$

Notice that here we wrote the x term as $9x + (-8)x$ since 9 and -8 were our 2 factors. Now we factor by grouping the first 2 terms and the last 2 terms.

$$(6x^2 + 9x) + (-8x - 12)$$

We factor these by finding the GCD of each set of numbers. The GCD of 6 and 9 is 3, and the GCD of -8 and -12 is -4. And so we have:

$$3x(2x + 3) + (-4)(2x + 3) = (2x + 3)(3x - 4)$$

after factoring out the $(2x + 3)$ part from each term.

So $12x^2 + 2x - 24 = 2(2x + 3)(3x - 4)$.

Another example:

Factor $3x^2 - 15x - 18$. First we notice that 3 factors in to each term, so factoring that out, we have: $3(x^2 - 5x - 6)$. The product of -6 and 1 is -6. Factors of -6 which add to -5 are -6 and 1. So:

$$3(x^2-5x-6) = 3(x^2-6x+x-6) = 3(x(x-6)+1(x-6)) = 3(x-6)(x+1)$$

Here's how I want you to set up this program:

(a) Write a function which prompts the user to enter the three integers a, b, and c, checks if they are valid entries (invalid will mean that $a = 0$, or that both $b = 0$ and $c = 0$), and if not, asks them to enter another set of values (explaining what was wrong). The function has no input, and will "return" these 3 **values**. This function should be called **prompt**.

(b) Factor out the GCD of the numbers a, b, and c (this would be given by $g = \gcd(\gcd(a, b), c)$). If $g = 1$, this means there are no common factors, but either way, you should replace a with a/g, b with b/g, and c with c/g (the GCD will never be 0). Do this by creating a program named **gcd** whose inputs are 2 integers. This function should return the integer corresponding to the GCD. I have shown the GCD algorithm in class. In order for the formula below to always work, gcd(p,q) should have the same sign as p (the first argument).

(c) Write a function which calculates whether there are any pairs of factors of $a \cdot c$ which add to be b. The function should "return" **both** values. If there are no such factors, it should "return" values 0 and 0. This function should be called **findFactors**. You have done most of the work for this function in a previous programming assignment.

(d) If the factors exist (and are non-zero), let's call them $f1$ and $f2$, a formula for the factored polynomial is then:

$$g\Big(\gcd(a, f1)\, x + \gcd(f2, c)\Big)\Big(\frac{a}{\gcd(a, f1)}\, x + \frac{c}{\gcd(f2, c)}\Big)$$

(e) Write a function called **display** which will display the factored polynomial to the monitor using the formula in the previous part, or indicate that it doesn't factor. This function should not return a value.

Requirements:

You will have five functions in your program: main, prompt, findFactors, gcd, display. You should make sure they satisfy the requirements listed in the setup above. It will be easier if you focus on each function separately (and figure out if they will work), and don't just try to throw it all together haphazardly. I have provided template for the **main** function. If your functions work as I require up above, your program should work.

You'll know you've done the program correctly if you test it on the examples I gave above, and the output is the same factored polynomial that I have written (up to permutation of the factors), or if there are only silly differences like $x + -3$ instead of $x - 3$, or $1x + 2$ instead of $x + 2$.