Math 450

Programming Assignment 1

Due: Friday October 5th by 4:00pm

Let $f(x) = x^6 + 7x^5 - 15x^4 - 70x^3 + 75x^2 + 175x - 125$.

1. Prove that $f(x)$ has a root in $[0, 1]$ and another root in $[2, 3]$.
2. Write a program that carries out the $Bisection\, Method$ on $f(x)$. You do not need to make your program take arbitrary input (i.e. you can tailor it to this specific $f(x)$). Your program should take as input the appropriate number of initial guesses, an interval $[a, b]$ and a desired error tolerance TOL. It should output a root r that is within the desired tolerance and it should output N, the number of iterations it took to get to within the error tolerance. Make it clear what procedure you are using to compute the error of your approximations.
3. Write a program that carries out $Newton's\, Method$ on $f(x)$. You do not need to make your program take arbitrary input (i.e. you can tailor it to this specific $f(x)$). Your program should take as input the appropriate number of initial guesses, an interval $[a, b]$ and a desired error tolerance TOL. It should output a root r that is within the desired tolerance and it should output N, the number of iterations it took to get to within the error tolerance. Make it clear what procedure you are using to compute the error of your approximations.
4. Write a program that carries out the $Secant\, Method$ on $f(x)$. You do not need to make your program take arbitrary input (i.e. you can tailor it to this specific $f(x)$). Your program should take as input the appropriate number of initial guesses, an interval $[a, b]$ and a desired error tolerance TOL. It should output a root r that is within the desired tolerance and it should output N, the number of iterations it took to get to within the error tolerance. Make it clear what procedure you are using to compute the error of your approximations.
5. Write a program that carries out $Modified\, Newton's\, Method$ on $f(x)$. You do not need to make your program take arbitrary input (i.e. you can tailor it to this specific $f(x)$). Your program should take as input the appropriate number of initial guesses, an interval $[a, b]$ and a desired error tolerance TOL. It should output a root r that is within the desired tolerance and it should output N, the number of iterations it took to get to within the error tolerance. Make it clear what procedure you are using to compute the error of your approximations.
6. Write a program that carries out $Aitken's\, Method$ on $f(x)$. You do not need to make your program take arbitrary input (i.e. you can tailor it to this specific $f(x)$). Your program should take as input the appropriate number of initial guesses, an interval $[a, b]$ and a desired error tolerance TOL. It should output a root r that is within the desired tolerance and it should output N, the number of iterations it took to get to within the error tolerance. Make it clear what procedure you are using as your underlying recursion, and how you are computing the error of your approximations.
7. Write a program that carries out $Steffensen's\, Method$ on $f(x)$. You do not need to make your program take arbitrary input (i.e. you can tailor it to this specific $f(x)$). Your program should take as input the appropriate number of initial guesses, an interval $[a, b]$ and a desired error tolerance TOL. It should output a root r that is within the desired tolerance and it should output N, the number of iterations it took to get to within the error tolerance. Make it clear what procedure you are using as your underlying recursion, and how you are computing the error of your approximations.
8. Write a program that carries out $Müller's\, Method$ on $f(x)$. You do not need to make your program take arbitrary input (i.e. you can tailor it to this specific $f(x)$). Your program should take as input the appropriate number of initial guesses, an interval $[a, b]$ and a desired error tolerance TOL. It should output a root r that is within the desired tolerance and it should output N, the number of iterations it took to get to within the error tolerance. Make it clear what procedure you are using to compute your initial points p_1 and p_2, and what procedure you are using to compute the error of your approximations.
9. Use each of the programs you wrote to find an approximation of the root of $f(x)$ as given above on the interval $[0, 1]$ to within 7 decimal places of accuracy (when appropriate, use $p_0 = 0$. Otherwise, use reasonably chosen initial values).
10. If possible, use each of the programs you wrote to find an approximation of the root of $f(x)$ as given above on the interval $[2, 3]$ to within 7 decimal places of accuracy (first try using $p_0 = 2$, then try $p_0 = 2.5$).
11. Based on the results of your algorithms, comment on the relative effectiveness of these methods of approximating roots. Which appears to be the fastest? Which appears to be the slowest?

    Extra Credit:

    Suppose that the population of the United States (in millions of people with t measured in years since 1790) is given by the function $f(t) = \frac{1900}{3.71875 + 450e^{-0.023675t}}$ while the non-white population of the United States is given by $g(t) = \frac{2250}{5.2345 + 2750e^{-0.23675t}}$. Use these models to predict when a majority of the people living in the United States will be non-white. [Hint: re-formulate this as a root-finding problem and then find an approximation using one of the algorithms you coded. Be sure to give a clear interpretation of your solution].