

CS 2073, Computer Programming with Engineering Applications
Spring Semester, 1992
Programming Assignment 3
Root of an Equation

The objective of this programming assignment is to find a real number x satisfying $\cos(x) = x$, where $0 \leq x \leq \pi/2$. (From a graph you can see that the line $y = x$ crosses $y = \cos(x)$, and the x -coordinate of the point where they cross is what we want.)

You are to try out two different root-finding methods for finding this x value, applying them to the equation $f(x) = \cos(x) - x$. (A root is a place where $f(x) = 0$, or $\cos(x) - x = 0$, or $\cos(x) = x$.)

1. Newton's method. The first approach will use *Newton's method*. For this method, one also needs the derivative of f , $f'(x) = -\sin(x) - 1$. Newton's method starts with a guess at the answer, call it x_0 , and produces a new, hopefully better guess x_1 using the "magic" formula

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Replace the old guess with the new one and repeat in a loop until the old guess and the new guess are within some small value ϵ of one another. In this assignment we will take ϵ to be 0.00001. You should use the new guess as your final answer. Your program must use two Pascal FUNCTIONS named "f" and "fprime" as shown below, so that the magic formula actually uses these functions.

Use $x_0 = 0$ for the initial guess.

```
function f(x: real): real;
begin
  f := cos(x) - x
end; (* f *)

function fprime(x: real): real;
begin
  fprime := -sin(x) - 1.0
end; (* fprime *)
```

2. Bisection method. The second approach will use the *bisection method*. Here one starts with two values x_1 and x_2 (with $x_1 < x_2$) for which $f(x_1)$ and $f(x_2)$ have opposite signs (i.e., one is positive and the other is negative, or $f(x_1) * f(x_2) < 0$). If f is a continuous function (no jumps or breaks), then there must be an x between x_1 and x_2 for which $f(x) = 0$. To get closer to that value, let $x_{mid} = (x_1 + x_2)/2$, the midpoint. Consider $f(x_{mid})$ and replace either x_1 or x_2 by x_{mid} , so that after the replacement we still have $f(x_1)$ and $f(x_2)$ with opposite signs, but now x_1 and x_2 are half as far apart as they were before. Repeat this process until the distance between x_1 and x_2 is less than ϵ , again taken to be 0.00001 in this assignment. Your final answer should be x_{mid} . Again you must use a Pascal FUNCTION named “f”.

Use the numbers 0 and $\pi/2$ as the initial values.

Directions for both approaches. As before your program should first print “This program written by:” followed by your name. Your program should next read the value 0.00001 for the number ϵ used to determine how close an approximation to the root one gets. (This number can be read from the standard file “input”, i.e., the terminal.) Then print “Tolerance value:” and the value for ϵ . For each of the methods the program must *count* the iterations and must terminate the loop in case the iteration count gets larger than 30. First you should print “Newton’s method:” and a short list of approximate values obtained at each iteration, ending with a final approximation. Next print “Bisection method” and a somewhat longer list of *pairs* of values obtained at each stage of the second method, again ending with the final approximation. Notice that you should get (almost) the same answer by the two methods. All real number must be printed out with 7 digits to the right of the decimal place.

Extras. If you are interested but for no extra credit, you could play around with other values for ϵ . You could also use the type “double” in place of “real” everywhere in your program, to get much more accurate answers. (In Turbo Pascal, “double” is a non-standard way to specify about 16 digits of accuracy rather than the (about) 11 digits of accuracy that one gets with “real”.) Try $\epsilon = 0.000000001$ and print numbers with 15 digits to the right of the decimal place.