EECS 10: Assignment 3

Prepared by: Sundar Krishnamurthy, Prof. Rainer Doemer

October 8, 2010

Due Monday 18 Oct 2010 at 12:00 pm

1 Homework Problem 1: Compute the approximate value of tan(x) [25 Points]

Write a C program to calculate the value of tan x. The result can be approximated using an infinite sum (Taylor expansion, x is in radian NOT in degree):

1a) Use Taylor series of Sine and Cosine functions below and compute sin(x) and cos(x) for any given value of x. For good precision, use first 5 terms in the Sine or Cosine series.

$$\sin(x) = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n+1}}{(2n+1)!} = x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040} + \frac{x^9}{362880} + \ldots$$

$$\cos(x) = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n}}{(2n)!} = 1 - \frac{x^2}{2} + \frac{x^4}{24} - \frac{x^6}{720} + \frac{x^8}{40320} + \ldots$$

1b) Use Taylor series of Tangent function below to compute tan(x) for given input value x. Use 7 terms from the Tan series as shown below.

$$\tan(x) = x + \frac{x^3}{3} + \frac{2}{15}x^5 + \frac{17}{315}x^7 + \frac{62}{2835}x^9 + \frac{1382}{155925}x^{11} + \frac{21844}{6081075}x^{13} + \ldots$$

1c) Divide Sin(x) by Cos(x) computed in step 1a and print along with the result of Tan(x) computed in Step 1b.

Your program must use only the basic operations such as addition, subtraction, multiplication and division. Also, please follow the same programming style as discussed in Lecture 5 for the cosine function (i.e. do not use any loops in your program). In order to make your code simple for higher power terms, precompute x^2, x^3, x^5, x^6 in temporary variables, and use them throughout. Use a maximum of 4 multiplications for each term in all the series above. (Hint: $x^8 = x^5 * x^3; x^{11} = x^6 * x^5$)

When executed, your program output should look as follows:

```
Please enter the real value x (in radians):0.8
Approximately, Sin(0.800000) = 0.717356, Cos(0.800000) = 0.696707
Approximately, Tan(0.800000) = 1.029569, Sin(0.800000)/Cos(0.800000)=1.029639
```

Note: All variables declared and used need to be of type ”long double”. Use appropriate type specifiers while printing.

You should submit your program code as file **`tan.c`**, a text file **`tan.txt`** briefly explaining how you designed your program, and a typescript **`tan.script`** which shows that you compile your program and run it using the values -0.7, 0.6 and 1.2 as inputs.

2 Bonus: Boundary values checking for tan(x) [5 points]

Tan(x) is almost linear in the range $-\frac{\pi}{2}+\varepsilon$ to $\frac{\pi}{2}-\varepsilon$. As x approaches $-\frac{\pi}{2}$ and $\frac{\pi}{2}$, tan(x) reaches $-\infty$ and ∞ respectively. Hence add boundary check conditions such that tan(x) program in previous problem accepts input "*x*" values only within $-\frac{\pi}{2}+\varepsilon$ to $\frac{\pi}{2}-\varepsilon$.

In this problem, choose π as 3.1416 and ε as 0.2708, so that tan(x) produces valid output for x within range $-1.3 < x < 1.3$. If x is outside this range, calculate and print sin(x) and cos(x), but print the following error message and exit the program with status 1.

```
Tan(x) could not be computed. Enter input x within range -1.3 to  1.3.
```

When executed, your program output should look as follows:

```
Please enter the real value x (in radians):1.4
Approximately, Sin(1.400000) = 0.985451, Cos(1.400000) = 0.169975
Tan(x) could not be computed. Enter input x within range -1.3 to 1.3.
```

NOTE: Include the code for this problem in the same **`tan.c`** file that you submit for Problem 1. In the **`tan.txt`** file, explain the logic that you used for checking boundary values. Your **`tan.script`** should contain program output for x = -2.4, -0.7, 0.6, 1.2, 1.5

3 Homework Problem 2: Calculate the solution for a system of linear equations [15 Points]

Cramers rule (source: http://en.wikipedia.org/wiki/Cramer_rule) is a theorem in linear algebra devised by Gabriel Cramer to calculate the solution of a system of linear equations by using determinants.

In a two variable case:
ax + by = e
cx + dy = f,

which in matrix format is

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} e \\ f \end{bmatrix}$$

then x and y can be found with Cramers rule as

x = $\begin{vmatrix} e & b \\ f & d \end{vmatrix}$ divided by $\begin{vmatrix} a & b \\ c & d \end{vmatrix}$ or x = $\frac{ed-bf}{ad-bc}$

And

y = $\begin{vmatrix} a & e \\ c & f \end{vmatrix}$ divided by $\begin{vmatrix} a & b \\ c & d \end{vmatrix}$ or y = $\frac{af-ec}{ad-bc}$

Finally, there is an exception in Cramers rule when the determinant $\begin{vmatrix} a & b \\ c & d \end{vmatrix} = 0$, the Cramer's rule cannot be applied (Denominator is 0 and division is meaningless).

Your linear equations solver program should contain the following sections:

1. Data input: Let the user enter valid equation coefficients in the following format:

```
Please enter the coefficients of the following equations:
***************
* ax + by = e *
* cx + dy = f *
***************
a = 1.0
b = 8.0
c = 4.0
d = 9.0
e = 6.0
f = 1.0
```

2. Data preprocessing: Handle the exception for the 0 denominator
If the denominator ad - bc = 0, print
Cramers rule cannot be applied to these coefficients
Hint: because a, b, c, d are all in floating-point format, it is better to determine the 0 denominator condition by using $|ad - bc| < \varepsilon$, instead of ad – bc = 0; for your implementation, set ε = 1e-5 and use double type variables.

3. Computation: Use Cramers rule to calculate the solution.

4. Output the solution. Use the following format:

```
The solution of the linear system is:
x = -2.000000
y = 1.000000
```

You should submit your program code as file **`cramer.c`**, a text file **`cramer.txt`** briefly explaining how you designed your program, and a typescript **`cramer.script`** which shows that you compile your program and run it. Use the following coefficient as inputs:

a	*b*	*c*	*d*	*e*	*f*
6	15	9	20	3	2
3	4	6	7	5	8
4	8	2	4	8	4

4 Submission

Submission for these files will be similar to last week's assignment. The only difference is that you need to create a directory called **`hw3/`**. Put all the files for assignment 3 in that directory and run the **`/ecelib/bin/turnin`** command to submit your homework.