MU ALPHA THETA NATIONAL CONVENTION 2007
GRAPHING CALCULATOR PROGRAMMING COMPETITION

Your two-person team is provided with two TI-83 graphing calculators, a graphing calculator link cable, a TI-83 instruction manual, and a folder of program requirements. You may not have any other calculators, books, electronic devices, or papers. Do not open the folder or remove the cover from a calculator until instructed to do so. Each calculator has a sticker on it. Write your school name on each sticker.

You will have _____ minutes to complete as many of the programs as possible. You will be given a warning at the 15 minute, 5 minute, and 1 minute remaining marks. Each program **must** have the name that is in bold at the top of the program requirement. The maximum possible point value for each program is given below the name of the program.

Each program will be tested with five predetermined sets of inputs. Each program will scored on how many of these input value sets yield the correct outputs.

- All of the calculators have been reset to their factory defaults.

- There is no need to change any mode of a calculator at any time. Be aware that all calculators will be reverted to the default modes before they are scored.

- If the same program appears on both calculators for your team, then the program with the higher score will count toward your team's score. Ideally, you and your partner should be working on different programs.

- You may create programs with other names on your calculator, and you may call those programs from other programs. For example, if you want to create a program called **MOD** that finds the remainder of integer division and stores it in the variable **M**, you may do that. Of course, only programs with names matching those on the program requirements will be scored.

- Unless otherwise specified, the format of numeric output should be the default for the TI-83. Any number that is not an integer will display as a decimal with 10 significant digits.

- There is no need to perform any type of input validation. All test inputs will be valid, as specified in the prompts.

- There is no need to provide any "user-friendly" input prompting. For example, if you are instructed to prompt for an integer that you plan to store in variable slot **Q**, you can simply use the instruction "Prompt Q" or "Input Q". You don't need to display a sentence such as "PLEASE ENTER AN INTEGER."

- Each ouput requested must be the result of a single **DISP** statement. Do not use the **OUTPUT** statement or the **CLRSCR** statement. Do not display any output that is not specified in the program requirements.

SAS
(20 points)

This program finds the missing side of a triangle when the other two sides and their included angle are known.

The program will prompt for and accept these three inputs, in order:

1. the length of one side of the triangle (you may assume this will be a positive number.)
2. the measure (in degrees) of the included angle (you may assume this will be a positive number greater than 0 and less than 180.)
3. the length of another side of the triangle (you may assume this will be a positive number.)

The program will provide one output:

- the length of the third side of the triangle.

SSA
(55 points)

This program finds all possible lengths of the missing side of a triangle when the other two sides and a non-included angle are known.

The program will prompt for and accept these three inputs, in order:

1. the length of one side of the triangle (you may assume this will be a positive number.)
2. the length of another side of the triangle (you may assume this will be a positive number.)
3. the measure (in degrees) of the angle that is adjacent to the second side entered, but not adjacent to the first side entered (you may assume this will be a positive number greater than 0 and less than 180.)

The program will provide either one or two outputs, according to the following:

- If no triangle is possible with the given constraints, the program will output the word **NONE** .
- If exactly one triangle is possible with the given constraints, the program will output the length of the third side.
- If two triangles are possible are possible with the given constraints, the program will output the both possible lengths of the third side.

NUMFACTR
(25 points)

This program determines how many positive integers are factors of a given positive integer.

The program will prompt for and accept one input:

1. the number to analyze (you may assume this will be a positive integer.)

The program will provide one output:

- The number of positive integers that are factors of the given number (including 1 and the number itself.)

QUADFORM
(35 points)

This program finds the real solutions to a given quadratic equation in the form $ax^2 + bx + c = 0$.

The program will prompt for and accept these three inputs, in order:

1. The coefficient of the x^2 term (a) (you may assume this will be a number.)
2. The coefficient of the x term (b) (you may assume this will be a number.)
3. The constant term (c) (you may assume this will be a number.)

The program will provide either one or two outputs, according to the following:

- ➢ If the quadratic equation has no real solutions, then the program will output the word **NONE** .
- ➢ If the quadratic equation has exactly one distinct real solution, then the program will output the solution.
- ➢ If the quadratic equation has two distinct real solutions, then the program will output both solutions.

OCTTODEC
(25 points)

This program converts a positive integer from base-8 to base-10.

The program will prompt for and accept one input:

1. the number to convert (you may assume this will be a positive integer with no digit greater than 7.)

The program will provide one output:

- The input number in base-10.

DECTOOCT

(40 points)

This program converts a positive integer from base-10 to base-8.

The program will prompt for and accept one input:

1. the number to convert (you may assume this will be a positive integer.)

The program will provide one output:

- The input number in base-8.

TWOPOINT

(30 points)

This considers two points with integer coordinates, and finds: the midpoint of the points, the distance between the points, and the slope of the line through the points.

The program will prompt for and accept these four inputs, in order:

1. The *x*-coordinate of the first point (you may assume this will be an integer.)
2. The *y*-coordinate of the first point (you may assume this will be an integer.)
3. The *x*-coordinate of the second point (you may assume this will be an integer.)
4. The *y*-coordinate of the second point (you may assume this will be an integer.)

The program will provide the following three outputs, in order:

- The *x* and *y* coordinates of the midpoint between the points, separated by a space or comma (no parentheses are necessary.)
- The distance between the points.
- The slope of the line through the points. If the slope is not an integer, then the slope must be given as a fraction**.** If the slope is undefined, then output the word **UNDEF**.

CIRCLEEQ
(35 points)

This program gives the equation of a circle in the form $Ax^2 + By^2 + Cx + Dy + E = 0$, using the coordinates of the circle's center and the radius of the circle.

The program will prompt for and accept these three inputs, in order:

1. The *x*-coordinate of the circle's center (you may assume this will be a number.)
2. The *y*-coordinate of the circle's center (you may assume this will be a number.)
3. The radius of the circle (you may assume this will be a positive number.)

The program will provide one output:

- The equation of the circle in the form $A\mathbf{X}^2 + B\mathbf{Y}^2 + C\mathbf{X} + D\mathbf{Y} + E = 0$. If any coefficients are zero, then that term should be excluded. For example, if $C = 0$, then the *y*-term should be excluded – do not print "0*y*". If a coefficient is negative, then a minus sign instead of a plus sign should be displayed. For example, if $C = -4$, then the term should be displayed as $"-4y"$, not $"+^{-}4y"$.

REGNGON1
(15 points)

This program considers a regular polygon with a given number (*N*) sides, and determines the measure of one interior angle and the number of diagonals in the polygon.
.
The program will prompt for and accept one input:

1. the number of sides of the polygon (you may assume this will be an integer greater than 2.)

The program will provide two outputs, in order:

- The measure, in degrees, of one interior angle of the polygon.
- The number of diagonals in the polygon.

REGNGON2
(35 points)

This program considers a regular polygon with a given number (*N*) sides and a given side length (*X*), and determines the perimeter of the polygon, the length of the polygon’s apothem, and the polygon’s area.
.
The program will prompt for and accept two inputs, in order:

1. the number of sides of the polygon (you may assume this will be an integer greater than 2)
2. the length of one side of the polygon (you may assume this will be a positive number.)

The program will provide three outputs, in order:

- The perimeter of the polygon.
- The length of the polygon’s apothem.
- The area of the polygon.

SIMPRAD
(40 points)

This program simplifies a radical of the form $\sqrt{A}$, where A is a positive integer, and reduces it to simplest form.

This program accepts one input:

1. the number (A) under the radical (you may assume this is a positive integer.)

This program provides one output:

- The number $\sqrt{A}$ in simplest radical form.
  - If A is a perfect square, then no square root symbol should be displayed. For example, if $A=9$, the output should be "3".
  - Otherwise, the square root symbol should be utilized. For example, if $A=12$, the output should be $"2\sqrt{\ }(3)"$.
  - If the square root cannot be simplified, you should not display a leading 1. For example, if $A=7$, the output should be $"\sqrt{\ }(7)"$, not $"1\sqrt{\ }(7)"$.

HANDANGL
(30 points)

This program calculates the measure (in degrees) of the smaller angle formed by the hour and minute hand of an analog clock at a given exact time.

This program accepts two inputs:

1. the current hour (you may assume this is an integer from 1 to 12, inclusive.)
2. the current minute (you may assume this is an integer from 0 to 59, inclusive.)

This program provides one output:

- The measure, in degrees, of the angle formed by the clock hands.

TOROMAN
(70 points)

Roman Numeral	**Number**
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

This program converts a positive integer to a Roman Numeral.

A string of letters means that their values should be added together. For example, XXX = 10 + 10 + 10 = 30, and LXI = 50 + 10 + 1 = 61.

If a smaller value is placed *before* a larger one, we subtract instead of adding. For instance, IV = 5 - 1 = 4.

- Subtract only powers of ten, such as I, X, or C. Writing VL for 45 is not allowed: write XLV instead.
- Subtract only a single letter from a single numeral.
  Write VIII for 8, not IIX; likewise, 19 is XIX, not IXX.
- Don't subtract a letter from another letter more than ten times greater. This means that you can only subtract I from V or X, and X from L or C, so MIM is illegal.

This program accepts one input:

1. the number (*N*) to convert (you may assume this is an integer from 1 to 3999, inclusive.)

This program provides one output:

- *N* as a Roman Numeral.

NTHPRIME
(35 points)

This program outputs the N^{th} prime number, given N.

This program accepts one input:

1. Which prime number (N) to find (you may assume this is a positive integer less than 1000.)

This program provides one output:

- The N^{th} prime number.

NTHFIB
(35 points)

This program outputs the N^{th} number in the Fibonacci sequence, given N. The Fibonacci sequence begins with the terms 1 and 1. Each subsequent term is found by adding the two previous terms. So the first ten terms are $1, 1, 2, 3, 5, 8, 13, 21, 34, 55$.

This program accepts one input:

1. Which term in the Fibonacci sequence (N) to find (you many assume this is a positive integer less than 1000.)

This program provides one output:

- The N^{th} term in the Fibonacci sequence.