CS 152 / SE 152
Programming Language Paradigms

Spring Semester 2013

Department of Computer Science
San Jose State University
Prof. Ron Mak

Assignment #1

Assigned: Monday, February 3
Due: Monday, February 10 at 11:59 pm
Team assignment, 100 points max

Algebra 1 nightmare in FORTRAN IV

The purpose of this assignment is to give you some experience writing in a classic programming language, FORTRAN IV. Appreciate the limitations that programmers faced in the 1960s and understand where today's languages come from.

Recall the quadratic formula:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

The formula produces two complex roots, x_1 and x_2:

$(x_{1real}, x_{1imaginary})$ and $(x_{2real}, x_{2imaginary})$

The discriminant $b^2 - 4ac$ determines the nature of the two roots:

Discriminant positive:

$$x_{1real} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$x_{2real} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

$$x_{1imaginary} = x_{2imaginary} = 0$$

Discriminant zero:

$$x_{1real} = x_{2real} = -\frac{b}{2a}$$

$$x_{1imaginary} = x_{2imaginary} = 0$$

Discriminant negative:

$$x_{1real} = x_{2real} = -\frac{b}{2a}$$

$$x_{1imaginary} = \frac{\sqrt{-\left(b^2 - 4ac\right)}}{2a}$$

$$x_{2imaginary} = -x_{1imaginary}$$

Your FORTRAN program quadratic.for should read the following 21 "punched cards" as input data (text file **`quadratic.in`**):

```
1.0       -2.0      1.0
1.0       -10.0     25.0
1.0       -3.0      2.0
2.0       -6.0      4.0
1.0       1.0       -2550.0
10.0      -20.0     10.0
1000.0    -2000.0   1000.0
0.02      -0.04     0.02
1.432     9.876     -4.567
8.813     -1.31     0.0
2.3       1.9917    0.0
1.0       0.0       -1.0
1.0       0.0       1.0
9.0       0.0       36.0
1.0       2.0       5.0
535.0     22.0      1583.0
-9.0      23.0      37.0
61.0      2.0       87.0
61.0      159.0     87.0
100.0     99.0      98.0
0.0
```

Each card contains one set of values for a, b, and c.

Hint: Use the input format specification **`3F10.0`** to read the three values from each card.

The 21st card is an "end of deck" marker.

For each set of input values a, b, and c, use the quadratic formula to compute the two roots x_1 and x_2. **Compute the quadratic formula in a FORTRAN subroutine.**

For each set of input values, your program should output the values of a, b, c, x_{1real}, $x_{1imaginary}$, x_{2real}, and $x_{2imaginary}$. Also print the card count.

Hint: Use the format specification **`1PE15.4`** to output each real number. You can specify a repeat factor such as **`1P3E15.4`**. The **`E`** means exponential notation, and the **`1P`** moves the decimal point to the right by one (e.g., print **`-2.5500E+03`** instead of the default **`-0.2550E+04`**).

Example output:

```
 COUNT         A             B             C          X1 REAL       X1 IMG        X2 REAL       X2 IMG

    1:     1.0000E+00   -2.0000E+00    1.0000E+00    1.0000E+00                    1.0000E+00
    2:     1.0000E+00   -1.0000E+01    2.5000E+01    5.0000E+00                    5.0000E+00
    3:     1.0000E+00   -3.0000E+00    2.0000E+00    2.0000E+00                    1.0000E+00
    4:     2.0000E+00   -6.0000E+00    4.0000E+00    2.0000E+00                    1.0000E+00
    5:     1.0000E+00    1.0000E+00   -2.5500E+03    5.0000E+01                   -5.1000E+01
    6:     1.0000E+01   -2.0000E+01    1.0000E+01    1.0000E+00                    1.0000E+00
    7:     1.0000E+03   -2.0000E+03    1.0000E+03    1.0000E+00                    1.0000E+00
    8:     2.0000E-02   -4.0000E-02    2.0000E-02    1.0000E+00                    1.0000E+00
    9:     1.4320E+00    9.8760E+00   -4.5670E+00    4.3500E-01                   -7.3316E+00
   10:     8.8130E+00   -1.3100E+00    0.0000E+00    1.4864E-01                    0.0000E+00
   11:     2.3000E+00    1.9917E+00    0.0000E+00    0.0000E+00                   -8.6596E-01
   12:     1.0000E+00    0.0000E+00   -1.0000E+00    1.0000E+00                   -1.0000E+00
   13:     1.0000E+00    0.0000E+00    1.0000E+00                  1.0000E+00                  -1.0000E+00
   14:     9.0000E+00    0.0000E+00    3.6000E+01                  2.0000E+00                  -2.0000E+00
   15:     1.0000E+00    2.0000E+00    5.0000E+00   -1.0000E+00    2.0000E+00   -1.0000E+00    -2.0000E+00
   16:     5.3500E+02    2.2000E+01    1.5830E+03   -2.0561E-02    1.7200E+00   -2.0561E-02    -1.7200E+00
   17:    -9.0000E+00    2.3000E+01    3.7000E+01   -1.1189E+00                    3.6744E+00
   18:     6.1000E+01    2.0000E+00    8.7000E+01   -1.6393E-02    1.1941E+00   -1.6393E-02    -1.1941E+00
   19:     6.1000E+01    1.5900E+02    8.7000E+01   -7.8145E-01                   -1.8251E+00
   20:     1.0000E+02    9.9000E+01    9.8000E+01   -4.9500E-01    8.5731E-01   -4.9500E-01    -8.5731E-01

 END OF PROGRAM: 20 CARDS READ.
```

Tips

Do not use modern FORTRAN features. Use only FORTRAN IV (also known as FORTRAN 66). A short reference:
http://www.math-cs.gordon.edu/courses/cps323/FORTRAN/fortran.html

There were no lower-case characters with punched cards, so everything is in upper case.

The letter **`C`** in column 1 is for a comment. Statement numbers are restricted to columns 1 through 5. Statements are in columns 7 through 72.

A non-blank character in column 6 means what follows is a continuation of the previous line. Keep your statements short and column 6 is always blank.

Variable names are limited to 6 characters. You can declare variables with the **`INTEGER`**, **`REAL`**, **`DOUBLE PRECISION`**, and **`LOGICAL`** (boolean) statements. Example:

```
      INTEGER I, J, K
```

However, variable declarations are optional! By default, any variable that starts with the letters **`I`** through **`N`** are type integer, and the rest are type real. You still have to declare **`DOUBLE PRECISION`** and **`LOGICAL`** variables.

The **`LOGICAL`** values are **`.TRUE.`** and **`.FALSE.`** (note the surrounding periods).

The relational operators are: **.EQ. .NE. .LT. .LE. GT. .GE. .NOT. .AND. .OR.** (more surrounding periods).

There are two types of **IF** statements, the logical **IF** and the arithmetic **IF**. The logical **IF** tests a logical (boolean) expression. You can have only one statement as the true part. There is no else part. Examples:

```
IF (A .GT. B) A = 3.2
IF ((A .LE. B) .AND. (C .NE. 0.0)) GO TO 7
```

The arithmetic **IF** tests an arithmetic expression. Guess how this example works:

```
IF (A-B) 110, 201, 3
```

There is also the computed **GO TO** statement, the precursor to the select or case statement in today's languages. Guess how this example works:

```
GO TO (43, 182, 672, 12, 8), K
```

You will undoubtedly stumble onto some unexpected syntax limitations. Remember, this is a nearly 60 years old language.

Windows

Download the Open Watcom FORTRAN development environment:
http://www.openwatcom.org/index.php/Main_Page

It implements FORTRAN 77, which is backwards compatible with FORTRAN IV.

Mac

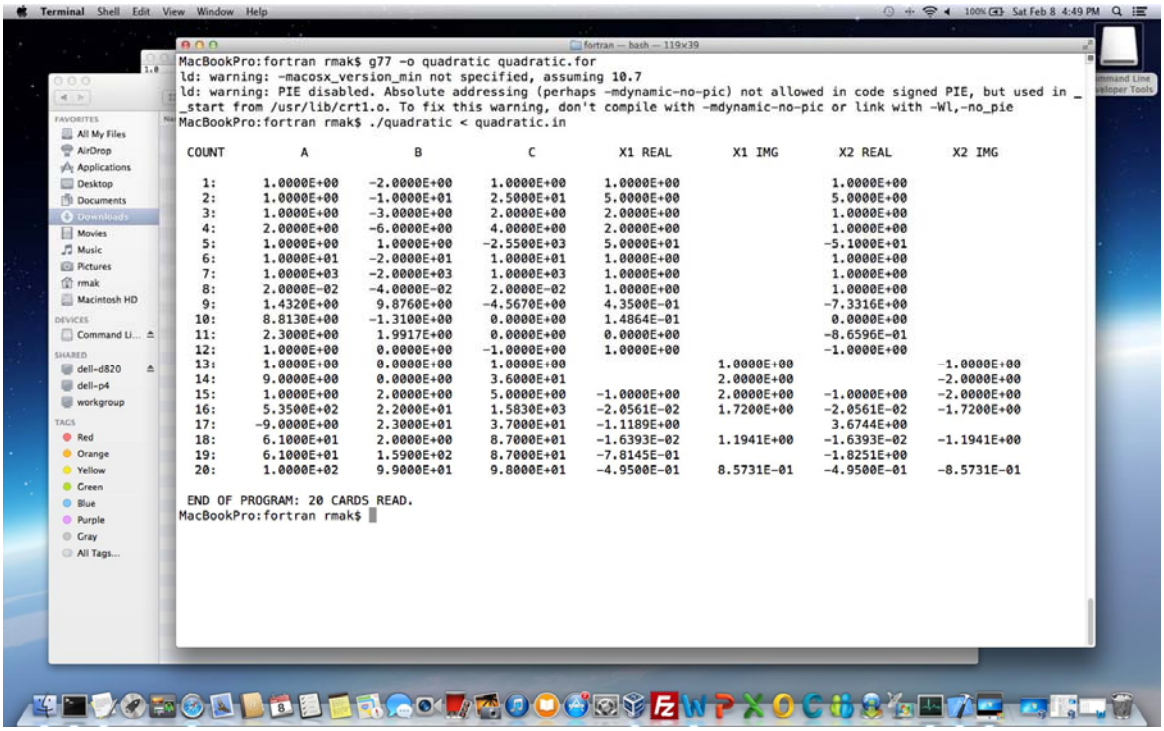
First, you must get gcc (the GNU C compiler) by downloading and installing the command-line tools for Xcode:
http://www.mkyong.com/mac/how-to-install-gcc-compiler-on-mac-os-x/

Next, download g77-intel-bin.tar.gz (g77 3.4) from http://hpc.sourceforge.net/. It will install the FORTRAN compiler into **/usr/local**.

Then you should then be able to compile and run with the commands

```
g77 -o quadratic quadratic.for
./quadratic < quadratic.in
```

Mac screen shot:



What to turn in

Each team turns in one assignment consisting of your FORTRAN source file and a text file of your output. Email them as attachments to: ron.mak@sjsu.edu.

Important: Your subject line should be: **CS 152 Assignment #1,** ***team name***
where ***team name*** is the name of your team.

Be sure to CC all the members of your team so that when I send you your team score, I can just do a “Reply all”.