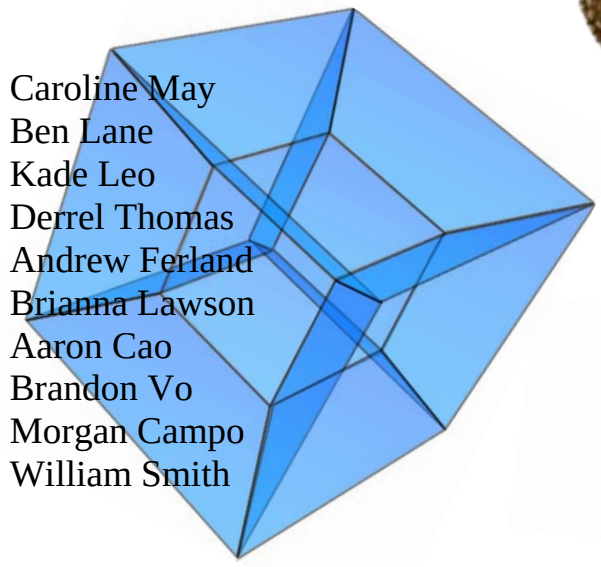
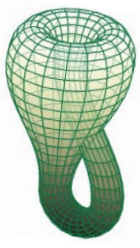
Special Projects 2013

With Mr. Dave Dino Andersen





Caroline May
Ben Lane
Kade Leo
Derrel Thomas
Andrew Ferland
Brianna Lawson
Aaron Cao
Brandon Vo
Morgan Campo
William Smith

Table of Contents

QFORMULA.8XP

Class

Description:

This program solves a quadratic equation of the form…

$$ax^2+bx+c=0$$
$$(a\neq 0)$$

The user must enter the values for a, b, and c. It then uses the quadratic formula …

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

and displays the two solutions as x1 and x2. If the solutions are complex, those are displayed in a+bi form.

File Name: QUADFORM.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 1Ø:46

```
Fix 4
a+bi
FnOff
PlotsOff
ClrHome

Disp "+++++++++++++++"
Disp "+++++++++++++++"
Disp "+++QUADRATIC++++"
Disp "++++FORMULA++++"
Disp "+++++++++++++++"
Disp "+++++++++++++++"
Disp "+++++++++++++++"

Pause

ClrHome
Prompt A
Prompt B
Prompt C
(⁻B+√(B²-4AC))/(2A)→θ
(⁻B-√(B²-4AC))/(2A)→X

ClrHome
Disp "X1="
Disp θ
Disp "X2="
Disp X

Real
Float
Stop
```

CPLXROOT.8xp
By: the class

This program calculates the “n” roots of complex numbers. It will ask the user to input *A*, *B*, and *N* from the form $(a+bi)^{\frac{1}{n}}$. It will then calculate the distance, R, of the point from the origin on the complex plane using $\sqrt{A^2+B^2}$. It will then calculate $\tan^{-1}(B/A)$ and store that in θ. It then displays the original number you entered and then calculates the roots using a “for” loop to increment k starting from zero until $k=n-1$ and use the formula $R^{\frac{1}{n}}(\cos(\frac{\theta+360K}{n})+\sin(\frac{\theta+360K}{n}))$ to calculate the roots and then display these roots.

File Name: CPLXROOT.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 1Ø:46

```
a+bi
Degree
Fix 4
FnOff
PlotsOff
ClrHome

Disp "+++++++++++++++"
Disp ""
Disp "    COMPLEX"
Disp "     ROOTS"
Disp ""
Disp ""
Disp "+++++++++++++++"

Pause
ClrHome

Disp "FIND ROOTS FOR"
Disp "(a+bi)^(1/n)"

Prompt A
Prompt B
Prompt N

√(A²+B²)→R

If A≠Ø
Then
tan⁻¹(B/A)→θ

If (A<Ø and B<Ø)
Then
(18Ø+θ)→θ

:Else
:If (A<Ø and B>Ø)
Then
(18Ø-θ)→θ

Else
If (A>Ø and B<Ø)
Then
(36Ø-θ)→θ

:::End
::End
:End

Else
If B>Ø
Then
9Ø→θ

Else
27Ø→θ

End
```

```
End

Disp "ORIGINAL NUMBER"
A+Bi→X
Disp X
Disp "ROOTS"

For(K,Ø,N-1,1)
(R^(1/N))→S
:cos((θ+36ØK)/N)→C
:sin((θ+36ØK)/N)i→D

:Disp (S*C)+(S*D)


Pause


End

Float
Stop
```

NEWT.8XP

William Smith

Description:

This program solves for the roots of equations (places where the equation touches the X-axis) using Newton's Method.

The user must enter the original equation and the derivative of that equation and the program will graph it.

The user should then choose a starting point (initial guess) and the program will find the next guess and input it as X until it finds the root using the function…

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

File Name: NEWT.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 1Ø:52

```
FnOff
PlotsOff
ClrHome
Disp "0000000000000000"
Disp ""
Disp ""
Disp "  NEWTONS METHOD"
Disp "   BY WILL SMITH"
Disp ""
Disp "0000000000000000"
Pause
ClrHome

Fix 5

Disp "EQUATION"
Input Str1
Disp "DERIVATIVE OF Y"
Input Str2
Str1→Y
Str2→Y₂
Lbl 2
FnOff 2
ClrHome
Disp "    USE GRAPH TO"
Disp " CHOOSE STARTING"
Disp "          POINTS"
Pause
DispGraph
Pause
Disp "START POINT"
Prompt X
Disp "PRESS ENTER TO"
Disp "GUESS AGAIN"
Lbl 1
Y=Ø→A
X-Y/Y₂→X
Disp X
Pause
If Y(X)≠A
Goto 1

If Y(X)=A
Goto 2
Float

Stop
```

Compound.8xp

By Brandon Vo

The program labeled COMPOUND is used to calculate Compound interest. The program asks the user to input the Principal, years accumulated, interest rate, and how often interest is compounded in a year, such as quarterly or biannually. It uses the equation

$$(P*(1+R/N))^{NT}$$

P is money

R is interest rate

N is the amount of times interest compounded

T is the amount of years

You can enter the interest in either percent or decimal.

The program would store the amount of money in a variable and display the amount of money.

File Name: COMPOUND.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 1Ø:54

```
PlotsOff
Disp "1Ø1Ø1Ø1Ø1Ø1Ø1Ø1Ø"
Disp ""
Disp "    COMPOUND"
Disp ""
Disp "    INTEREST"
Disp ""
Disp "Ø1Ø1Ø1Ø1Ø1Ø1Ø1Ø1"
Pause
ClrHome

Disp "INSERT THE MONEY"
Prompt M

Disp "INSERT THE TIME IN YEARS"Prompt T

Disp "INTEREST"
Prompt I

Disp "HOW OFTEN"
Disp "IN A YEAR"
Prompt N

Fix 2

If I<1
I*1ØØ→I

(M*(1+(I*.Ø1)/N)^(NT)→C

FnOff
PlotsOn
ClrHome
Disp "----------------"
Disp ""
Disp "      YOUR"
Disp ""
Disp "    MONEY IS"
Disp C
Disp "----------------"
Pause
Float
Stop
```

ECCENTRI.8xp

Program Description by Aaron Cao

Fundamentally, all conics have a fixed ratio between a point called a focus and a line called a directrix. Though the focus and directrix differ in location from conic to conic, the ratio between them is known as the eccentricity. This program finds the eccentricity of any conic function.

We begin by making a table of all of the conic sections for selection: Parabola, Circle, Ellipse, and Hyperbola. All circles have an eccentricity of 0, so when the user selects the circle in the table, the program will display $e = 0.$ Similarly, all parabolas have an eccentricity of 1, so the program will display $e = 1$ when selected.

The ellipse and hyperbola are a bit trickier. For an ellipse of the form:

$$\frac{X^2}{A^2} + \frac{Y^2}{B^2} = 1$$

The eccentricity is dependent on the values of A and B, or the lengths of the major and minor axis respectively. Our eccentricity for an ellipse is

$$\frac{\sqrt{A^2 - B^2}}{A}$$

Where $\sqrt{A^2 - B^2}$ is known as our value C.

Similarly, for a hyperbola of the form: $\frac{X^2}{A^2} - \frac{Y^2}{B^2} = 1$

Our eccentricity is: $\frac{\sqrt{A^2 + B^2}}{A}$

With $\sqrt{A^2 + B^2}$ as our C value.

With this, the program can display the eccentricity of any conic.

File Name: ECCENTRI.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 1Ø:46

```
a+bi
ClrHome
PlotsOff
Disp "eeeeeeeeeeeeeee"
Disp ""
Disp ""
Disp "  ECCENTRICITY"
Disp ""
Disp ""
Disp "eeeeeeeeeeeeeee"
Pause
ClrHome

Lbl AØ

Menu("WHAT CONIC?","CIRCLE",A1,"PARABOLA",A2,"ELLIPSE",A3,"HYPERBOLA",A4,"GET OUTTA HERE
",A5)

Lbl A1
Disp "e=Ø"

Pause
ClrHome
Goto AØ

:Lbl A2
:Disp "e=1"

:Pause
:ClrHome
:Goto AØ

::Lbl A3
::ClrHome
::Disp "ELLIPSE OF FORM"
::Disp "(X²/A²)+(Y²/B²)"
::Disp "=1"
::Prompt A,B

::If A>B:Then
::√(A²-B²)→C
::C/A→E
::Disp "e="
::Disp E▶Frac

::Else
::If B>A:Then
::√(B²-A²)→C
::C/B→E
::Disp "e="
::Disp E▶Frac

::Pause
::ClrHome
::Goto AØ

:::Lbl A4
:::ClrHome
```

```
:::Disp "HYPERBOLA OF"
:::Disp "FORM"
:::Disp "(X²/A²)-(Y²/B²)"
:::Disp "=1"
:::Prompt A,B

:::√(A²+B²)→C
:::C/A→E

:::Disp "e="
:::Disp E▶Frac

:::Pause
:::ClrHome
:::Goto AØ

::::Lbl A5
::::Output(4,5,"SEE YAH!")
::::Pause
::::ClrHome
::::Stop
```

NEWFIB.8xp

Andrew Ferland

Special Projects 2013 – David Andersen

The program NEWFIB.8xp is designed to display the desired element in the Fibonacci sequence that the user requests, below 480.

The code includes a simple splash screen, displaying the words, “FIBONACCI SEQUENCE”. Then, it prompts the user to input the number of the number of the element in the Fibonacci sequence desired. If the number is less than 480, but greater than 2, then it goes through the process of generating the number, using a “For” loop. If the number is less than 2, then the program displays the number 1.

The Fibonnaci Sequence: 1, 1, 2, 3, 5, 8, 13, …

File Name: NEWFIB.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 13:14

```
FnOff
PlotsOff
ClrHome
Disp "XXXXXXXXXXXXXXXX"
Disp ""
Disp "    FIBONACCI"
Disp "    SEQUENCE"
Disp ""
Disp "XXXXXXXXXXXXXXXX"
Pause
ClrHome


Prompt N
If (N<481) and (N>2)
Then
1→A
1→B
For(I,3,N,1)
A+B→C
B→A
C→B
Disp C
End


Disp C
Else
Disp "CANNOT COMPUTE"
If (N≤2)
Then
Disp "1"
```

DEQ.8XP

By Brianna Lawson and Ben Lane

This program finds the general solution for linear homogeneous differential equations with constant coefficients up to the third order. For example, $a_0y'''+a_1y''+a_2y'+a_3y=0$ where a_0, a_1, a_2, a_3 are constants. First, it will solve the characteristic equation, which is $a_0m^3+a_1m^2+a_2m+a_3=0$. It then uses the output function to output the general solution depending on the roots. If the differential equation was of order one, its root is called *m*. For order two, the roots are called *m* and *n* and for third order, the roots are called *m, n,* and *p*. Because all cubics have at least one real root, the guaranteed real root is *p.* Imaginary answers are in form $m=a+bi$ and if *m* is imaginary, then *n* will be the conjugate such that $n=a-bi.$ The roots affect the answer depending upon whether they are repeating or imaginary. When displaying the solution on the calculator we assume all constants to be 1, however the constants will be included in the following table.

Order	**Roots**	**Output** **General Solution**
1	$m,\ m\in\mathbb{R}$	$y=c_1e^{mx}$
2	$m,\ n;\ m\neq n; m,n\in\mathbb{R}$	$y=c_1e^{mx}+c_2e^{nx}$
2	$m,n; m=n;\ \ m,n\in\mathbb{R}$	$y=c_1e^{mx}+c_2xe^{nx}$
2	$m,n;\ m,n\in\mathbb{C}$	$y=e^{ax}(c_1\cos(bx)+c_2\sin(bx))$
3	$m,n,p;\ m\neq n\neq p\ ; m,n,p\in\mathbb{R}$	$y=c_1e^{mx}+c_2e^{nx}+c_3e^{px}$
3	$m,n,p;\ m=n=p\ ; m,n,p\in\mathbb{R}$	$y=c_1e^{mx}+c_2xe^{nx}+c_3x^2e^{px}$
3	$m,n,;, m=n;\ m,n,p\in\mathbb{R}$	$y=c_1e^{px}+c_2e^{mx}+c_3xe^{nx}$
3	$m,n,p; m=p\ ;\ m,n,p\in\mathbb{R}$	$y=c_1e^{nx}+c_2e^{mx}+c_3xe^{px}$
3	$m,n,p; n=p;\ m,n,p\in\mathbb{R}$	$y=c_1e^{mx}+c_2e^{nx}+c_3xe^{px}$
3	$m,n,p;\ p\in\mathbb{R}; m,n\in\mathbb{C}$	$y=c_1e^{px}+e^{ax}(\cos(bx)+\sin(bx))$

$\mathbb{R}$ = real numbers $\mathbb{C}$ = complex number

File Name: DEQ.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 1Ø:58

```
FnOff
PlotsOff
ClrHome
a+bi
Ø→θ

Disp "***************"
Disp ""
Disp " HL DEQ SOLVER"
Disp ""
Disp "BEN and BRIANNA"
Disp ""
Disp "***************"
Pause
ClrHome

Menu("PICK AN ORDER","1ST ORDER",1,"2ND ORDER",2,"3RD ORDER",3,"QUIT",4)

Lbl 1
1→O
Disp "AY+B=Ø"
Prompt A,B

Lbl 5

solve(AY+B,Y,Ø,{⁻1E99,1E99})→M
Goto 9

Lbl 2
Disp "AY²+BY+C=Ø"
Prompt A,B,C
2→O

Lbl 6

If θ=1:Then:E→A:S→B:T→C:End

((⁻B+√(B²-4AC))/(2A))→M
((⁻B-√(B²-4AC))/(2A))→N

If θ=1:Then:Goto 7:End:

Goto 8

Lbl 3

Disp "AY³+BY²+CY+D=Ø"
Prompt A,B,C,D
3→O
solve(AY³+BY²+CY+D,Y,Ø,{⁻1E99,1E99})→P
A→E:AP+B→S:AP²+BP+C→T:AP³+BP²+CP+D→U


1→θ
Goto 6


Disp "ROOTS:"
```

```
Disp ""
Lbl 7
Disp P
Lbl 8
Disp N
Lbl 9
Disp M

Disp ""
Disp "CONTINUE?"
Pause
prgmDIF
Lbl 4
ClrHome
Stop
```

File Name: DEQ2.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 1Ø:58

```
Fix 2
a+bi

If O=1
Then
Output(3,1,"e^")
Output(3,3,M)
Output(3,7,"X")

Else

If O=2
Then
If imag(M)=Ø and M=N
Then
Output(3,1,"e^")
Output(3,3,M)
Output(3,7,"X")
Output(3,8,"+")
Output(3,9,"Xe^")
Output(3,12,M)
Output(3,16,"X")
Pause
ClrHome
Else
If M≠N
Then
Output(3,1,"e^")
Output(3,3,M)
Output(3,7,"X+e^")
Output(3,11,N)
Output(3,15,"X")
Pause
ClrHome
End
End
```

File Name: DIF.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 1Ø:59

```
a+bi
Fix 2

ClrHome
Disp "GENERAL SOLUTION"
Disp "C=1, Y="

If O=1
Then
prgmDEQ2
Pause
ClrHome
End

If O=2
Then
If imag(M)≠Ø
Then
Output(3,1,"e^"
Output(3,3,real(M)
Output(3,7,"X(cos("
Output(3,13,imag(M))
Output(4,2,"X)+sin("
Output(4,9,imag(M))
Output(4,13,"X))")
Pause
ClrHome
End
End

If O=2
Then
If imag(M)=Ø
Then
prgmDEQ2
Disp ""
End
End

If O=3
Then
If imag(M)≠Ø
Then

Output(3,1,"e^")
Output(3,3,P)
Output(3,7,"X+e^")
Output(3,11,real(M))
Output(3,15,"X")
Output(4,2,"(cos(")
Output(4,7,imag(M))
Output(4,11,"X+sin(")
Output(5,1,imag(M))
Output(5,5,"X))")

Pause
ClrHome
End
```

```
End

If O=3
Then
If imag(M)=Ø
Then
If M=N and N=P
Then
Output(3,1,"e^")
Output(3,3,M)
Output(3,7,"X+Xe^")
Output(3,12,M)
Output(4,2,"X+X² e^")
Output(4,8,M)
Output(4,12,"X")
Pause
ClrHome
End
End
End
If O=3
Then
If imag(M)=Ø
Then
If M=N and M≠P
Then
Output(3,1,"e^")
Output(3,3,P)
Output(3,7,"X+e^")
Output(3,11,M)
Output(3,15,"X+")
Output(4,2,"Xe^"
Output(4,5,M)
Output(4,9,"X")
Pause
ClrHome
End
End

If O=3
Then
If imag(M)=Ø
Then
If M=P and M≠N
Then
Output(3,1,"e^")
Output(3,3,M)
Output(3,7,"X+e^")
Output(3,11,N)
Output(3,15,"X+")
Output(4,2,"Xe^")
Output(4,5,N)
Output(4,9,"X")
Pause
ClrHome
End
End

If O=3
Then
If imag(M)=Ø
```

```
Then
If N=P and M≠P
Then
Output(3,1,"e^")
Output(3,3,M)
Output(3,7,"X+e^")
Output(3,11,N)
Output(3,15,"X+")
Output(4,2,"Xe^")
Output(4,5,N)
Output(4,9,"X")
Pause
ClrHome
End
End

If O=3
Then
If imag(M)=Ø
Then
If M≠N and N≠P and M≠P
Then
Output(3,1,"e^")
Output(3,3,M)
Output(3,7,"X+e^")
Output(3,11,N)
Output(3,15,"X+")
Output(4,2,"e^")
Output(4,4,P)
Output(4,8,"X")
Pause
ClrHome
End
End

If O=1
Then
Disp "A"
prgmDEQ2
Pause
ClrHome
End

Stop
```

Kade Leo
Special projects - D. Andersen

Primes.8xp

Description:

The program finds all prime numbers up to a given maximum, which cannot exceed 999. The program uses the Sieve of Eratosthenes to calculate primes. The sieve finds primes by eliminating all multiples of prime numbers.

First the program calls ERASPLAS.8XP to provide a splash screen which says “THE SIEVE OF ERATOSTHENES”. Then you are prompted to input a number less than 1000. Finally, it displays prime numbers in a 4x7 grid. If the primes do not fit on one screen, the user can hit enter to see the next set of numbers.

File Name: PRIMES.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 13:18

```
prgmERASPLAS
ClrHome
Disp "INPUT A NUMBER"
Disp "LESS THAN 1ØØØ"
Prompt N
999→dim(L)
For(K,1,N,1)
K→L(K)
End
ClrHome
For(K,2,N,1)
For(J,2K,N,K):Ø→L(J)
End
End

1→C
1→R
For(K,2,N,1)
If L(K)≠Ø
Then
Output(R,C,L(K)
C+4→C
If C>16
Then
1→C
R+1→R
End

If R>7
Then
Pause
ClrHome
1→R
End
End
End
Pause
```

File Name: ERASPLAS.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 13:17

```
FnOff
PlotsOff
ClrHome
Disp "***************"
Disp ""
Disp "   THE  SEIVE"
Disp "       OF"
Disp "  ERATOSTHENES
Disp ""
Disp "***************"
Pause
ClrHome
Return
```

PIRATES.8xp
Caroline May

My project is a game titled Pirates. The player controls the letter “P,” using the up/down and left/right arrow buttons. The goal of the game is to pick up a key, the letter “K”, and open a door, the letter “D”, all while avoiding the walls. Once the player reaches the door with the key, they complete the game. The game can be exited at anytime by pressing MODE.

The map used in the game is entered by the user into matrix [J] using a 1 for empty space, a 2 for the player, a 3 for the key, and a 4 for the door. The program will execute any map using these numbers.

Example map:

```
[ [ 1 1 1 1 1 1 1 1 1 1 1 1 1 0 4 0 ]
  [ 1 1 0 0 0 0 1 1 0 0 0 1 1 0 1 0 ]
  [ 0 0 0 1 1 0 1 1 0 3 0 0 0 0 1 0 ]
  [ 0 1 1 1 1 0 0 0 0 1 1 1 1 1 1 0 ]
  [ 0 2 1 1 1 1 1 1 1 1 1 1 1 1 1 0 ]
  [ 0 1 0 0 0 1 1 1 1 0 0 0 0 0 0 0 ]
  [ 0 0 0 1 0 0 0 0 0 0 1 1 1 1 1 1 ]
  [ 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 ] ]
```

File Name: PIRATE.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 13:Ø9

```
Menu("START?","YES",1,"NO",2)

Lbl 1
[J]→[I]
ClrHome
Disp "PRESS MODE TO"
Disp "QUIT AT ANY TIME"
Disp ""
Disp "GET THE KEY (K)"
Disp "TO OPEN THE DOOR"
Disp "(D)"
Pause

ClrHome
For(X,1,8,1)
For(Y,1,16,1)

If ([J](X,Y)≠1 and [J](X,Y)≠2 and [J](X,Y)≠3 and [J](X,Y)≠4)
Then
Output(X,Y,Ø)

Else
If [J](X,Y)=2
Then
Output(X,Y,"P")
Else
If [J](X,Y)=3
Then
Output(X,Y,"K")

Else
If [J](X,Y)=4
Then
Output(X,Y,"D")

End
End
End
End

End
End

Ø→D
Lbl G

Repeat (K=26 or K=25 or K=34 or K=24 or K=22)
getKey→K
End

If (K=26)
Then

For(X,1,8,1)
For(Y,1,16,1)

If ([J](X,Y)=2)
Then
```

```
If ([J](X,Y+1)=3)
Then
1→D
Output(X,Y," P"
1→[J](X,Y)
2→[J](X,Y+1)
Goto G

Else
If ([J](X,Y+1)=4)
Then

If (D≠Ø)
Then
ClrHome
Output(4,5,"YOU WON"
[I]→[J]
Pause
Stop

Else
If (D=Ø)
Then
Goto G

End
End

Else
If ([J](X,Y+1)=1)
Then
Output(X,Y," P"
1→[J](X,Y)
2→[J](X,Y+1)
Goto G

End
End
End

End

End
End


Else
If (K=25)
Then

For(X,1,8,1)
For(Y,1,16,1)

If ([J](X,Y)=2)
Then

If ([J](X-1,Y)=3)
Then
1→D
Output(X-1,Y,"P")
```

```
Output(X,Y," ")
1→[J](X,Y)
2→[J](X-1,Y)
Goto G

Else
If ([J](X-1,Y)=4)
Then

If (D≠Ø)
Then
ClrHome
Output(4,5,"YOU WON"
[I]→[J]
Pause
Stop

Else
If (D=Ø)
Then
Goto G

End
End

Else
If ([J](X-1,Y)=1)
Then
Output(X-1,Y,"P"
Output(X,Y," "
1→[J](X,Y)
2→[J](X-1,Y)
Goto G

End
End
End

End

End
End

Else
If (K=24)
Then

For(X,1,8,1)
For(Y,1,16,1)

If ([J](X,Y)=2)
Then

If ([J](X,Y-1)=3)
Then
1→D
Output(X,Y-1,"P")
Output(X,Y," ")
1→[J](X,Y)
2→[J](X,Y-1)
Goto G
```

```
Else
If ([J](X,Y-1)=4)
Then

If (D=1)
Then
ClrHome
Output(4,5,"YOU WON"
[I]→[J]
Pause
Stop

Else
If (D≠1)
Then
Goto G

End
End

Else
If ([J](X,Y-1)=1)
Then
Output(X,Y," "
Output(X,Y-1,"P"
1→[J](X,Y)
2→[J](X,Y-1)
Goto G

End
End
End

End

End
End

Else
If (K=34)
Then

For(X,1,8,1)
For(Y,1,16,1)

If ([J](X,Y)=2)
Then

If ([J](X+1,Y)=3)
Then
1→D
Output(X+1,Y,"P")
Output(X,Y," ")
1→[J](X,Y)
2→[J](X+1,Y)
Goto G

Else
If ([J](X+1,Y)=4)
Then

If (D=1)
```

```
Then
ClrHome
Output(4,5,"YOU WON"
[I]→[J]
Pause
Stop

Else
If (D≠1)
Then
Goto G

End
End

Else
If ([J](X+1,Y)=1)
Then
Output(X,Y," ")
Output(X+1,Y,"P"
1→[J](X,Y)
2→[J](X+1,Y)
Goto G

End
End
End

End

End
End

Else
If (K=22)
Then
[I]→[J]
ClrHome
Stop

End
End
End
End
End

End
End

Lbl 2
ClrHome
Output(4,5,"GOODBYE"
Pause
```

Binary.89p

Derrel Thomas

Special Projects 2013

This program converts traditional decimal, or base 10, numbers to binary, or base 2. It does this using a backwards "for" loop to repeat the basic conversion template for decimal to binary. It uses logarithms to determine the correct power of 2 to start with for the conversion. This is very important, because starting too high slows down the program, and starting too low makes it produce inaccurate results for higher numbers.

The user need only enter the decimal number to be converted.

Formulas used:

$$\text{decimal} = 2^x$$

$$x = \log_2(\text{decimal})$$

$$x = \frac{\ln(\text{decimal})}{\ln 2}$$

File Name: MAINBI~1.89P

```
()
Prgm

ClrHome
ClrIO
FnOff
PlotsOff

Disp  "{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}"
Disp  ""
Disp  "       Decimal to"
Disp  "    Binary converter"
Disp  ""
Disp  "{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}"

Pause
ClrIO
Lbl  b

Disp  "Decimal?"
Prompt  d
Ø→c

ln(d)/(ln(2))→a
int(a)+1→a
For  x,a,Ø,-1
:If d-2^x≥Ø Then
::c+1Ø^x→c
::d-2^x→d
:EndIf
EndFor

Disp  "In binary it is..."
Disp  c

EndPrgm
```

COMPINT.8xp

Morgan Campo

Simplified Compound Interest—this is to find the principal and any interest from the past years. The equation to find a person's compound interest is:

P = C(1+R)^T

P = the future value of money

C = the initial deposit

R = interest rate (this needs to be expressed as a decimal) (example: 6% = .06)

T = number of years invested

N = this value is not in the equation because it is expressed always as 1. This is the number of times that someone is invested in the compound.

In my program, if you have only three variables (it doesn't matter which one), you can always find the fourth variable.

File Name: COMPINT.8XP Type: Program Protected: No
Comment:Program file Ø5/22/13, 12:Ø6

```
Fix 2
FnOff
PlotsOff
ClrHome
Disp "????????????????"
Disp ""
Disp ""
Disp "    COMPOUND"
Disp ""
Disp "   INTEREST"
Disp ""
Disp "????????????????"

Pause
ClrHome

Lbl AØ
Menu("SOLVE FOR...","P",A1,"C",A2,"R",A3,"T",A4,"QUIT",A5)

Lbl A1
Disp "P=C(1+R)^T
Disp ""
Prompt C
Prompt R
Prompt T

C(1+R)^T→P
ClrHome

Disp "P IS"
Disp ""
Disp P
Pause
Goto AØ

Lbl A2
Disp "C=(P)/(1+R)^T
Disp ""

Prompt P
Prompt R
Prompt T

P/((1+R)^T)→C
ClrHome

Disp "C IS"
Disp ""
Disp C
```

```
Pause
Goto A0

Lbl A3
Disp "R=((P^(1/T))/(C)-1)
Disp ""

Prompt P
Prompt C
Prompt T

((P^(1/T)/C)-1→R
ClrHome
Disp "R IS"
Disp ""
Disp R
Pause
Goto A0

Lbl A4
Disp "T=(ln(P))/(ln(C(1+R))
Disp ""

Prompt P
Prompt C
Prompt R

(ln(P))/(ln(C(1+R)→T
ClrHome
Disp "T IS"
Disp ""
Disp T
Pause
Goto A0


End
Stop
```

RANDBINA.8xp
Program Description by Aaron Cao

This program generates a series of random binary numbers for as long as you need it to run until you quit (by hitting the On button). It is very useful for creating binary seeds or for those moments when you just feel like looking at a bunch of binary numbers on your screen.

The first initial processes that we do are for optimization. First, we store a random integer from 0 to 999 into the variable D using the int( ) function. Next, we store 0 into the variable C. Finally, we do the calculation:

$$\frac{ln(D)}{ln(2)}$$

Which we store into the variable A.

Using the int( ) function again, we store int(A)+1 into A again. All of these steps simply give the power of 2 that our following For( ) loop will repeat. The For( ) loop to generate a binary number for a random integer between 0 and 999 is:

$$For(X, A, 0, -1)$$

With an If:Then statement, we store the binary number into the variable C like so:

$If(D - 2^x) \geq 0$

$Then$

$C + 10^x \to C$ (the store function)

$D - 2^x \to D$ (the store function)

Finally, we end our For( ) loop and our If:Then statement, then display C. If we put a label in the beginning of the program, then after the Disp function, we can insert a Goto function to constantly bring us back to the beginning of the program. This will make the program constantly spit out a binary number until you tell it to stop.

File Name: RANDBINA.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 1Ø:46

```
ClrHome
Lbl B

int(999rand)→D
Ø→C
(ln(D)/ln(2))→A
int(A)+1→A
For(X,A,Ø,⁻1)
:If (D-(2^X))≥Ø
Then
::C+(1Ø^X)→C
::D-2^X→D
:End
End

Disp C
Goto B
```

GAUSS.8xp

By Brianna Lawson

This program will use Gauss Jordan to solve or simplify a system of linear equations. This program does not require the matrix to be invertible. The user will first input the entire matrix into matrix *A* in the calculator. They will then call the Gauss program which will ask if the matrix is in *A* and then ask the dimensions of the coefficient matrix as 2 *x* 2, 3 *x* 3, or 4 *x* 4. The program will simplify the left side into the identity matrix as much as possible. It uses the row functions built into the matrix operations. It will first try to make the values along the diagonal one. It will then use the multiply then add row function to make the other values in each column other than the diagonal to zero. If you get the identity matrix and then one column on the right, this column is the solution to the system. However if it does not simplify to the identity, it will go as far as it can. For example, given system $\begin{matrix} Ax+By+Cz=D \\ Ex+Fy+Gz=H \\ Jx+Ky+Lz=M \end{matrix}$, you will input it into matrix *A* as $\begin{bmatrix} A & B & C & D \\ E & F & G & H \\ J & K & L & M \end{bmatrix}$ which the program will try, if possible to simplify to $\begin{bmatrix} 1 & 0 & 0 & D' \\ 0 & 1 & 0 & H' \\ 0 & 0 & 1 & M' \end{bmatrix}$ which will then print this matrix and *D'* = *x*, *H'* = *y*, *M'* = *z*. If the matrix cannot be reduced to identity form, the answer will need to be parameterized or the user will need to use back substitution to find answer. If this happens, the user needs to know how to interpret matrices after Gauss Jordan.

File Name: GAUSS.8XP Type: Program Protected: No
Comment:Program file Ø5/22/13, 1Ø:29

```
FnOn
PlotsOff
ClrHome
Disp "***************"
Disp ""
Disp ""
Disp "     GAUSS"
Disp ""
Disp ""
Disp "***************"
Pause
ClrHome

Menu("DATA IN MTRX A","YES",1,"NO",2)

Lbl 1
Menu("DIM?","2X2",A,"3X3",B,"4X4",C)
Lbl B

If [A](1,1)≠Ø
Then
*row(1/([A](1,1)),[A],1)→[A]
*row+(⁻[A](2,1),[A],1,2)→[A]
*row+(⁻[A](3,1),[A],1,3)→[A]
End

If [A](2,2)≠Ø
Then
*row(1/([A](2,2)),[A],2)→[A]
*row+(⁻[A](3,2),[A],2,3)→[A]
*row+(⁻[A](1,2),[A],2,1)→[A]
End

If [A](3,3)≠Ø
Then
*row(1/([A](3,3)),[A],3)→[A]
*row+(⁻[A](2,3),[A],3,2)→[A]
*row+(⁻[A](1,3),[A],3,1)→[A]
End
Pause
ClrHome

Lbl A
If [A](1,1)≠Ø
Then
*row(1/[A](1,1),[A],1)→[A]
*row+(⁻[A](2,1),[A],1,2)→[A]
End

If [A](2,2)≠Ø
Then
*row(1/[A](2,2),[A],2)→[A]
*row+(⁻[A](1,2),[A],2,1)→[A]
End

Disp [A]
Pause
ClrHome
```

```
Stop

Lbl C

If [A](1,1)≠Ø
Then
*row(1/[A](1,1),[A],1)→[A]
*row+(⁻[A](2,1),[A],1,2)→[A]
*row+(⁻[A](3,1),[A],1,3)→[A]
*row+(⁻[A](4,1),[A],1,4)→[A]
End

If [A](2,2)≠Ø
Then
*row(1/[A](2,2),[A],2)→[A]
*row+(⁻[A](1,2),[A],2,1)→[A]
*row+(⁻[A](3,2),[A],2,3)→[A]
*row+(⁻[A](4,2),[A],2,4)→[A]
End

If [A](3,3)≠Ø
Then
*row(1/[A](3,3),[A],3)→[A]
*row+(⁻[A](1,3),[A],3,1)→[A]
*row+(⁻[A](2,3),[A],3,2)→[A]
*row+(⁻[A](4,3),[A],3,4)→[A]
End

If [A](4,4)≠Ø
Then
*row(1/[A](4,4),[A],4)→[A]
*row+(⁻[A](3,4),[A],4,3)→[A]
*row+(⁻[A](2,4),[A],4,2)→[A]
*row+(⁻[A](1,4),[A],4,1)→[A]
End
Disp [A]
Pause
ClrHome



Lbl 2
Stop
```

ESCAPE.8XP

By Benjamin Lane

This program finds the escape velocity of a body given the mass of the body and the distance from the body.

The program prompts the user for the mass of a body in kilograms and the distance from the center of the body in kilometers. The equation

$$\sqrt{\frac{2GM}{R}}$$

is used to find the escape velocity where M is mass, R is radius from the center, and G is the gravitational constant ($G=6.67384 \times 10^{-11}\ m^3\ kg^{-1}\ s^{-2}$).

The escape velocity is output in meters per second.

File Name: ESCAPE.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 13:11

```
FnOff
PlotsOff
ClrHome
Disp "***************"
Disp ""
Disp "ESCAPE VELOCITY"
Disp ""
Disp "     BY BEN"
Disp ""
Disp "***************"
Pause
ClrHome

6.67384E-11→G

Disp "CALCULATE ESCAPE"
Disp "VELOCITY"
Disp ""
Disp "MASS OF BODY(KG)"
Input M
Disp "KM FROM CENTER"
Input R

M*1ØØØ→M

√((2GM)/R)→V

ClrHome
Disp "ESCAPE VELOCITY:"
Disp W
Disp "M/S"
Pause
Stop
```

KEP1.8XP

By Benjamin Lane

This program uses Kepler's Third Law to solve for either the mass of the system, the period of the orbit, or the orbital radius given the other two. The mass must be given in kilograms, the period in Earth years, and the orbital radius in astronomical units (the average distance from the Earth to the Sun, about $1.49 \times 10^8 km$).

The equation that is to be manipulated is

$$T^2 = \frac{4\pi^2 A^3}{GM}$$

The user is first asked which of the three variables they are trying to find. They are then prompted to enter the values for the two other variables. The program runs a manipulated version of the equation solved for the missing variable, which is then displayed.

File Name: KEP1.8XP Type: Program Protected: No
Comment:Program file Ø5/22/13, 12:2Ø

```
FnOff
PlotsOff
ClrHome

Disp "***************"
Disp ""
Disp " LAW OF PERIODS"
Disp ""
Disp "     BY BEN"
Disp ""
Disp "***************"
Pause
ClrHome

6.67384*1Ø^-11→G
Disp "T²=(4π²A³)/(GM)"

Disp "T EARTH YEAR"
Disp "A AU"
Disp "M IN KG"
Pause

Menu("FIND...","PERIOD",1,"SEMIMAJOR AXIS",2,"MASS",3,"QUIT",5)

Lbl 1
ClrHome
Disp "SEMIMAJOR AXIS?"
Input A
Disp "MASS?"
Input M

√((4π²A³)/(G(M)))→T
Disp T
Disp "EARTH YEAR"
Pause
Stop

Lbl 2
ClrHome
Disp "PERIOD?"
Input T
Disp "MASS?"
Input M
³√((T²G(M))/(4π²))→A
Disp A
Disp "AU"
Pause
Stop

Lbl 3
ClrHome
Disp "PERIOD?"
Input T
Disp "SEMIMAJOR AXIS?"
Input A
((4π²A³)/(GT²))→M
Disp "M"
Disp M
```

```
Disp "KG"
Pause
Stop

Lbl 5
Stop
```

LRSUMS.8XP

William Smith

This program finds the area under a curve from the left and right using n rectangles. It uses the formula…

$$\int_{A}^{B} f(x)dx$$

…with the starting point A and the ending point B. The program finds the area of the rectangles and then averages the left and right sums for a more accurate approximation.

The user must enter the function first, then the start point, the end point, and the number of divisions.

File Name: LRSUMS.8XP Type: Program Protected: No
Comment:Program file Ø5/21/13, 1Ø:52

```
FnOff
PlotsOff
ClrHome
Disp "0000000000000000"
Disp ""
Disp ""
Disp "  LEFT AND RIGHT"
Disp "            SUMS"
Disp "   BY WILL SMITH"
Disp ""
Disp "0000000000000000"
Pause
ClrHome
Disp "        FUNCTION"
Input Str1
Str1→Y
Disp "     START POINT"
Prompt A
Disp "       END POINT"
Prompt B
Disp "       DIVISIONS"
Prompt N
(B-A)/N→H
A→X
Ø→L
1→J
Lbl 1
L+Y*H→L
X+H→X
IS>(J,N)
Goto 1
B→X
L+Y*H→R
A→X
R-Y*H→R
(L+R)/2→T
Disp "            LEFT"
Disp L
Disp "           RIGHT"
Disp R
Disp "         AVERAGE"
Disp T
```

MESSENGER.8xp
Caroline May

This program allows two connected TI-83/84 calculators to communicate. Both calculators must have the program and one calculator needs to edit the program and switch the string variables (str1 is changed to str2 and vice versa).

The menu screen gives the user the option to send or receive a message. If the user chooses send, the calculator stores the message into a string. When the other calculator chooses receive, the calculator uses the getCalc function to get the string and displays the message.

File Name: MESSAGER.8XP Type: Program Protected: No
Comment:Program file Ø5/22/13, 1Ø:25

```
ClrHome
Lbl R

Menu("MENU","SEND",1,"RECEIVE",2,"QUIT",3)

Lbl 1
Input ("MESSAGE:",Str1)
Disp Str1
Goto R

Lbl 2
GetCalc(Str2)
Disp Str2
Pause
Goto R

Lbl 3
ClrHome
Stop
```

CRAMER.8xp
By Brianna Lawson

This program will solve a square system of linear equations using Cramer's rule. The user will input an invertible square into matrix *A*. They will then call the program which will solve a system of 2, 3, or 4 variables and equations. The user will be asked to input the dimension (number of variables) and then enter the constants, C, D, E, F depending on the dimension.

Suppose you are given the system $\begin{aligned} a_{1,1}w + a_{1,2}x + a_{1,3}y + a_{1,4}z &= C \\ a_{2,1}w + a_{2,2}x + a_{2,3}y + a_{2,4}z &= D \\ a_{3,1}w + a_{3,2}x + a_{3,3}y + a_{3,4}z &= E \\ a_{4,1}w + a_{4,2}x + a_{4,3}y + a_{4,4}z &= F \end{aligned}$

It will then calculate the determinant $\begin{vmatrix} \mathrm{C} & a_{1,2} & a_{1,3} & a_{1,4} \\ D & a_{2,2} & a_{2,3} & a_{2,4} \\ E & a_{3,2} & a_{3,3} & a_{3,4} \\ F & a_{4,2} & a_{4,3} & a_{4,4} \end{vmatrix}$ and store this value as *N*. It repeats this process like such: calculating $\begin{vmatrix} a_{1,1} & \mathrm{C} & a_{1,3} & a_{1,4} \\ a_{2,1} & D & a_{2,3} & a_{2,4} \\ a_{3,1} & \mathrm{E} & a_{3,3} & a_{3,4} \\ a_{4,1} & \mathrm{F} & a_{4,3} & a_{4,4} \end{vmatrix}$ and store this as *O*. And repeat calculating determinants shifting the constant values column through the matrix until, *N*, *O*, *P*, and *Q* are solved. It will then calculate the *w*, *x*, *y*, *z* using $w=\frac{N}{M}$, $x=\frac{O}{M}$, $y=\frac{P}{M}$, $z=\frac{Q}{M}$ and then display the values for *w*, *x*, *y*, *z* which is the intersection of the four lines. It will repeat this similar process for 3 *x* 3 and 2 *x* 2 systems.

File Name: CRAMER.8XP Type: Program Protected: No
Comment:Program file Ø5/22/13, 1Ø:28

```
FnOn
PlotsOff
ClrHome
Disp "***************"
Disp ""
Disp ""
Disp "  CRAMERS RULE  "
Disp ""
Disp ""
Disp "***************"
Pause
ClrHome

Menu("COEFF IN MTRX A","YES",1,"NO",2)

Lbl 1
det([A])→M
Disp "NUM OF CONSTANTS"
Prompt D
Disp "ENTER CONTANTS"

If D=2
Then
Prompt C
Prompt D
det([A])→M
dim([A])→dim([B])
C→[B](1,1)
D→[B](2,1)
[A](1,2)→[B](1,2)
[A](2,2)→[B](2,2)
det([B])→N

C→[A](1,2)
D→[A](2,2)
det([A])→Q

N/M→X
Q/M→Y
ClrHome
Disp "ANSWERS ARE..."
Disp X,Y
Pause
ClrHome
End

If D=3
Then
Prompt C,D,F
[A]→[B]
det([A])→M
C→[B](1,1)
D→[B](2,1)
F→[B](3,1)
det([B])→N
[A]→[C]
C→[C](1,2)
D→[C](2,2)
```

```
F→[C](3,2)
det([C])→P
[A]→[D]
C→[D](1,3)
D→[D](2,3)
F→[D](3,3)
det([D])→Q

N/M→X
P/M→Y
Q/M→Z
Disp "ANSWERS ARE..."
Disp X,Y,Z
Pause
ClrHome
End

If D=4
Then
[A]→[B]
[A]→[C]
[A]→[D]
[A]→[E]
Prompt C,D,E,F
C→[B](1,1)
D→[B](2,1)
E→[B](3,1)
F→[B](4,1)
det([B])→N
det([A])→M
C→[C](1,2)
D→[C](2,2)
E→[C](3,2)
F→[C](4,2)

det([C])→O
C→[D](1,3)
D→[D](2,3)
E→[D](3,3)
F→[D](4,3)
det([D])→P

C→[E](1,4)
D→[E](2,4)
E→[E](3,4)
F→[E](4,4)
det([E])→Q

N/M→W
O/M→X
P/M→Y
Q/M→Z
Disp W,X,Y,Z
Pause
ClrHome
End

Lbl 2
Stop
```