APX ATARI® PROGRAM EXCHANGE

John Crane

RPN CALCULATOR SIMULATOR

A Reverse Polish Notation calculator

Cassette: 16K (APX-10105) Diskette: 24K (APX-20105)

User-Written Software for ATARI Home Computers

John Crane

RPN
CALCULATOR SIMULATOR

A Reverse Polish Notation calculator

Cassette: 16K (APX-10105)   Diskette: 24K (APX-20105)

RPN CALCULATOR SIMULATOR

by

John Crane

Distributed By

The ATARI Program Exchange
P.O. Box 3705
Santa Clara, CA 95055

To request an APX Product Catalog, write to the address above, or call toll-free:

800/538-1862 (outside California)
800/672-1850 (within California)

Or call our Sales number, 408/727-5603

Trademarks of Atari

Printed in U.S.A.

SYSTEM REQUIREMENTS:

Tape version- ATARI 400/800 Computer
16K memory (minimum)
ATARI BASIC Language Cartridge
ATARI 410 Program Recorder

Disk Version- ATARI 400/800 Computer
24K memory (minimum)
ATARI BASIC Language Cartridge
One ATARI 810 Disk Drive (minimum)

SYSTEM OPTIONS:

Both Versions- ATARI 820 Printer (40 columns)
ATARI 822 Printer (80 columns)
ATARI 825 Printer (80 columns)
ATARI 850 Interface Box
Other suitable printer

LOADING THE PROGRAM FROM TAPE:

1. Apply power to the Computer and T.V. (or video monitor)
2. Place the RPN CALCULATOR PROGRAM tape into the Tape Drive.
3. Rewind the tape to the start of either side.
4. Press the PLAY key on the Tape Drive.
5. Turn on your computer while holding down the START key. After a beep, press RETURN, and the program will load and run automatically.

LOADING THE PROGRAM FROM DISKETTE:

1. Apply power to Disk Drive number one only - everything else should be off.
2. After the busy light goes out, place the RPN CALCULATOR PROGRAM disk into the drive.
3. Apply power to the computer console.
4. The disk drive should start to run and, after a short pause, the word READY should appear on the T.V. screen---<u>DO NOT REMOVE THE DISKETTE AT THIS TIME</u>.
5. After a short pause, the disk drive should again start to run and some text should begin to scroll across the screen.
6. After the program name and other text have passed, the disk drive will again start to run and within a few seconds the screen should show the calculator display.

THE REVERSE POLISH NOTATION FORMAT:

It is not within the scope of this document to educate the user about the technical specifics of Reverse Polish Notation. There are many good books which may be purchased to perform this function; however, just becoming familiar with this program and its use will provide a great deal of insight into the operation of this type of system and should provide many hours of enjoyment and education.

THERE ARE 4 AREAS OF INFORMATION ON THE T.V. SCREEN:
(Run the program and locate the 4 areas indicated)

AREA 1. The STACK OF REGISTERS shows the contents of:

- a. T register (top register in the stack)
- b. Z register (next register down)
- c. Y register (next below Z)
- d. X register (bottom of the stack)
- e. LX register (Last contents of the X register)

This area is on the upper left of the screen and is bounded by the heavy, reverse video lines. Each register is identified at its left edge.
NOTE: The word RADIANS or DEGREES within the reverse-video area denotes the current angular measurement mode of the system (the default value is RADIANS).

AREA 2. 20 MEMORY REGISTERS (0 to 19):

This area is just to the right of the STACK and shows the contents of each of the 20 memory registers. If there is an asterisk (*) next to any of the "R" numbers, this indicates that there is a non-zero value in that register.

AREA 3. COMMAND SYNTAX REMINDER AREA:

The area below the STACK and REGISTER areas shows all of the commands recognized by the system. This is present at all times as a reminder should you forget the exact syntax which is to be used with this program.

AREA 4. VALUE/COMMAND INPUT AREA:

This area is where you will enter either a number to be pushed onto the stack or a function (such as "ADD") to be performed on numbers which are already in the stack or the memory registers. <u>All</u> inputs to this program are entered here!

RPN USES POST-FIX NOTATION:

Post-fix notation is a scheme of mathematical operations where the function to be performed follows the values on which it operates. The following is an example of post-fix notation:

3 4 +

In the example above, the values 3 and 4 are added together. If the result were shown, it would, of course, equal 7. Please note that the values 3 and 4 which are to be added together precede the add operator. The term post-fix means that the operator comes after the values to be operated upon. Reverse Polish Notation uses Post-fix Notation.

Most weekend mathematicians are more familiar with ALGEBRAIC entry math. Algebraic entry is an in-fix math scheme where the operator lies between the values which are operated upon. Using algebraic entry, the problem above would be written as follows:

3 + 4 =

This program uses post-fix notation which leads to the concept of a stack.

USING THE RPN STACK:

In this program the stack is represented by the registers called X, Y, Z, and T. We can put numbers into the stack, one at a time, until we have enough (usually 1 or 2) to perform the desired operation.

Let's use the example above. Remembering the lesson on post-fix notation, we can see that we must somehow enter the numbers 3 and 4 before we can operate upon them with the add function. To do this, we "push" these two numbers onto the stack by first entering the 3 into the X register, then the 4. When the 4 is entered into the X register, the 3 which was just there is "pushed" up into the Y register.

Now that we have the stack loaded with two numbers- a 3 in the Y register and a 4 in the X register- we can request that the add function be performed upon these numerical values. When this happens, the calculator takes the 3, adds it to the 4, places the result in the X register, and "drops the stack" (another way of saying that all of the values in the stack move down one register).

WHAT'S IT ALL GOOD FOR?

A clear example of the usefulness of this feature can be seen by examining the following problem:

A PROBLEM SOLVED USING BOTH SYSTEMS:

(3 X 4) + (5 X 6) = 42

ALGEGRAIC OPERATIONS TO SOLVE THIS PROBLEM:

1) Enter the 3.
2) Enter the multiply operation.
3) Enter the 4.
4) Obtain the product by entering the equals operation.
5) Store the interim result (12 in this case) somewhere.
6) Enter the 5.
7) Enter the multiply operation.
8) Enter the 6.
9) Obtain the product by entering the equals operation.
10) Store the interim result (30 in this case) somewhere.
11) Recall the first interim result (12).
12) Enter the add operation.
13) Recall the second result (30).
14) Obtain the final result by entering the equals operation.

RPN OPERATIONS TO SOLVE THIS PROBLEM:

1) Enter the 3.
2) Enter the 4.
3) Enter the multiply operation.
4) Enter the 5.
5) Enter the 6.
6) Enter the multiply operation.
7) Enter the add operation.

As you can see, the stack automatically keeps interim results until they are either used or pushed off the stack by subsequent entries. This means that it is possible to reduce the number of entries required to obtain the results from a series of many operations. Some algebraic-type calculators add extra features to help reduce the number of keystrokes required to perform these operations, but none can reduce the number to that of Reverse Polish Notation.

It can be shown that problems of any complexity may be solved using RPN with a stack which is four registers deep. One merely begins evaluating the innermost terms of the expression working outward and letting the stack hold interim results until needed by the next step in the problem.

GENERAL OVERVIEW OF OPERATION:

One-number functions (SIN, COS, LOG, etc.):

1. Type in the number on which the function will operate.
2. Type in the function which will operate on the number.

Example: (type as shown, [keyname] is a key to be pressed)

100 [return]
LOG [return]

Notice what happened - the 100 was first placed into the X register, then the LOG operation was performed upon that number and the resulting number 2 (the LOG of 100) was written in place of the preceding number 100.

Two-number functions (+, -, *, /, etc.):

1. Type in the first number.
2. Type in the second number.
3. Type in the function which is to operate on the two numbers just entered.

Example: (type as shown, [keyname] is a key to be pressed)

355 [return]
113 [return]
/   [return]

Here is what happened:

- The number 355 was placed into the X register
- When the number 113 was entered, 2 things happened:
  First - the 355 was moved up into the Y register,
  Second- the 113 was placed into the X register.
- When the "/" (divide) command was entered, 2 things happened again:
  First - 355 was divided by 113,
  Second- Both the 355 and the 113 were removed from the stack and the answer (3.1415...) was placed into the X register.

FUNCTIONS:

Calculator functions fall into 4 categories:

1. SYSTEM COMMANDS   2. STACK OPERATIONS
3. MATH FUNCTIONS   4. STATISTICAL FUNCTIONS

The next page describes each of these commands.

SYSTEM COMMANDS:

CLX	Clears the X register to 0.
CLR	Clears all stack registers to 0.
CLST	Clears all stack registers, except LX, to 0.
CLRG	Clears all 20 memory registers to 0.
RTN or ENT	Same as pressing the RETURN KEY.
DEG	Places the calculator into the DEGREES mode. Interprets all numbers as Degrees for Trig use.
RAD	Places the calculator into the RADIANS mode. Interprets all numbers as Radians for Trig use.
PRT	Prints the contents of the STACK, MEMORY REG'S, and REGISTER NAMES on the system printer.
END	Ends program execution - returns to BASIC.

STACK OPERATIONS:

<	Rolls the stack DOWN. T-->Z, Z-->Y, etc.
>	Rolls the stack UP. X-->Y, Y-->Z, etc.
XY or YX	Swaps the contents of the X and Y registers.
XZ or ZX	Swaps the contents of the X and Z registers.
XT or TX	Swaps the contents of the X and T registers.
YZ or ZY	Swaps the contents of the Y and Z registers.
YT or TY	Swaps the contents of the Y and T registers.
ZT or TZ	Swaps the contents of the Z and T registers.
LX	Places the last value of X into the X register.

MATH FUNCTIONS:

+	ADD the contents of the X and Y register.
-	SUBTRACT the contents of the X from Y register.
*	MULTIPLY the contents of the X and Y registers.
/	DIVIDE the contents of Y register by X register.
INV	Replaces contents of X register with the reciprocal of the contents of X (1/X).
CHS	Changes the sign of the number in the X register.
PI	Places the value of PI into the X register.
SIN	Calculates the SINE of the X register value.
COS	Calculates the COSINE of the X register value.
TAN	Calculates the TANGENT of the X register Value.
ASIN	Finds the ARC SINE of the X register value.
ACOS	Finds the ARC COSINE of the X register value.
ATAN	Finds the ARC TANGENT of the X register value.
SQR	Calculates the SQUARE ROOT of the X register.
Y^X or ^	Raises the value in the Y reg. to the X power.
LOG	Finds the LOG (base 10) of the X register.
LN	Finds the LOG (natural base) of the X register.
ALOG	Finds the ANTILOG (base 10) of the X register.
STOrr	Places the value of the X STACK REGISTER into MEMORY REGISTER rr.
RCLrr	Places the value of teh MEMORY REGISTER rr into the X STACK REGISTER, pushing the stack UP.
NOTE:	REGISTER VALUE (rr) MUST ALWAYS BE 2 DIGITS.

COMMANDS (continued)

STATISTICAL CALCULATIONS:

In addition to all of the above functions, the calculator has been designed to do some standard statistical functions. When the command is given to do these functions, the program makes special use of some of the MEMORY REGISTERS; specifically, registers 10 to 19, so users should avoid using these registers for data storage when doing statistical functions.

THE STATISTICAL REGISTERS:

Register 10	SUM OF ALL X values.
Register 11	SUM OF ALL X^2 values.
Register 12	SUM OF ALL Y values.
Register 13	SUM OF ALL Y^2 values.
Register 14	SUM OF ALL X times Y values.
Register 15	TOTAL NUMBER OF DATA POINTS (N).
Register 16	ARITHMETIC MEAN OF ALL X values.
Register 17	ARITHMETIC MEAN OF ALL Y values.
Register 18	STANDARD DEVIATION OF ALL X values.
Register 19	STANDARD DEVIATION OF ALL Y values.

GETTING READY TO USE THE STATISTICAL FUNCTIONS:

1. If you do not have important data in any of the STORAGE REGISTERS then execute these commands:

```
CLRG
CLST
```

2. If you do have important data in registers 10 to 19 then move it into registers 0 through 9 and execute:

```
CLST
STO10
.
.
.
STO19
```

3. Begin entering data (either ONE-NUMBER or TWO-NUMBER):

ONE-NUMBER DATA- Place number into X register.

TWO-NUMBER DATA- Place Y number into X register first. Place X number into X register, thus pushing previously entered number up into the Y register where it belongs.

COMMANDS (continued)

GETTING READY TO USE THE STATISTICAL FUNCTIONS (continued):

4. Type the command SUM+ to insert this data into the system. Notice that asterisks (*) will appear as results are added to the appropriate registers.

5. If BAD DATA gets inadvertently added to the system, simply re-enter the same data (the mistake data), and type the command SUM- to remove it and all of its effects.

6. Repeat steps 3-5 above until all data has been added.

7. When you are ready to calculate the ARITHMETIC MEAN or STANDARD DEVIATION on the data entered, enter the following commands:

   To calculate ARITHMETIC MEAN - type MEAN [return]
   To calculate STANDARD DEVIATION - type MEAN [return]
   type SD [return]

   If you were working with ONE-NUMBER DATA then the answer will appear in the X register.
   If you wereworking with TWO-NUMBER DATA then the results for the X-data will be in the X register, and the results for the Y-data will be in the Y register.

NOTE: While entering data using SUM+ the data point entered will be returned to you in the LX register while the data point number (N) shows in the X register. This feature is provided to help keep you from losing your place while entering long lists of data.

ONE FINAL WORD:

There is no substitute for experience. Work with the program as much as you can, and it will become almost second nature to you. I have taken great pains to correct some of the more obvious weaknesses in the ATARI BASIC language handling of math functions. For example, if you try to find the sine of 30 degrees using ATARI BASIC statements, it will tell you .49999999 rather than the exact 0.5 which is expected.

In order to do this, I had to sacrifice one or two digits of accuracy in the normal display. If you plan to use the calculator for dollars-and-cents calculations, totals exceeding 9,999,999.99 can be expected to show some rounding errors. You may also notice a reduction in the accuracy of very small numbers. For example, the calculator thinks that pi and 355/113 are exactly the same value even though there is a difference of approximately 4 millionths of one percent.

Limited Warranty on Media and Hardware Accessories. We, Atari, Inc., guarantee to you, the original retail purchaser, that the medium on which the APX program is recorded and any hardware accessories sold by APX are free from defects for thirty days from the date of purchase. Any applicable implied warranties, including warranties of merchantability and fitness for a particular purpose, are also limited to thirty days from the date of purchase. Some states don't allow limitations on a warranty's period, so this limitation might not apply to you. If you discover such a defect within the thirty-day period, call APX for a Return Authorization Number, and then return the product along with proof of purchase date to APX. We will repair or replace the product at our option.

You void this warranty if the APX product: (1) has been misused or shows signs of excessive wear; (2) has been damaged by use with non-ATARI Home Computer products; or (3) has been serviced or modified by anyone other than an Authorized ATARI Computer Service Center. Incidental and consequential damages are not covered by this warranty or by any implied warranty. Some states don't allow exclusion of incidental or consequential damages, so this exclusion might not apply to you.

Disclaimer of Warranty and Liability on Computer Programs. Most APX programs have been written by people not employed by Atari, Inc. The programs we select for APX offer something of value that we want to make available to ATARI Home Computer owners. To offer these programs to the widest number of people economically, we don't put APX products through rigorous testing. Therefore, APX products are sold "as is," and we do not guarantee them in any way. In particular, we make no warranty, express or implied, including warranties of merchantability and fitness for a particular purpose. We are not liable for any losses or damages of any kind that result from use of an APX product.

ATARI®
PROGRAM
EXCHANGE
P.O. Box 3705
Santa Clara, CA 95055

Review Form

We're interested in your experiences with APX programs and documentation, both favorable and unfavorable. Many of our authors are eager to improve their programs if they know what you want. And, of course, we want to know about any bugs that slipped by us, so that the author can fix them. We also want to know whether our instructions are meeting your needs. You are our best source for suggesting improvements! Please help us by taking a moment to fill in this review sheet. Fold the sheet in thirds and seal it so that the address on the bottom of the back becomes the envelope front. Thank you for helping us!

1. Name and APX number of program.

____

____

2. If you have problems using the program, please describe them here.

____

____

____

3. What do you especially like about this program?

____

____

____

4. What do you think the program's weaknesses are?

____

____

____

5. How can the catalog description be more accurate or comprehensive?

____

____

6. On a scale of 1 to 10, 1 being "poor" and 10 being "excellent", please rate the following aspects of this program:

_______ Easy to use
_______ User-oriented (e.g., menus, prompts, clear language)
_______ Enjoyable
_______ Self-instructive
_______ Useful (non-game programs)
_______ Imaginative graphics and sound

7. Describe any technical errors you found in the user instructions (please give page numbers).

8. What did you especially like about the user instructions?

9. What revisions or additions would improve these instructions?

10. On a scale of 1 to 10, 1 representing "poor" and 10 representing "excellent", how would you rate the user instructions and why?

11. Other comments about the program or user instructions:

From

STAMP

ATARI Program Exchange
P.O. Box 3705
Santa Clara, CA 95055

(seal here)