

The HP-71B “Math Machine” 25 Years Old

Joseph K. Horn, Richard J. Nelson, & Dale Thorn



The HP-71B, shown full size with hand pulled card reader module, was introduced February 1, 1984.

Executive Summary

Joseph, Richard, and Dale provide a historical review of this Gen3 HP “calculator.”

Richard provides the general HP-71B features compared to the HP-75C/D which is also classified as a Gen3 machine – Gen2 is the HP-41C.

Joseph explains the exceptional Algebraic capability of the HP-71B Calc Mode.

Dale Thorn describes the powerful HP Basic string commands which utilize the HP-71B’s extensive memory (more than an IBM PC of its day) to implement large and powerful sorted Data Bases.

The HP-71B “Math Machine” – 25 Years Old

Joseph K. Horn, Richard J. Nelson, & Dale Thorn

Introduction

Often high technology products are divided into “generations” based on major advances that separate a time series of products into generational groups. HP high-end programmable calculators may be grouped as shown in table 1 below.

Table 1 – Four Generations of HP High End Programmable Calculators

- **January 1972** - HP-35A made advanced calculations possible for everyone.
- **January 1974, Gen1** – HP-65A added programmability, program sharing, and problem solving equalizing.
- **July 1979, Gen2** – HP-41 added alphanumeric, expandability, universal interfacing.
- **August 1982, Gen3** – HP-75/71 added enhanced numerical accuracy, a computer language, and an improved math user interface (71).
- **June 1986, Gen4** – HP 28 ⇒HP 50 added symbolic math, more memory, more speed, & I/O at less cost.

The HP-41, Gen2, was reviewed in *HP Solve* Volume 16 to celebrate the 30th anniversary of this exceptional and very popular machine. The HP-41 was a hand held vertical format machine.

The next generation machines, HP-75C/D & HP-71B, are desk top horizontal format machines that were highly influenced by the rapid microelectronics technology development and personal computer influences of the early 1980’s. Essentially these are miniature desktop computers. Figures one and two show how the two machines compare in relative size and Table 1 compares their basic specifications.



Fig. 1 – HP-75C introduced on August 23, 1982.





Fig. 2 - HP-71B introduced February 1, 1984.

The HP-75C/D, code named Kangaroo, and the HP-71B, code named Titan, were developed concurrently by two “competing” teams of HP engineers.

Kangaroo was based on the HP-85 desktop computer and its development was suggested and promoted by HP Labs (corporate) and not the calculator division. Titan was inspired by the calculator division as the next generation. Titan was designed using a predictive technology approach in terms of anticipating the availability of larger and lower cost memory, and denser integrated circuit technology. It also leveraged

the technology developed from Kangaroo. The two machines appear to be similar, but they are not because Kangaroo was not designed to be a calculator, but to be more of a platform for program development as a smaller version of the HP-85. Titan is a true Gen3 “Math Machine.”

Table 1 – Basic Specifications of the Gen3 HP-75C and HP-71B

Comparison	HP-75C 	HP-71B 
Product category	Handheld desk computer	BASIC language handheld computer
Initial cost	\$ 995	\$ 525
Weight with batteries	26 oz.	12 oz., 54% lighter.
Width	10”	7-1/2”, 25% narrower.
Height	5”	3-7/8”, 23% shorter.
Thickness	1.25”	1”, 20% thinner.
Number of keys	65	55, 15% fewer keys.
Total surface area	10” x 5” = 50 in ²	7.5” x 3.875” = 29 in ² , 42% smaller
Keyboard area	9.37” x 2.42” = 22.7 in ²	6.90” x 2.29” = 15.8 in ² , 30% smaller
Card Reader, hand pulled	Built in	Accessory
HP-IL Interface	Built in, single controller	Accessory, dual ¹ , may also be device
Battery	Custom NiCad Pack, (HP-35A). 1.5 Ah	Four AAA cells, 1.25 Ah.
Display (both LCD)	1 line, 32 characters, 5 x 8 dots	1 line, 22 characters, 5 x 8 dots
Display annunciators	4	14, 250% more annunciators.
Expansion ports	3 general, 1 RAM (batt. compartment)	4 general, 33% more ports.
Programming	BASIC, 194 key words	BASIC, 240 key words, 24% more.
Language Extension Files	YES	YES
RAM	16 K Bytes, exp. to 24 K	17.5 K Bytes, exp to 33.5 K ²
System ROM	48 K Bytes	64 K Bytes, 33% more.
Time keeping (crystal)	Real time clock & calendar	Real time clock & calendar
IC technology	Custom CMOS	Custom CMOS
Numeric entry	Embedded keypad	Separate keypad
System documentation, IDS	NO	A First; Three extensive volumes

1. The single HP-IL module was the 82401A, the dual HP-IL module was the 82402A.

2. Expandable to over 400 K bytes using third party modules (more than an IBM PC of the time).

The Technical Achievements of the HP-71B

Richard J. Nelson

Each team thought they had the “best” product, but Kangaroo was further ahead. I was asked to assemble a user community alpha-test group to visit Corvallis and test the Kangaroo system. Four active and dedicated HP users met in Corvallis and spent three days in a hotel room with every piece of working hardware brought to us from the factory to study and evaluate. I had no idea of Titan at that time, but I knew that something was “different” with this machine other than the physically obvious.

The HP-75C/D was severely memory limited. The HP-71B was not as limited, especially when third party suppliers started offering additional memory. Here is a brief description of the achievements made by the HP-71B. HP promoted the HP-71B as a “BASIC language handheld computer.”

Hardware:

- **Six ports.** The HP-71B has four ports in the front for memory modules. These could be 16K, 32K, 48K, and 64K bytes of custom ROM modules or for up to four 4 K bytes of RAM modules. There are two ports in the back. One is for the card reader where the rectangular area of the upper right portion

of the machine pops out and is replaced by a card reader module. On the other end of the machine is a port for the HP-IL module.

- **Hand pulled card reader** that uses 10 inch long (3/8" wide) magnetic cards.
- An **AC connector** is provided for an AC adapter to plug into the connector located between the HP-IL and card reader ports.
- A **Keyboard overlay** is accommodated to label customized key assignments. Unlike the HP-41 keyboard the HP-71B keyboard only allows a single overlay to be attached at a time.
- **Quartz crystal time of day (timer) clock.** The built-in date/time functions were more limited than those of the HP-41 or HP-75.
- **HP-IL Module**, HP 82401A. Unlike the HP-75 or HP-41, the HP-71B could also act as a device in an HP-IL loop controlled by another device, allowing several of them in the same loop, communicating with each other (or an HP-75 or HP-41) or sharing peripherals. Using an HP 82402A Dual HP-IL controller, it was even possible to connect one HP-71B to two HP-IL loops simultaneously. One could be a controller in one loop and another could be a device in another loop.
- **Extension modules** which could be seamlessly integrated with the operating system. The Math module was outstanding and exceptional in its capability. Among the features added by the Math module were natural handling of complex numbers, and a recursively callable numerical solver. HP-71B modules include AMPI Statistics, Circuit Analysis, Curve Fitting, Data Acquisition, Data Communications, Finance, FORTH/Assembler, HP-41/HP-71 Translator, Mathematics, Surveying, and Text Editor.
- **LEX Files** are language Extension files that help make the HP-71B a handheld computer with maximum potential for expansion and customization. The July 1984 HP Journal article describes the power of LEX files.

LEX files do this in two ways. First, they provide a means of adding BASIC keywords to the operating system. Up to 256 keywords can be added per LEX file, in addition to the keywords already available in the base operating system and other LEX files. No capability is sacrificed when adding keywords to the machine — the operating system is just further enhanced. Second, LEX files can contain machine language routines called poll handlers that have an opportunity to respond when a particular piece of operating system code is executed. For example, poll handlers can be used to take over control of the display and keyboard, add file types to the system, and substitute local language messages for the existing English ones, making it easy to adapt software on the HP-71B to other languages. The challenge to the design team was to provide the hooks, known as polls, in all the key parts of the operating system so that the HP-71B could be customized for virtually any application.

Software:

- **BASIC programming language** with over 240 keywords.
- **IEEE 754-1985 Standard** for Floating Point Arithmetic was implemented on the JHP-71B before the standard was formally adopted. The standard defines arithmetic formats (including signed zeros, sub-normal numbers, infinities, and special not a number values, NaNs); interchange formats; rounding algorithms; math operations; and exception handling.

The hardware and software achievements of the HP-71B are more than sufficient to make it a new generation. The most important calculator achievement, however, is the mathematics problem solving

user interface called CALC Mode. Here is how a December 1983 HP-71B sales brochure (5953-5575D) describes CALC Mode.

“Get fast, easy solutions to your calculation problems with no extra effort on your part. CALC mode is easy:

- *Quick and simple to learn.*
- *Key in the expression from left to right just as you read it.*
- *See your intermediate results develop as you proceed.*
- *Use the powerful set of editing tools to correct errors or experiment with different values.*

Use built-in mathematical functions, and your own functions for fast easy solutions. CALC Mode puts these features to work on your calculations:

- *Immediate evaluation of expressions*
- *Automatic parenthesis matching.*
- *Variables share the same value whether assigned in BASIC or CALC Mode.*
- *Result of a previous calculation can be recalled and modified for use in a current equation.”*

Next Joseph Horn describes the HP-71B CALC Mode and lastly Dale Thorn describes how he used the HP-71B file structure and BASIC for very fast sorted data base applications.

About the Authors

Joseph K Horn has been using HP calculators since September 1976 as a math teacher and author of many articles related to HP calculators. He is webmaster of the HHC websites and is especially known for his collections of HP 48 programs made available on Goodies Discs. Joseph has written an HP-71 book titled, *HP-71 BASIC Made Easy*, which provided a collection of programs and very useful facts and tips about the HP-71



Richard J. Nelson has been using HP calculators since July 1972 as an electronics engineer and technical writer. He has organized and sponsored national and international HP user community meetings and conferences since 1974. Richard proposed and managed, with a formal committee of five (and hundreds of other contributors) the HP-41 PPC ROM project. He has edited and published four newsletters dedicated to HP calculators and published hundreds of HP calculator related articles in various US and foreign technical publications



Dale Thorn has been using HP calculators, palmtops, and desktops since 1975 as a professional programmer specializing in data base applications. Dale was responsible for the "String Index Database" application that ran on five platforms, including the HP-71, HP-85/86, HP-9000 series BASIC O/S, and the HP-9000 HPUX series.



References:

The best technical reference for the HP-75C/D and the HP-71B machines are the two Hewlett-Packard Journals dedicated to these machines. See the links below.

HP-75C/D: *Hewlett-Packard Journal* June 1983.

<http://www.hpl.hp.com/hpjjournal/pdfs/IssuePDFs/1983-06.pdf>

HP-71B: *Hewlett-Packard Journal* July 1984

<http://www.hpl.hp.com/hpjjournal/pdfs/IssuePDFs/1984-07.pdf>

RPN + AOS = HP-71B CALC Mode

Joseph K. Horn

RPN and AOS (TI's term for Algebraic Operating System) each have their own strengths and weaknesses.

	Intermediate Results	Algebraic Entry
RPN	•	
AOS		•
HP-71 CALC Mode	•	•

One benefit of RPN is that it lets you see all the **intermediate results** as you progress through the calculation. This gives the user confidence in the final answer. One drawback is that RPN does not let the user key algebraic expressions the way they are printed in books, written by hand, and taught in school. This requires the user to learn a new entry system that is not taught in school, and can cause a lack of confidence during the calculation process.

One benefit of AOS is that it does let you type algebraic expressions the way they are printed, written, and taught. This gives the user confidence during the calculation process. One drawback is that AOS waits until the whole expression is typed before calculating anything, so **no intermediate results** are shown. This can cause a lack of confidence in the final answer.

It has always been assumed that these strengths and weaknesses are inherent to the two systems, and that there is no way to get the benefits of both while eliminating the drawbacks of both. However, one calculator managed to do this. Unfortunately its solution to this problem was forgotten when it went out of production.

The calculator which solved the RPN vs AOS problem was the HP-71B. Its "CALC Mode" has the best of both worlds as demonstrated in the examples below. It also avoids the drawbacks of both systems. It's a brilliant design that HP should revive before somebody else does. I would in fact recommend that *all* future HP algebraic calculators implement this method of expression entry.

The idea is simple: The user types algebraic expressions the way they are printed and taught, but the calculator automatically simplifies all subexpressions *while they are being typed*, and algebraically *replaces* those subexpressions with their intermediate result, right in the display.

EXAMPLE #1

Printed problem:

$$1 + 3 + 5 + 7$$

RPN:

Keystrokes	Display
1 ENTER 3 +	4
5 +	9
7 +	16

Upside: We see the **intermediate results** (“subtotals”).
 Downside: It’s keyed differently than the way it’s printed.

AOS:

Keystrokes	Display
1+3+5+7=	16

Upside: It’s entered the way it’s printed.
 Downside: No intermediate answers.

HP-71 CALC Mode:

Keystrokes	Display
1+3+	4
5+	9
7=	16

Upside: It’s entered the way it’s printed.
 Upside: All **intermediate results** are shown.
 Pressing up-arrow displays 1+3+5+7 for review or editing.

EXAMPLE #2

Printed problem:

$$1 + \frac{2}{3 + \frac{4}{5}}$$

RPN:

Keystrokes	Display
2 ENTER 4 ENTER 5 ÷	0.8
3 +	3.8
÷	0.5263
1 +	1.5263

Intermediate results are seen.

AOS:

Keystrokes	Display
1+2÷(3+4÷5) =	1.5263

Intermediate results are not seen.

HP-71 CALC Mode:

Keystrokes	Display
1+2/(3+4/5)	1+2/(3+4/5)
RUN	1+2/(3+0.8)
)	1+2/(3.8)
RUN	1+.5263
=	1.5263

It's entered the way it's printed.

The RUN key is optional; it does step-by-step simplification, perfect for students.
The **intermediate results** AND the entire simplification process are clearly displayed.
Pressing up-arrow displays the entire expression for review or editing.

EXAMPLE #3

Printed problem, from HP 48G User's Guide, pg. 3-3:

$$23^2 - (13 \times 9) + \frac{5}{7}$$

RPN:

Keystrokes	Display
$23 x^2$	529
13 ENTER 9 *	117
-	412
5 ENTER 7 ÷	.7143
+	412.7143

Upside: We see the **intermediate results**.

Downside: It's keyed differently than the way it's printed.

AOS:

Keystrokes	Display
$23^2-13*9+5/7 =$	412.7143

Intermediate results are not seen.

HP-71 CALC Mode:

Keystrokes	Display
23^2-	529-
$13*9$ RUN	529-117
+	412+
$5/7$ RUN	412+.7143
=	412.7143

It's entered the way it's printed.

The RUN key is optional; it does step-by-step simplification, perfect for students.
The intermediate results AND the entire simplification process are clearly displayed.
Pressing up-arrow displays the entire expression for review or editing.

EXAMPLE #4

Printed problem, from HP 39G/40G User's Guide, pg. 1-18:

$$\frac{23^2 - 14\sqrt{8}}{-3} \ln(45)$$

RPN:

Keystrokes	Display
23 x ²	529
8 √x	2.8284
14 *	39.5980
-	489.4020
3 CHS ÷	-163.1340
45 LN	3.8067
*	-620.9961

Upside: We see the **intermediate results**.

Downside: It's keyed differently than the way it's printed.

AOS:

Keystrokes	Display
((23^2-14*√8)÷-3)*LN(45) =	-620.9961

Intermediate results are not seen.

HP-71 CALC Mode:

Keystrokes	Display
((23^2-	((529-
14*SQR(8) RUN	((529-14*2.8284)
RUN	((529-39.5980)
)	(489.4020)
/-3*	-163.1340
LN(45) RUN	-163.1340*3.8067
=	-620.9961

It's entered the way it's printed.

The RUN key is optional; it does step-by-step simplification, perfect for students.

The **intermediate results** AND the entire simplification process are clearly displayed.

Pressing up-arrow displays the entire expression for review or editing.

EXAMPLE #5

Printed problem, from HP-97 Owner's Handbook, pg. 33:

$$\sqrt{(2+3) \times (4+5)} + \sqrt{(6+7) \times (8+9)}$$

RPN:

Keystrokes	Display
2 ENTER 3 +	5
4 ENTER 5 +	9
*	45
√x	6.7082
6 ENTER 7 +	13

8 ENTER 9 +	17
*	221
\sqrt{x}	14.8661
+	21.5743

Upside: We see the **intermediate answers**.

Downside: It's keyed differently than the way it's printed.

AOS:

Keystrokes	Display
$\sqrt{(2+3)*(4+5)} + \sqrt{(6+7)*(8+9)}$	21.5743

Intermediate results are not seen.

HP-71 CALC Mode:

Keystrokes	Display
SQR((2+3)	SQR((5))
* (4+5)	SQR(5*(9))
)	SQR(45)
+	6.7082
SQR((6+7)	6.7082+SQR((13))
* (8+9)	6.7082+SQR(13*(17))
)	6.7082+SQR(221)
RUN	6.7082+ 14.8661
=	21.5743

It's entered the way it's printed.

The RUN key is optional; it does step-by-step simplification, perfect for students.

The **intermediate results** AND the entire simplification process are clearly displayed.

Pressing up-arrow displays the entire expression for review or editing.

HP-71 CALC Mode vs the HP 49/50 EquationWriter










Some may object that the HP 49/50 EquationWriter (EQW) can do what HP-71 CALC Mode does. This is false, for two reasons.

Reason #1: EQW requires many specialized keystrokes other than the algebraic expression. These extra keystrokes are so non-intuitive that nobody uses EQW for step-by-step algebraic simplification

Reason #2: EQW does not allow the user to view the entire algebraic expression after performing the calculation. If the entire expression is keyed in first, then the work required to evaluate it while viewing the intermediate results becomes even greater.

Here are the keystrokes required by EQW to evaluate Example #5 above. HP-71 CALC Mode requires none of the arrow keys or SIMP keys (shaded below) that EQW requires, as can be seen in the following step-by-step use of EQW to evaluate the expression.

HP-71 CALC Mode compared with EQW:

Extra Keys	Keystrokes	Display
	$\sqrt{(2+3)}$	$\sqrt{(2+3)}$
1	▶	$\sqrt{(2+3)}$
2		$\sqrt{(5)}$
3	▶	$\sqrt{5}$
	$* (4+5)$	$\sqrt{(5) \cdot (4+5)}$
4	▶	$\sqrt{(5) \cdot (4+5)}$
5		$\sqrt{(5) \cdot (9)}$
6	▶	$\sqrt{(5) \cdot (9)}$
7	▶	$\sqrt{5 \cdot 9}$
8		$\sqrt{45}$
9	▶	$\sqrt{45}$
	$+ \sqrt{(6+7)}$	$\sqrt{45} + \sqrt{(6+7)}$
10	▶	$\sqrt{45} + \sqrt{(6+7)}$
11		$\sqrt{45} + \sqrt{(13)}$
12	▶	$\sqrt{45} + \sqrt{(13)}$
	$* (8+9)$	$\sqrt{45} + \sqrt{(13) \cdot (8+9)}$
13	▶	$\sqrt{45} + \sqrt{(13) \cdot (8+9)}$
14		$\sqrt{45} + \sqrt{(13) \cdot (17)}$
15	▶	$\sqrt{45} + \sqrt{(13) \cdot (17)}$
16	▶	$\sqrt{45} + \sqrt{(13) \cdot (17)}$
17		$\sqrt{45} + \sqrt{221}$
18	▶	$\sqrt{45} + \sqrt{221}$
19		$\sqrt{45} + 14.8660687473$
20	◀	$\sqrt{45} + 14.8660687473$
21		$6.7082039325 + 14.866068...$
22	▲	$6.7082039325 + 14.866068...$
23		21.5742726798

As can be seen, EQW requires 23 more keystrokes than RPN, AOS, and HP-71 CALC Mode. Even worse, those keystrokes are not intuitive. Worst of all, there is no way to see the whole algebraic expression after it is evaluated. Thus EQW is far worse than RPN, AOS, and HP-71 CALC Mode.

HP-71 String Handling

Dale Thorn

My first experience with HP's own BASIC language, and hence the inspirations for my future work with the HP-71, occurred with the HP-85 portable/desktop computer in 1980. That computer came with 32 kb of RAM, and lacked the ability to employ string arrays with the default built-in language.

Since HP BASIC allowed strings to be any length up to 32 kb, and because I needed to build pre-sorted lists of short integer numbers, I decided to code each integer as a two-byte string and embed those substrings dynamically within a large single string that could be written to external backup media very quickly.

My first use of the HP-71, given the limited 16 kb of RAM with a maximum of four 4-kb plug-in modules, was to build the same large strings as I did with the HP-85, since a 30,000-byte string could contain 15,000 short integer values without any of the extra memory space or overhead required for a large numeric array.

Visualize an external data file, with up to 15,000 records written randomly (not sorted), where each record's position is represented by a two-byte integer substring within a single large string in the HP-71's RAM. If those two-byte substrings were arranged just right, those external records could be read in a sorted (i.e. indexed) sequence, even when the two-byte index value represents hundreds of sorted characters in a typical computer index.

Example Data and Index

Record No.	Data	Index
1	Jerry	3
2	Alicia	2
3	Alice	6
4	Kenneth	1
5	Siobhan	4
6	Barry	5

So how is this HP-71 string magic accomplished? Quickly or slowly, depending on the speed of access to the external device or file, and the data operation being attempted. When the operation was to add a new indexed record in real time (and since my aim was to preclude the use of tree-type indexes with their extra space requirements), I would do a binary search of the HP-71's two-byte substrings, and for each search iteration look up the external record from the internal two-byte index value, comparing the current record value to the prior search value, as with any typical binary search.

Then, when the proper insertion point was found within the HP-71's large string in RAM (between two of the 2-byte substrings), the portion of the string from the insertion point to the end of the string would be shifted down by two bytes, allowing the new index value to be popped into place instantly. This two-byte shift was a single command with HP-71 BASIC. In the middle 1980's, with the HP-71 and its typically associated peripherals, this type of data management might have seemed impractical for one reason or another, but the results spoke for themselves - minimum allocation of memory for dynamic indexing of large amounts of data.

One caveat: If some of the RAM were used for many other objects such as small files or variables, and the large string were located before those other objects in RAM, then dynamically adjusting the length of the

large string might be too time-consuming due to the HP-71's automatic repositioning of memory objects when an object at the beginning or in the middle of RAM is resized. My experience with the HP-71 taught me to move the most dynamic objects to the end of the memory list, however that might be accomplished.

When the 32 kb single-chip RAM modules were released between late 1985 and early 1986, I had an HP-71 outfitted with 393,000-plus bytes of RAM, all in one contiguous space, by embedding 256 kb of RAM on the motherboard and adding four of the 32 kb modules to the four front ports of the computer. This extra memory, coupled with the HP-71's maximum string size and very efficient string commands, gave me direct real-time control over numerous external datasets at the same time - a capability that would not be matched by any other pocket-sized computation device until the mid-1990's when multi-megabyte flash memory cards were introduced.

I was sufficiently impressed with what HP (or the actual authors) did in implementing Visicalc on the HP-75, that I did something similar for the HP-71 database (actually a one-file manager for most of the features).

I wrote the program so you could scroll up and down one record, 5 (or a user-defined number) records, or to the beginning and end of the dataset, besides the usual search features which were many. You could also scroll left and right one character or one field, or go to the beginning or end of the record immediately.

With 393,000-plus bytes of possible HP-71B RAM of course, performance could be improved tremendously with the additional caching. With standard memory and an HP-IL floppy disk attached, a specific record of ~10,000 records on the floppy could be found and loaded within about 5 to 7 seconds, given the necessary binary search with my string indexes, and the HP-71's seek-and-load time to the floppy drive.

When printing a list using the index, data could be fed to a LaserJet printer nearly as fast as the printer could output the text. LaserJet speeds were stated in the mid-1980's as a few hundred characters per second, CPS, however, since the LaserJet is a whole-page printer, if you compressed the text to, say, 15,000 characters per page, then the limiting factor was the parallel or serial interface. I used the HP-71 with an HP-IL to RS-232 box only, so I was able to print at about 1,000 CPS, where the RS-232 baud rate was stated as 9,600 (I hope I remember that correctly).

The HP-71's HP-IL transfer rate ran as high as 6,000 CPS in some applications, and I don't recall the actual transfer rate to and from the floppy disk copying a file in one command, but it was probably not much less than the 6,000 CPS maximum.

It wasn't until around 1990 that I began making utility programs for QA testing of DOS software for a database vendor, and despite the HP-71's exceptional string memory capabilities, the nominal 0.6 MHz processor and 4-bit bus made PC-style text processing impractical from the point of view I was working from at that time.

Looking back on the HP-71 circa 1990, it would have made an interesting platform for further development with special techniques, and if I were doing this over again, I would find a marketing guru who could provide the kinds of niche requirements and user stories I would need to create sellable utility programs from.