GET IT COVERED

This is a simple little application of the much touted area of a triangle algorithm; it is of use to a friend of mine who would use it to work out areas of irregular surfaces for tarmacking. This program is in BASIC and written for the '71B' (now a very affordable unit from 'Tuscan Consultants') but should easily translate to other dialects.
The program starts by presenting the title 'AREA BY TRIANGLES', (no, tell that wag that the author is not called triangles) when 'END LINE' is pressed the program continues by zeroing the total 'T', before prompting for input of the three sides 'A', 'B' and 'C'. Line 70 partially evaluates the area but stops short of evaluation of the square root; as, if the length of the sides entered do not allow themselves to join, or one or all sides are zero then there has been an input error; we don't really want the square root of a minus number do we. Should you have been unfortunate enough to have qualified for rejection on the last count then line 80 will see you right, and tell you that your 'FAULTY INPUT HAS BEEN IGNORED', before seeing you safely back to line 40 for fresh input. When you make it to line 90 a mini menu will be presented to you offering you four options which can be selected by pressing the capital letter of your choice (flag -15 was set on line 90 to ensure input is in upper case).
The option are as follows:
'Use' will accept the current input and proceed to prompt for further input.
'Lose' will reject the current input and proceed to prompt for more,
'New' will clear the area total; in affect it will start you from scratch.
'Results+' will display the total area including the last input (hence the +), and then end the program.
It is important to remember that the results will only be displayed as accurately as you have them set; if in doubt execute 'STD'.
As a small test the four triangles with sides (4,5,6),(7,2,8),(10,11.2,9),(7,8,5) should yield 76.4819343874 as a total.

Have Fun Clifford Wylie M.No. 596

```
10 DISP "AREA BY TRIANGLES" @ DELAY .5
20 IF KEY$="" THEN GOTO 20
30 T=0
40 INPUT "A=";A
50 INPUT "B=";B
60 INPUT "C=";C
70 S=(A+B+C)/2 @ E=S*(S-A)*(S-B)*(S-C)
80 IF E<=0 THEN DISP "FAULTY INPUT IGNORED" @ WAIT 1 @ GOTO 40
90 DISP "Use Lose New Result+" @ CFLAG -15
100 F$=KEY$
110 IF F$="" THEN GOTO 100
120 IF F$="U" THEN T=T+SQR(E) @ GOTO 40
130 IF F$="L" THEN GOTO 40
140 IF F$="N" THEN GOTO 30
150 IF F$="R" THEN DISP "AREA=";T+SQR(E) @ END
160 GOTO 100
```

THE HP94 STORY - AUDAX CORPORATION

Craig A. Finseth (member #745)

(This is the last of a series of four articles.)

October 1985

I joined the staff of Steinmetz & Brown, Ltd. This is a small, St. Paul, Minnesota consulting company that had done a number of HP-related products. Readers may well be familiar with their 5 1/4" HP-IL disk drive. My first task was to work with National Car Rental on a bar code project. The customer wished to attach bar code to all of their vehicles and use HP-75D's with wands to track the vehicles as they were hired and returned.

The application was quickly designed and coded. It worked rather well, having only three "commands." One started or continued a collection run, another uploaded the collected data, and the third cleared the data from memory. (Uploading was by means of an HP-IL serial interface and a very minimal implementation of XMODEM (send only) in HP-75 Basic. It was not pretty, but it worked.)

November 1985

Testing continued, and our sample bar code for attaching to the vehicles arrived. We ran into the first glitch: the wand was not reliably reading the bar code.

Winter also continued to arrive, and we ran into our other glitch: the HP-75 was not terribly happy operating with outside temperatures averaging -12 deg C during the day. This problem was easy to correct. I went to Radio Shack, purchased a holder for 4 AA batteries (this was back when they sold actual electronics supplies (:-)), cannibalised an HP battery pack, and attached the two with about one meter of wire. I now had a battery holder that could stay in my pocket -- inside my jacket -- while the HP-75 and wand were outside. The LCD display didn't show anything, but no one looked at the display when scanning bar code anyway, and the beeper worked fine. A more polished version would be required for final customer use, but at least we solved that problem.

December 1985

We were still having problems reading bar code. The customer had specified -- for overriding aesthetic reasons -- that the bar code should be mounted on the inside of the windows in two places: the lower left corner of the windscreen and the left rear side window.

Reading bar code _on_ glass was a solved problem, reading bar code _through_ glass was not. To make it worse, the two places had very different thicknesses and curvatures of glass. HP took this problem seriously, and several HP engineers and I met at an HP plant near Santa Clara, California. We reviewed the physics of reading bar code and came up with two ways to address the problem:

1) Switch to lower-density bar code. (We were using a fairly high-density code.) The customer wanted to keep the bar code discreet, so this option was not taken. (I later learned that a sizeable

amount of the car-hire business was to local people who wanted to hire a car to impress a girl/boy friend or business client. For these people, it is very important that there be minimal visible evidence that the car has been hired.)

2) The bar code wands at the time were designed to be placed in physical contact with the code and dragged along it. Hence, their focal length was the same as the wand length. We could calculate the anticipated thickness of the glass and alter the tips (grinding, for the tests) to allow for that thickness.

We selected the second option, and it worked. The first read rate was, for a trained operator, up in the high 90%s.

On the return flight, I had just passed through airport security when I noticed that my bag hadn't. The X-ray machine operator was staring intently at the screen. I enquired as to the matter and they asked that I open my bag. Inside was a bar code wand with a "pistol grip." I immediately understood their concern, and explained about reading bar code ("you know, those silly UPC labels that you find at the grocery market") and such. Eventually, they let me pass. On later flights, I either made sure that I carried a pencil-type wand, or I had the wand at hand and had them hand inspect it.

January 1986

The customer was happy, and our part was done. We heard the first rumours of a new unit from HP.

February 1986

More rumours.

March 1986

We learned enough about the HP-94 to begin some serious thinking. We knew it would have a largish display, be 8086-compatible, have at least 64K bytes of memory, and have an I/O port. We considered the ramifications of this information and considered our experiences with our recent bar code customer. We knew that the customer did not like the open-ended nature of hourly consultants, and that they _did_ like products like Lotus 1-2-3.

We had recently finished a project where we had created an embedded system in C for an 8086-compatible processor. We thus had the tools and development environment right at hand.

We came up with the idea of offering a development system for this machine. This was not intended to be a general-purpose language (we knew that HP were offering a version of Basic with the HP-94), but a highly-tailored system that supported bar code-style data collection in every possible way. It would be a DBase- or 1-2-3-type product: it didn't do anything out of the box. Rather, it made it easy for the user to solve his or her problem in a way that was very natural.

We sounded out our HP sales representative, Roland Mattson. He felt that this type of system would make it easier to sell HP-94s. Bolstered by that news, we started development. As the first requirement of any software project is a fancy code name, we came up with "Rosetta" for the language and "Thoth" for the software. (If you don't understand these references, wander by the British Museum sometime.) The names eventually were dropped from use, but you can still see traces of them in the program header files.

April 1986

We had enough of the system running (on an IBM PC) to offer a demo and trundled out to the customer and gave them a demo. They liked the general approach and the demo, and we felt that we had a winner on our hands.

May 1986

Development continued and we received our first HP-94 and documentation. As little hard information was available, we promptly disassembled the computer, noted the chip numbers, and got data sheets on all of the chips. We were also able to figure out the bus.

We also disassembled part of the ROM to figure out how keyboard, timer, and related code worked. We were looking for "gotchas" that would prevent us from running our product. We found none.

Susan Firestone joined Steinmetz & Brown as a marketing consultant, initially about the need for technical information.

This was announced as the official release date for the HP-94. The release was postponed to October.

June 1986

Our activity had aroused interest in Corvallis, and Jeff Brown and Peter Steinmetz went out to talk with them. Upon his return, they believed in us enough to grant us permission to contact them with technical questions. We had also signed a non-disclosure agreement and had obtained some valuable internal documentation.

July 1986

During this month, coding was essentially complete on version 1.0 of the software. We also formally organised Audax Corp., and spun off the Collect-94 product to that corporation. A private stock offering was also made and we were off an running with capital.

All of that sounds quick and easy, doesn't it? It actually required a huge amount of work: preparing proposals, calling potential investors, revising proposals, calling potential investors. In addition, there was calling potential investors, calling potential investors, and calling potential investors. You get the idea. The "fun" part of all this was "switching gears" between programming and talking with investors. But don't get the wrong idea: people other than I did a lot of talking with investors, too.

However, it was a _very_ nice feeling to walk out of the first investor-turned-stockholder meeting with $84,000 in the bank.

August 1986

Work started on other aspects of the product, including packaging, keyboard overlays, and documentation. We continued showing the product to other potential customers, including 3M Corp. 3M had a human factors department that reviewed all products that they might purchase. After a thorough review, they returned a glowing report on Collect-94, with their only comments being a couple of refinements on the keyboard overlay. (Part of the reason that the review was so thorough was that the reviewer liked playing with the product.) We had a few other changes to make to the keyboard overlay anyway, so we decided to do a second run right away, writing off the bulk of our initial run of 500.

In late August, we received word from Corvallis: JUST WHAT WERE WE UP TO, ANYWAY?? Our sales efforts were very visible, and they were wondering just what sort of vapourware we were selling. Little did they know...

September 1986

We arranged a meeting/presentation for September 14 in Corvallis at which we would explain to them what we were doing. All of our efforts were focused around getting ready for that meeting. Well, almost all. I got side-tracked ordering bookcases for the office during that time, much to the consternation of Susan Firestone.

In 1986, most computer software was shipped in 3-ring, vinyl binders. We worked with a local binder company and reviewed the options. One concern was the cold Minnesota weather. We didn't want products shipped during the winter to arrive cracked due to sitting in the delivery van while going from our offices to the shipping terminal. So, instead of the traditional thin vinyl over cardboard, we selected a "slab" plastic style.

This style of binder had no explicit hinge, just a thinned-area of plastic. We were concerned about cracking in this "hinge." So, we performed three tests on the binders.

First, we placed some in my home freezer. (This was September, not January.) After letting them sit overnight, I pulled them out and, while still folded, hit them on the hinge with a hammer. No breakage.

Second, I sat down one night, pulled out my video disc of "Journey to the Center of the Earth," and proceeded to fold a binder back and forth 360 degrees, Every 50 folds I made a mark. The movie -- and my arms -- ran out after about 5,000 folds. (5,000 folds is equivalent to 20 folds per day for a year.) While signs of wear were visible, the hinge had not deteriorated significantly.

Third, we sent a few to a materials testing consultant. He tested them from -40 deg C to +65 deg C and judged them acceptable. (Some of them failed at -195 deg C.)

The binders were judged acceptable. The next step was to finish the documentation. We did, printed the master copy, and brought it to Kinko's to be copied. The clerk looked at the copyright notice and said that they couldn't copy it. Panic! We had to have copies in two days. "But, these copies are _for_ Audax Corp.," I state. "OK," said the clerk, "prove it." I was stymied. Our cards wouldn't arrive until the next day. Fortunately, we thought to put our phone number on the manual and, even more fortunately, I knew that someone would be at the office even though it was 22.00. So I had the clerk call the company for confirmation. Whew! One more problem solved.

The next day, we had copies of the manual, binders, and a distribution disk. We were still lacking printed disk labels and the new keyboard overlays. But, we were as ready as we could be, so we left for Corvallis.

Our HP Sales Representative's luggage was not delivered, so she attended the meeting in informal travelling clothes: a "referee"-style black-and-white striped shirt, jeans, and tennis shoes. She wound up spending three days in those clothes. Fortunately, that did not set the tone for the meeting.

Corvallis. We were in a fairly large conference room, with the Audax people (Susan Firestone, Jeff Brown, and I) clustered at one table, and the rest of the tables filled with 20-30 HP people. We had our software loaded on a HP-94 and we showed how it worked. During that talk, a Federal Express package arrived that contained some sample disk labels. A label was quickly attached to the distribution disk.

We were questioned at length about quality, and I explained how we developed in a high-level language, re-used modules between interpreters, and had the compiler/decompiler. I also explained how we extended quality so other areas such as the binders. During this talk, another Express courier arrived with samples of the new keyboard overlay. By the time we were finished, a complete Collect-94 product had been assembled in the room!

After the meeting, the HP quality assurance person drew me aside and said that he wished he could get the people at HP to take quality as seriously as we did!

We were taken on a tour of the building while the HP people conferred. They agreed that we had a real product (and were blown away by having it assembled during the meeting (that was not a coincidence: it had been carefully orchestrated by Susan Firestone)). They also made a landmark decision for HP.

While the HP-94 hardware had been pre-released in May, shipments were held up because their development system wasn't ready and it was pointless to ship hardware that the customer could not do anything with. However, our development system was clearly ready. So, for the first time in HP history, HP would ship hardware on the condition that the customer had purchased a third party development system!

With our customers now able to obtain product, we were able to start marketing and selling more effectively. We sold quite a few development systems, and build a modest dealer/VAR/OEM network.

October 1986

This was announced as the (second) official release date for the HP-94. The release was postponed to January.

The HP development system eventually shipped in late October. The entire HP-94 package wasn't officially released until 7 January 1987!

Susan Firestone and I attend the ScanTech (bar code industry) trade show in San Francisco and meet other handheld manufacturers. In particular, Furuno Electronics.

February 1987

Susan Firestone and I go to venture capitalist to try and get more money. Unfortunately due to problems with a different local venture offering (Endotronics), local venture capitalist were not interested.

March 1987

Susan Firestone goes to Hannover Cebit and talks with Parcon and a potential German VAR.

May 1987

People are buying development systems but not interpreters. We do consulting work for the City of Minneapolis to get money.

Furuno visited us.

September 1987

We change royalty agreements, we are not in good shape, BUT we are getting an increase in interest and are running on a majority of HP-94's.

October 1987

US stock market crash. We become listed and soon to be referenced software suppliers of HP.

November 1987

The dollar starts falling. HP Colorado returns our development system. We get a large increase in calls from both customers and representatives from the new software solutions book HP puts out. Business looks like it will pick up.

February 1988

HP announces that it will stop shipment of HP-94's in May. Still a lot of interest, one development system sale, but things start dropping off as people find out about the product stopping.

May 1988

No more HP-94 (:-(and we sell our last interpreter to Weed Instruments.

OBSERVATIONS

Several customers purchased the HP development system. By the time that the HP-94 story was over, all of those customers had converted to Collect-94, giving us 100% market penetration.

We had extensive plans for the product. Version 2 -- which was well under development but never finished -- had:

- full floating point and the usual log and trig function support
- strings up to 255 characters, NULL character allowed
- more control structures (For, While, nested Ifs)
- "dumb" bar code wand support (courtesy of a deal with Bernie Musch's company, Firmware Corp.)
- a much improved manual

It quickly became fairly obvious that the market would not explode overnight, so we looked at porting our software to other vendors' hardware and attempted to sell another round of stock. Both efforts came to naught, and by May 1986 the company was essentially out of business.

In hindsight, we should have raised more money and came up on other machines. Our planned second round of financing would have added three new vendors (Furuno, Telxon, and an unspecified third).

As time has gone by and Susan Firestone spoke with different HP and former HP people and other manufacturers, she observes that the dollar's fall in late 1987 and early 1988 -- from which it still has not completely recovered -- would have meant certain death to any software company not running on US-made portable data collectors (at least two of our four would have been US-made) _and_ even those companies took a big dip at that time.

The Yen/$ problem was supposedly the _real_ reason HP split from tradition and dropped the 94.

The HP-94 had originally been conceived by HP as part of a "route accounting" package. However, the VAR and peripheral support (particularly a rugged printer) never materialised. So, an attempt was made to shift its focus to inventory applications.

However, inventory applications usually meant bar coding. In order to make use of the HP-94, the customer would also have to select a bar code type, define a coding system, arrange to have it applied to the items, and modify their mainframe applications to make use of this, as well as adjust their internal procedures. This is a lot of work. Many companies were willing to make this investment. However, as the "lead" product for this system, the HP sales rep (and us!) would have to invest a lot of time and energy making the sale. Once made, the sale could easily amount to a few million US$, so there was a reason to invest this effort.

The problem -- as near as I can figure it out -- was in two, closely related areas. First, the Corvallis division had a great deal of experience selling to bookstores, EduCALC, and similar organisations. Sales were quick and for small amounts of money. That division's culture simply did not think in terms of two years for just one multi-million dollar sale. (The mainframe divisions, on the other hand, operate this way all the time.) Second, people were expecting the HP-94 to "take off" right away. When it didn't, people "got scared" and backed off from the product. Again, it is not the case of a bad product, but of a bad match between peoples' expectations and the product's reality.

We learned a few interesting things as part of our marketing.

In many discussions with potential customers, the managers expressed deep concerns about their employees' ability to learn and handle such complex equipment. When investigated, the employees' had little trouble learning things: it was the managers that couldn't figure out what an "on" button did. For example, when showing it to a grocery store, the manager could only get a 10% or so first read rate. He asked a passing employee to try it, and the employee immediately achieved a 99% first read rate.

Our development system initially sold for US$1,999. We quickly raised the price to US$3,200, as the market was very price-sensitive, but in an inverse way. Susan Firestone and Kristine Young compared our pricing with that of other HP software suppliers (in different markets, of course!, or we wound have been guilty of price-fixing) and found that customers wanted a premium product: HP's customers were not bargain hunters. We specifcally choose US$3,200 for two reasons:
1) typically, any products priced around US$2,000 or above had to go through special purchasing within a company (and there was typically another level at US$4-$5,000), and 2) if it was US$3,200 a manager could say it costs "about 3K." In general it was a heavier number, all an image thing, snob appeal in a way.

For this amount, the customer received a one-year warranty on the software(!), unlimited technical support for one person, and the right to run the development system on as many systems as they wanted. Bug fixes were free. (After the first year, the customer could purchase a service contract that extended all benefits for another year.)

Payment for our product was a combination of up-front money for the development system plus a charge for using the software on each HP-94. We originally called the per-94 price a charge for "run-times." However, customers did not like the idea of paying for the run-time system. We changed the name to "interpreters" and the complaints went away.

One pervasive problem is how to handle volume purchases. I had been involved with companies that had annual purchase commitments, from which per-order discounts were calculated. Hard feelings arose when the purchaser did not make the volume commitment. We tried a novel approach, which was to calculate the discounts on each order by itself, with no long-term commitment, under the theory that the customer could choose how much they wanted to "stock up and save." We found this hard to explain and hard for customers to deal with. Upon reflection, we should have stuck with the "annual commitment" model.

We were very careful to publish list prices for software and stick to them. In fact, we guaranteed to our dealers and OEMs that we would not undersell our list prices. This made us popular with our dealers and OEMs. We also guaranteed that we would not supply finished applications to customers, instead referring them to a dealer. Again, very successful.

PEOPLE

Jeff Brown	Vice-President of Steinmetz & Brown, Director, kibbitzer (did most of the user-interface design)
Ray Connover	Director
Craig Finseth	President, programmer, documentation
Susan Firestone	VP Marketing
Phil Hsiao	kibbitzer
Jennie Kratzer	accountant
William Losby	Director, helped arrange the stock sale
Frank Mabley	Secretary, lawyer
Janet Mills	Administrative assistant
Del Peterson	Director
Peter Steinmetz	President of Steinmetz & Brown, Director, programmer
Sten Anderson	Corvallis
Richard Kirby	Corvallis marketing
Roland Mattson	initial S&B/Audax HP Sales Representative
Eric Vogel	Corvallis technical contact
Kristine Young	later S&B/Audax HP Sales Representative

IN CLOSING

Here is a summary of what Collect-94 and Audax was all about. It was written by Jeff Brown.

Collect-94: Improving the state of the art in data collection.

Our main objective in designing Collect-94 was to build a development system that would make any data collection application genuinely easy to learn and operate. Traditional data collection software

led the user by the nose through a series of questions, giving little user control and making it difficult to correct mistakes. Some software "cured" this problem by creating a labyrinth of nested menus. The procedural programming languages that were popular then (like Basic), made that the only easy way to build a user interface. Every application was a new and -- frequently -- frustrating experience for the user. Working with Apple's Macintosh since 1984 had quickly escalated our standards for software usability. Our experience with the Macintosh had taught us that the only way to produce consistently good applications was to make it easier for the programmer to do it the right way than any other way. Our system would have a standard user interface and tools that would make it easy for anyone to build a good data collection application. We were determined to make Collect-94 substantially modeless and completely under the control of the user.

Our major challenge arose from the fact that the design of the HP-94 was already frozen. It's keyboard, nomenclature and display were not built with our plans in mind and at this point, they could not be changed. The keyboard was small and combined numbers and letters onto the same keys, so we used only a few keys and got a lot of mileage out of them. We designed Collect-94 to leverage the 4-line by 20-character display, which was large for its day. The essence of Collect-94 is the way it feels to the user. It would take several pages of narrative to describe this in terms of features and behaviours. Even then, words are a poor substitute for a few minutes of hands on experience. This is not unique to Collect-94. Most well-designed user interfaces are much harder to describe than to learn first hand. Undaunted by the odds, I will attempt a brief overview in the next couple of paragraphs.

Since people more easily remember things if they can locate them in a physical place, the heart of Collect-94 is a single "ring" of up to ten menus. Each menu has a number (0-9), a title and up to three item choices (A, B, and C). We used the HP-94's LCD display as a window onto the menu ring. When the user presses an arrow key, the display smoothly scrolls right or left to show the menu next door. This helps users develop a sense of the "location" of things. Users can browse through all their options without getting lost in a tangle of submenus. This design also makes full use of the screen, largely eliminating the need for obscure abbreviations. As users gain experience, they can jump to any menu by pressing its corresponding number key, 0-9. If the user knows the exact item he wants, he can press a number and a letter to instantly move to the menu and select the item. This design nurtures beginners without slowing down experts. Since choices are displayed on the screen, with a maximum of two keys needed to select any of up to 30 items, there was no need for soft keys.

Once a menu option is chosen, the application begins a dialog with the user to perform a task such as data collection or transfer. At any time, the user can end the dialog and return to the menu ring by pressing a "Return to menu" key. The user participates in the dialog by picking choices from list browsers, scanning in barcodes or typing numbers and text. Icons indicate the type of data entry needed. Standard error messages trap many data entry errors (like entering text where numbers were needed) as they happened: in many cases without any programmer effort. Working with Collect-94 is intuitive and highly consistent -- in other words, comfortable. Users learn applications quickly with minimal training, feel a strong sense of control and tend not to make procedural errors. Programming Collect-94 is also fairly simple (by 1986 standards). Nearly any common data entry activity was covered by a built-in feature. Most programs are just a couple of pages of calls to the built in user interface toolbox. If you would like to know more about the now-defunct Collect-94, refer to the 22-page Collect-94 Operators Guide. Unlike the hateful world of Unix (ca. 1992) it has styled text and illustrations which do a better job explaining this interesting little sidetrack in history.