

THE BULLETIN OF THE



USER GROUP

+ TI 92

C o n t e n t s :

- 3 Letter of the Editor
- 4 Editorial - Preview
- 5 *DERIVE* & TI-92 User Forum
S. Pollak, T. Yeshno, D. Levi.Rashti, N. Zehavi
- 8 Vedic Mathematics using *DERIVE*
Nelson Urrego P.
- 14 Basic Concepts on Recursion Theory with *DERIVE*
René Hugelshofer
- 24 Dynamic Algebra
Josef Rolfs
- 31 Parallel and Flighty Ribbon Curves
Thomas Himmelbauer
- 36 Mistakes in Representing Graphs / Rounding Errors
- 38 FACTORS, FACTORS and many messages
Johann Wiesenbauer
- 40 Titbits (20)

- [1] **Abituraufgaben mit Graphikrechnern und Taschencomputern, Teil 2**
H.Knechtel & W. Weiskirch (Hrsg), Schroedel 2001, ISBN 3-507-73238-6
- [2] **Prüfungsaufgaben für das Unterrichten mit Derive und dem TI-89/92/92+ (Band 2),**
Vlasta Kokol-voljc & Bernhard Kutzler, bk teachware 2001, ISBN 3-901-769-39-0
- [3] **Moderne Matrix Algebra**, Mit Anwendungen in der Statistik,
K. Schmidt & Götz Trenkler, Springer 1998, ISBN 3-540-64194-7
- [4] – [8] **Skripten von T³ Deutschland zu den Themen:**
SI – Materialien (80 Seiten)
Lineare Algebra und Analytische Geometrie (84 Seiten)
Analysis (107 Seiten)
Analysis Ergänzung (30 Seiten)
Stochastik (59 Seiten)
können bei T³ Deutschland zum Selbstkostenpreis bezogen werden.
Information: <http://www.zkl.uni-muenster.de/t3>

Interesting WEB sites <http://.....>

If you know some interesting web sites, then please let us know!

www.acdca.ac.at

Austrian Center of Didactics of Computer Algebra & T³ Austria
New: we opened an English Section, where you can find a lot of
materials written in English. Enjoy it!

euclid.math.fsu.edu/Science/math

Virtual Library

euclid.math.fsu.edu/Science/Education.html

Section Mathematics Education

euclid.math.fsu.edu/Science/Books.html

Online Books

euclid.math.fsu.edu/Science/Journals.html

Mathematics Electronic Journals

euclid.math.fsu.edu/Science/science.html

Science and Math

www.visualmath.com

Your Virtual Tutor in Math, (Numbers, Set Theory, Algebra,
Trigonometry, Analytic Geometry, Calculus and Statistics)

www.virtu-software.com/articles-math.asp

Articles for Educators
(Example: The Value of "Impossible" Problems)

www.hawaii.edu/suremath/intro_algebra.html

Algebra Story and Word Problems

stratford.8m.com/

An Invitation to Mathematics

www.emis.de

The Electronic Library of Mathematics
Mathematical Journals many of them downloadable for free!
Applied Mathematics Electronic Notes

www.educase.edu

Articles for Educators
(Example: Distributes Education and Its Challenges)

Unfortunately there was a typing error in Nils Hahnfeld's URL. His address is:

www.nilshahnfeld.com and/or www.islands.vi/~shawn/ti_89.htm

Dear friends,

I hope that you had a fine summer and that you could find enough time for yourself and your family to take a rest and to do all the things you have left during the year for the holidays - or at least some of them.

Thanks some colleagues from Colombia and Spain I had the unique chance to visit Bogota (or more correct Santa Fe de Bogota) for giving some workshops at the Distrital University and a talk at the Javeriana University at the occasion of the 1st CIAMAC (Conferencia Iberoamericana de Matemática Computacional). You can imagine that I immediately felt home there when I met a student sitting on the stairs exploring complex numbers with his TI-92. It was a very special experience to meet the wonderful people of South America and to enjoy their overwhelming hospitality. During the conference I met Nelson Urrego, one of the contributors of this issue. Best regards to you.

I am very grateful for Milton Lesmes' care during my stay. We had a lot of mathematical talks and a collection of his ideas will appear in one of the next DNLs.

In this issue you can find two articles which have been waiting for a longer time to be published (Josef Rolfs, Thomas Himmelbauer). They are originally written in German and I tried to translate them partially, resp. completely. (Josef and Thomas, I beg your pardon, if there are some translating errors.) Thomas' considerations on accuracy seem to be very peculiar but they can form a good base for discussions about reliability of computer results in general. Recently I came across a strange behaviour of the TI-92 treating arguments different being members of a list or not. I'll show this in the next DNL.

At several occasions I met DUG members who are enjoying and permanently using Walter Schiller's tool TOM.EXE for comfortably editing *DERIVE5* programs (DNL#40). Among others Nurit Zehavi used TOM in her lecture at university. She sent the original code for the program given on page 13, but because of space problems I can only offer to download the respective text file (together with my file belonging to the "Heronic Triangles") from the well known address (see below).

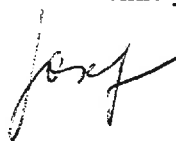
Giulio Barrozzi - the "CAS pope" from Italy - likes programming and TOM, too, and some days ago I received an email, which I'd like to partially quote:

..... I am a great admirer of the programming abilities and the deep knowledge of number theory that Johann Wiesenbauer has. I read that the latest routines in number theory incorporated in DERIVE followed his directions. This is good since the old routines were quite rudimentary.

At last I'd like to thank the many DUG members and who came to attend the ICTMT5 in Klagenfurt and helped to make it a success. My special thanks will go to the contributors and participants of Special Group 1 "DERIVE, TI-92 and other CAS". As we had the most submissions of all in our group, all lectures were very restricted in time, but thanks your wonderful cooperation and discipline we all together mastered the schedule. Thanks to all of you who sent the papers for the proceedings in time.

Best regards together with my wishes for a successful academic year

Josef



Find all the DERIVE and TI-files on the following web sites

<http://www.acdca.ac.at/t3/dergroup/index.htm>

<http://www.bk-teachware.com/main.asp?session=375059>

I'd like to offer an additional service to all our members. If you'd like to have one or another DNL-contribution in electronic form (as a file) then please send a mail and I will send you the file by email.

The *DERIVE-NEWSLETTER* is the Bulletin of the *DERIVE & TI-92 User Group*. It is published at least four times a year with a contents of 40 pages minimum. The goals of the *DNL* are to enable the exchange of experiences made with *DERIVE* and the *TI-92/89* as well as to create a group to discuss the possibilities of new methodical and didactical manners in teaching mathematics.

As many of the *DERIVE* Users are also using the *TI-92/89* the *DNL* tries to combine the applications of these modern technologies.

Editor: Mag. Josef Böhm
A-3042 Würmla
D'Lust 1
Austria
Phone/FAX: 43-(0)2275/8207
e-mail: nojo.boehm@pgv.at

Contributions:

Please send all contributions to the Editor. Non-English speakers are encouraged - but not obliged - to write their contributions in English to reinforce the international touch of the *DNL*. It must be said, though, that non-English articles will be warmly welcomed nonetheless. Your contributions will be edited but not assessed. By submitting articles the author gives his consent for reprinting it in the *DNL*. The more contributions you will send, the more lively and richer in contents the *DERIVE & TI-92 Newsletter* will be.

Next issue: December 2001
Deadline 15 November 2001

Preview: Contributions for the next issues

Inverse Functions, Simultaneous Equations, Speck, NZL
A Utility file for complex dynamic systems, Lechner, AUT
Examples for Statistics, Roeloffs, NL
Quaternion Algebra, Sirota, RUS
Various Training Programs for the TI
Sand Dunes, Halprin, AUS
Type checking, Finite continued fractions, Welke, GER
Kaprekar's "Self numbers", Schorn, GER
Some simulations of Random Experiments, Böhm, AUT
Comparing statistics tools: a pie chart with *DERIVE*,
Stem & leaf diagram with *DERIVE* and on the *TI-92*
Milton's Mathematical Ideas, Lesmes, COL
Another Task for Endexamination, Lechner, AUT
and
Setif, FRA; Vermeylen, BEL; Leinbach, USA; Aue, GER; Koller, AUT,

Impressum:

Medieninhaber: *DERIVE* User Group, A-3042 Würmla, D'Lust 1, AUSTRIA
Richtung: Fachzeitschrift
Herausgeber: Mag. Josef Böhm
Herstellung: Selbstverlag

Richard Schorn, Germany

Richard.Schorn@t-online.de

Concerning the Dragon Curves in DNL#42:

Unfortunately I didn't give references at the end of my contribution in DNL#42, page 19. The "ancestors" in literature for me were the following:

- [1] "Mathematical Games" column by Martin Gardner, Scientific American, Vol. 216, No. 3, pp. 124-125 (March 1967); Vol. 216, No. 4, pp. 118-120 (April 1967); Vol. 217, No. 1, p. 115 (July 1967).
- [2] "Number Representations and Dragon Curves - I", C. Davis & D. E. Knuth, J. Recreational Math., Vol. 3, No. 2, pp. 66-81 (April 1970).
- [3] "Number Representations and Dragon Curves - II", C. Davis & D. E. Knuth, J. Recreational Math., Vol. 3, No. 3, pp. 133-149 (July 1970).

I successfully decoded your RSA-message from DNL#42, page 28

Regards, Richard

Theresa Shelby, Hawaii

tshelby@ti.com

Recently I have received several queries as to how to create a DMO file with Derive 5. Albert and Bernhard thought that you might be interested in my response for inclusion in a future DUG newsletter.

Aloha, Theresa

A DMO (demo) file is in the same format as a MTH file. However, the expressions in a DMO file are automatically simplified as they are read into the worksheet. The annotation will be displayed with the simplified expression.

To create a Demo (*.DMO) file in Derive 5, first enter the expressions that you want to simplify in the worksheet using the Author > Expression command. Then, use the Edit > Annotation command to enter a comment that will be displayed when the expression is simplified upon running the demonstration. Use the File > Save As command to save the file, saving it as a Math file. For example: File Name: "test.dmo", Save as type: Math file (*.mth). Note that if you don't put quotes around the filename it will be saved as "test.dmo.mth". See the Load > Demo File topic in Derive 5's help file for more information.

Michael Johann, Germany

johann@UNI-LANDAU.DE

Is it possible in DERIVE5 to plot a 2D- or 3D-vectorfield?

See two examples:

$F(x,y) = [-y,x]$ for a 2D-representation or

$F(x,y,z) = [x^2+y^2, 2x, \exp(-x^2-z^2)]$ for a 3D-representation.

DNL:

I tried (my best?) using Sergey Birjukow's tool from DNL#16 to generate arrows.

My function

vector_field(v_, xst, xend, xstep, yst, yend, ystep, 1)

sets the middle of the arrow on the point:

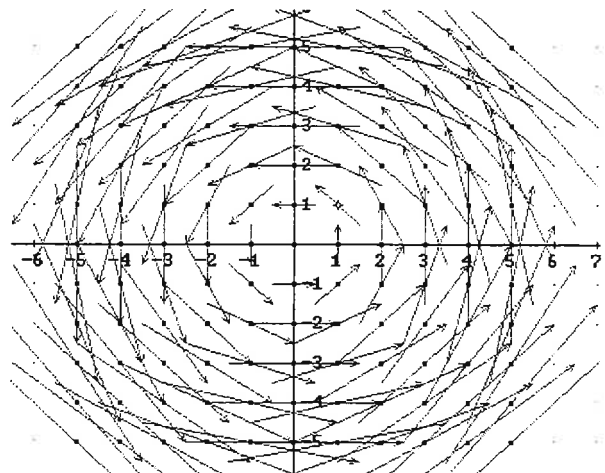


```
vector_field([-y, x], -5, 5, 1, -5, 5, 1, 0.2)
pts(-5, 5, 1, -5, 5, 1)
```

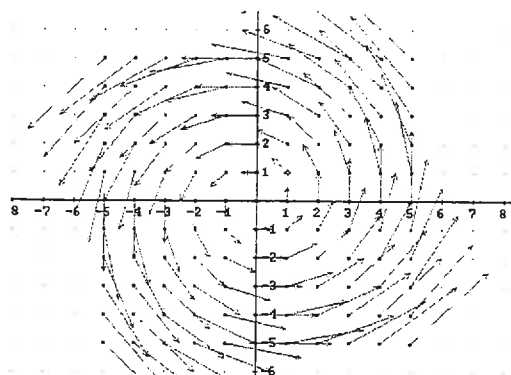
You have to change one auxiliary function to obtain arrows which are starting in the points **pts** (see next page).

I also tried to produce a vectorfield in 3D space. It works, but I am not fully satisfied, because I am missing the arrows – I am working on it. At the other hand it is nice to be able to move the vector field.

Josef

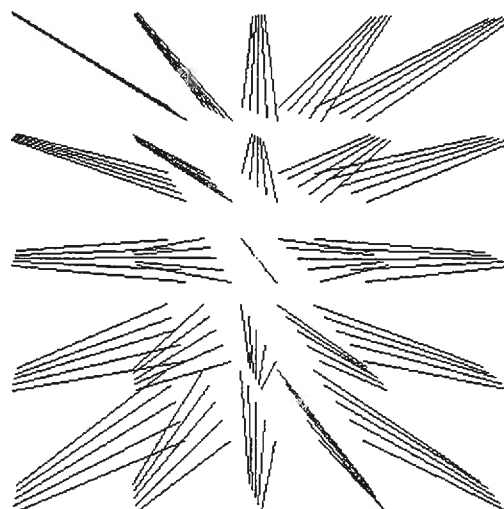
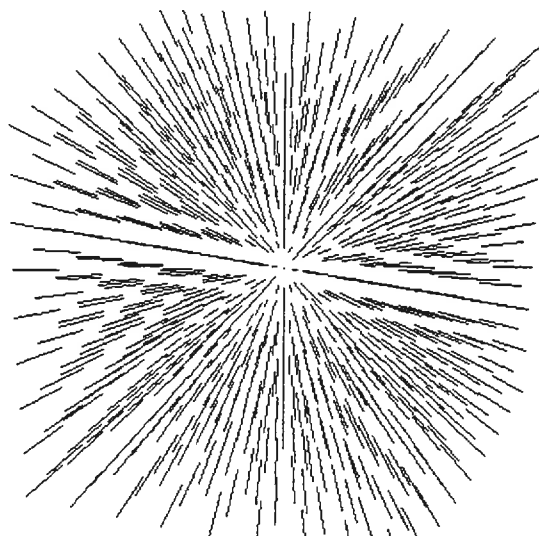


```
vector_field2([-y, x], -5, 5, 1, -5, 5, 1, 0.2)
pts(-5, 5, 1, -5, 5, 1)
```



```
vector_field3\left(\frac{[x, y, z]}{\sqrt{x^2 + y^2 + z^2}}, -4, 4, 1, -4, 4, 1, -4, 4, 1\right)
```

```
vector_field3\left(\left[x, -y, \frac{z}{2}\right], -4, 4, 2, -4, 4, 2, -4, 4, 2\right)
```



DNL:

In this issue you can find a contribution treating RECURSION. So it might be interesting to publish an answer to Hank Schenker's question posed in the DERIVE-News. He asked:

Is an iterative process different from a recursive process, or can it be considered a special case of a recursive process? I've been told at the SCI/MATH forum that they are different things. Many thanks for any help with this. Hank

Lars Erup from Canada (lerup@videotron.ca) offered an answer:

To quote an old computer dictionary: "Recursion: See recursion" :-)

I am not a mathematician, but my understanding of the two is the following:

A recursive algorithm is one which is expressed in terms of itself, such as $\text{factorial}(n) := n * \text{factorial}(n-1)$. Such an algorithm must of course include some "stop criterion", such as "..., except for $\text{factorial}(1)$ which is equal to 1".

An iterative algorithm simply repeats itself a number of times. For example, root finding by bisection:

- 1) Let a be the left end of the interval, b the right end. Let $f_a=f(a)$ and $f_b=f(b)$
- 2) Let $c=(a+b)/2$; let $f_c=f(c)$
- 3) If $\text{sign}(f_c) = \text{sign}(f_a)$ then let $a=c$ and $f_a=f_c$, else let $b=c$ and $f_b=f_c$
- 4) If $\text{abs}(f_c)$ is small enough, or $(b-a)$ is small enough, stop; else go to (2)

Hope this helps, Lars

Hanks reply was:

Dear Lars,

sounds pretty good to me. Of course there may be a theorem that says that any iterative method can be made recursive by appropriate syntactical changes, e.g. by double recursion.

Do you have another explanation for the difference? DERIVE offers the ITERATE – function and RECURSIVE functions!

Marlene Torres-Skoumal from the Vienna International School closed her wonderful and inspiring lecture at the DERIVE & TI-92 Summer Academy in Gössing with a collection of quotes demonstrating how changes in "technology" were accepted by teachers and school authorities during the centuries.

- "Students today depend too much on hand held calculators." (Anonymous, 1985)
- "Ballpoint pens will be the ruin of education in our country. Students use these devices and then throw them away. The American virtues of thrift and frugality are being discarded. Business and banks will never allow such expensive luxuries." (Federal Teacher, 1950)
- "Students today depend upon these expensive fountain pens. They can no longer write with a straight pen and nib, (not to mention sharpening their own quills). We parents must not allow them to wallow in such luxury to the detriment of learning how to cope in the real business world, which is not so extravagant." (PTA Gazette, 1941)
- "Students today depend upon store bought ink. They don't know how to make their own. When they run out of ink they will be unable to write words or ciphers until their next trip to the settlement. This is a sad commentary on modern education." (The Rural American Teacher, 1929)
- "Students today depend too much upon ink. They don't know how to use a pen knife to sharpen a pencil. Pen and ink will never replace the pencil." (National Association of Teachers, 1907)
- "Students today depend upon paper too much. They don't know how to write on slate without chalk dust all over themselves. They can't clean a slate properly. What will they do when they run out of paper?" (Principal's Association, 1815)
- "Students today can't prepare bark to calculate their problems. They depend upon their slates which are more expensive. What will they do when their slate is dropped and it breaks? They will be unable to write!" (Teachers Conference, 1703)
- ... Can you predict what tomorrow's argument will be? ...
- (Tom Seidenberg, Eisenhower High School, Yakima, WA, Washington Mathematics 34 (1) Fall, 1989, p8.

If you would like to find more quotes on mathematics and mathematicians, then you are invited to visit Bernhard Kutzler's home page: www.kutzler.com

Ancient Mathematics and 20th Century Technology: Programming of Vedic Mathematics using *Derive*

Sarah Pollack, Tzippora Yeshno, Daphna Levi-Rashti, and Nurit Zehavi
Weizmann Institute of Science, Israel

Nurit.Zehavi@weizmann.ac.il

Introduction

Using examples from ancient Vedic Mathematics we describe our experience in programming algorithms, which constituted part of our work in a graduate course, *Integration of topics in mathematics through Computer Algebra Systems*. The course was held at the Weizmann Institute of Science, given by the last author.

The *Derive* software can be used as a programming environment. It includes a large collection of built-in mathematical functions. These built-in functions and the computation power enable the user to investigate a problem, to examine specific cases and related concepts, and to check results and solutions. Furthermore, *Derive* enables writing new functions for a specific use. These features can be useful scaffolding for programming solutions to mathematical problems.

In order to demonstrate these features we will present three algorithms from the fascinating topic of ancient Vedic Mathematics which was rediscovered between 1911 and 1918 by Sri Bharati Krsna Tirthaji. Vedic Mathematics is a system of calculations formed of simple rules and principles based on 16 *sutras* (Agrawala, 1971). A *sutra* is one or two line aphorisms that can be easily memorized. This system enables solving orally many mathematical problems. From an algorithmically point of view, the Vedic rules are sets of instructions for solving problems. Therefore, a CAS having both mathematical and programming power is suitable for revisiting Vedic Mathematics.

In the solution process, we explored the Vedic rules and translated algorithms, using the built-in *Derive* functions, to examine specific cases. After we obtained a better understanding of the algorithms, we began writing programs that implemented and generalized the algorithms. We should note, however, that the algorithms presented here do not capture the rich variety of rules for calculation found in the ancient Vedas scriptures.

Since the *Derive* environment does not include an editor, it is not convenient to write large programs. To overcome this problem, we used the pre-compiler, *tom*, written by Schiller (2000). The pre-compiler has two major advantages: (a) the program can be written using a standard editor, and (b) the pre-compiler enables the use of instructions like CASE or WHILE that are not included in the standard *Derive* programming instructions. Therefore we can write more structured programs that are more readable.

Problem 1

Multiplication of two numbers x (the multiplicand) and y (the multiplier), where the multiplier's digits consist entirely of nines; using the *Ekanyena Purvena* *sutra*, which means: **by one less than one before.**

According to this rule, the result of $x \cdot y$ comprises two parts: the prefix is the number $x - 1$ and the suffix is the difference $y - (x - 1)$, which is easy to calculate 'in the head'. For example: $x = 735$ and $y = 999$; the prefix of the multiplication is 734 and the suffix is $999 - 734 = 265$ (where $\text{length}(x) \leq \text{length}(y)$). In this problem the algorithm (as well as the program) is very simple and consists of the following assignment instruction.

Program 1:

**Multiplication of two numbers where
the multiplier's digits consist entirely of nines**

```
multiply(x, y) :=
  Prog
#1:   u := [x - 1, y - (x - 1)]
      r := ELEMENT(u, 1) * (y + 1) + ELEMENT(u, 2)
      RETURN r
```

Testing

```
#2:   multiply(735, 999) = 734265
#3:   735 * 999 = 734265
#4:   multiply(87, 9999) = 87 * 9999 = (869913 = 869913)
```

Question left for the reader:

Why does the procedure `ekapur(x, y)` which obviously transfers the algorithm applying string functions not work properly? Find a multiplication with `ekapur(x, y)` giving a wrong result. (Nice question for pupils!)

```
#5:   ekapur(x, y) := ADJOIN(STRING(x - 1), STRING(y - (x - 1)))
#6:   ekapur(735, 999) = 734265
#7:   ekapur(87, 9999) = 869913
```

Problem 2

A natural number x can be expressed as the difference of two squares,
 $x = m^2 - n^2$, using the Vedic rule: **by addition, by subtraction.**

According to this rule, given a natural number x , find a and b so that $x = a \cdot b$ and then compute $m = (a+b)/2$ and $n = (a-b)/2$.

For example: $x = 12 \Rightarrow a=6, b=2 \Rightarrow m = 4, n = 2$.

If we choose alternatively $a = 4, b = 3$, we will get $m = 3.5, n = 0.5$;

Indeed $12 = 3.5^2 - 0.5^2$.

First we explored the Vedic rule numerically and algebraically by examining specific examples e.g., $x = 23456$. We used the built-in function `DIVISORS` that lists all the divisors of a given number.

From this list we chose $a = 16$ and activated the rule:

(Caution: You must preload the utility file `NUMBER.MTH` to have `DIVISORS` available!)

```

#1:  LOAD(C:\DfW5\Math\Number.mth)
#2:  DIVISORS(23456)
#3:  [1, 2, 4, 8, 16, 32, 733, 1466, 2932, 5864, 11728, 23456]
#4:   $\left[ x :=, a :=, b := \frac{x}{a}, m := \frac{a + b}{2}, n := \frac{|a - b|}{2} \right]$ 
#5:  [x := 23456, a := 16]
#6:  b = 1466
#7:  m = 741
#8:  n = 725
#9:   $x = m^2 - n^2$ 
#10:  23456 = 23456

```

The next step was to generalize the algorithm and to write a program that finds all pairs of natural numbers (m, n) so that $x = m^2 - n^2$, x is greater than 1.

Program 2

For a natural number x find two numbers m, n such that $x = m^2 - n^2$

The function `diff_of_squares(x, a)` gets a natural number x and divider a , and finds the second divider $b = x/a$ and two numbers m and n so that $x = m^2 - n^2$.

```

diff_of_squares(x, a) :=
  Prog
    b := x/a
#11:  m := (a + b)/2
    n := ABS((a - b)/2)
    RETURN [n, m]

```

The function `all_natural_diff_of_squares(x)` gets a natural number, $x > 1$, and finds all the pairs of natural numbers m and n so that $x = m^2 - n^2$:

```

all_natural_diff_of_squares(x) :=
  Prog
    If x < 2 ∨ ¬ INTEGER?(x)
      RETURN false
    u := []
    [v := DIVISORS(x), r := DIM(v), p := ROUND(r/2)]
    i := 1
#12:  Loop
    If i > p
      RETURN u
    a := ELEMENT(v, i)
    diff_of_squares(x, a)
    i := i + 1
    If INTEGER?(n) ∧ INTEGER?(m)
      u := APPEND(u, [[m, n]])

```

Testing

```

#13:  all_natural_diff_of_squares(1) = false
#14:  all_natural_diff_of_squares(12) = [[4, 2]]
#15:  all_natural_diff_of_squares(961) =  $\begin{bmatrix} 481 & 480 \\ 31 & 0 \end{bmatrix}$ 

```

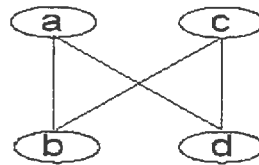
#16: $\text{all_natural_diff_of_squares}(1009) = [[505, 504]]$

#17: $\text{all_natural_diff_of_squares}(23456) = \begin{bmatrix} 5865 & 5863 \\ 2934 & 2930 \\ 1470 & 1462 \\ 741 & 725 \end{bmatrix}$

Problem 3

Multiplication of two numbers using the *Urdhva-Tiryagbhyam* sutra,
which means: **Vertically and cross-wise**.

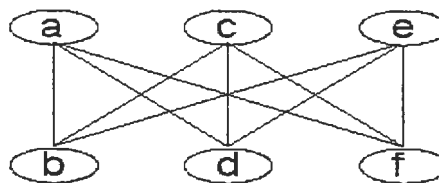
To multiply two-digit numbers $ac \cdot bd$, compute: $a \cdot b$; $a \cdot d + b \cdot c$; $c \cdot d$ (as illustrated in the figure), and carry over numbers as needed.



For example, to multiply 12 by 13, multiply 1 by 1, 1 by 3 added to 2 by 1, 2 by 3, and then write the answer as $[1, 5, 6]$ (meaning 156). Note that the answer is obtained in one line.

To multiply two numbers with three digits $ace \cdot bdf$, compute:

$a \cdot b$; $a \cdot d + c \cdot b$; $c \cdot d + a \cdot f + e \cdot b$; $c \cdot f + e \cdot d$; $e \cdot f$.



First we explored the Vedic rule using the case “multiplication of two numbers with three digits each”: $ace \cdot bdf$, and prepared a preliminary algorithm based on it.

Case 1:

$ace \cdot bdf \Rightarrow [a \cdot b, a \cdot d + b \cdot c, b \cdot e + a \cdot f + c \cdot d, c \cdot f + e \cdot d, e \cdot f]$, then carry over numbers as needed.

#1: $194 \cdot 285 = 55290$

#2: $[a := 1, c := 9, e := 4, b := 2, d := 8, f := 5]$

#3: $[a \cdot b, a \cdot d + b \cdot c, b \cdot e + a \cdot f + c \cdot d, c \cdot f + e \cdot d, e \cdot f] = [2, 26, 85, 77, 20]$

Carry over ‘2’ from the 5th element to the 4th element, carry over ‘7’ from the 4th element to the 3rd element, and so on.

#4: $[2 + 2, 6 + 8, 5 + 7, 7 + 2, 0] = [4, 14, 12, 9, 0]$

#5: $[4 + 1, 4 + 1, 2, 9, 0] = [5, 5, 2, 9, 0]$

Next we dealt with the following case:

Case 2:

$ace \cdot df \Rightarrow ace \cdot 0df$, in this case we set 'zero' as the 'first digit' of the shorter number (df) and then activate the rule for $ace \cdot bdf$.

We prepared a sketch of the algorithm for $m \cdot n$ where $m=48$ and $n=53$.

Step 1. Find the digits of m and n :

#6: $[a :=, b :=, c :=, d :=, e :=, f :=]$

#7: $[m := 48, n := 53]$

#8: $\left[c := \text{MOD}(m, 10), a := \frac{m - c}{10} \right] = [8, 4]$

#9: $\left[d := \text{MOD}(n, 10), b := \frac{n - d}{10} \right] = [3, 5]$

Step 2. Activate the Vedic rule

#10: $v := [a \cdot b, a \cdot d + b \cdot c, c \cdot d]$

#11: $[20, 52, 24]$

Step 3. Handle a carry over from 24 to 52 and from 54 (=52 + 2) to 20:

#12: $u := \left[u_1, \text{FLOOR}(u_3, 10) + u_2, \text{MOD}(u_3, 10) \right]$

#13: $[20, 54, 4]$

#14: $\left[u_1 + \text{FLOOR}(u_2, 10), \text{MOD}(u_2, 10), u_3 \right]$

#15: $[25, 4, 4]$

The result is $48 * 53 = [2, 5, 4, 4]$.

After the exploratory stage we were ready to write the main algorithm for multiplying any two natural numbers.

Program 2

Vertically and Cross-Wise

```

decompose(x) :=
  Prog
  y := []
  Loop
#16:   If  $\neg x > 0$  exit
        n := MOD(x, 10)
        y := APPEND([n], y)
        x := (x - n)/10
  RETURN y

```

The function `decompose(x)` decomposes a number x to its digits and returns a list y of the digits.

The function `set_zeroes(x, s)` gets x as a list of digits, and returns a list with s zeroes as the most significant digits. The purpose of this function is to extend the number of digits of each of the multipliers, m and n , to $\text{length}(m) + \text{length}(n) - 1$.

```

set_zeroes(x, s) :=
  Prog
  Loop
#17:   If  $\neg s > 0$  exit
       x := APPEND([0], x)
       s := s - 1
  RETURN x

```

The main function `line_mul(m,n)` multiplies two natural numbers m and n using the Vedic rule: "Vertically and Cross-Wise". The function decomposes the numbers m and n , multiplies and accumulates their digits starting with the least significant digit.

```

line_mul(m, n) :=
  Prog
  [q := decompose(m), k := DIM(q), r := decompose(n), l := DIM(r)]
  [t := k + l - 1, u := t - k]
  [q := set_zeroes(q, u), u := t - 1, r := set_zeroes(r, u)]
  [z := 0, h := 1, c := 0, i := t]
  Loop
  If  $\neg i \geq 1$  exit
#18:   [v := t, w := 0, j := i]
       Loop
       If  $\neg j \leq t$  exit
       [e := ELEMENT(q, j), d := ELEMENT(r, v)]
       [w := w + e·d, v := v - 1, j := j + 1]
       [w := w + c, z := z + MOD(w, 10)·h]
       [c := (w - MOD(w, 10))/10, h := h·10, i := i - 1]
  z := z + c·h
  RETURN z

```

Testing

```

#19:           524·345 = 180780
#20:           line_mul(524, 345) = 180780
#21:           132450·9876 = 1308076200
#22:           line_mul(132450, 9876) = 1308076200

```

A concluding remark

We have described here the use of *Derive* as an environment for investigating, developing, and implementing algorithms for solving mathematical problems. We used problems from the ancient system of Vedic Mathematics to demonstrate new features in *Derive*. We believe that exploring this interesting mathematics with new technology would give students an opportunity to learn both about the history of mathematics and about programming. Mathematics teachers can use the material to enrich the practice of algebraic manipulations with meaningful problems, by challenging students to justify the Vedic rules using symbolic expressions. Computer Science teachers can benefit from the repertoire of new problems that can be used in a basic course on algorithms and in advanced computer science courses like concurrency.

References:

- Agrawala, V.S. (1971). *Vedic Mathematics*, Delhi: Motilal Banarsidass.
 Schiller, W. (2000). Programming with Derive, *THE DERIVE NEWSLETTER*, #40, 25-38.
 Deepa J. (2000). The Natural Calculator, Proceedings Portoroz, 89 – 92, bk teachware

Websites on Vedic Mathematics:

- <http://www.vnn.org/editorials/ET9907/ET19-4325.html>
<http://www.vedicmaths.org.uk>
<http://www.ourkarnataka.com/vedicm/vedicms.htm>
<http://www.ics.uci.edu/~rgupta/vedic.html>
<http://www.magicalmethods.com/Default.htm>

BASIC CONCEPTS ON RECURSION THEORY WITH *DERIVE*

Nelson Urrego P.
Department of Mathematics
Pontificia Universidad Javeriana
Bogotá, Colombia
nurrego@javeriana.edu.co

Abstract

Recursion Theory is one of the most important branches of Mathematics because, when considering Recursive Functions as those applications which can be carried out by a computer machine, we are dealing with a powerful discipline that supplies the limits of mechanical procedures (algorithms). In this Paper, the author gives a short introduction to the main concepts about Recursive Functions and some examples are programmed using DERIVE. These exercises can motivate students in the design of algorithms for solving rigorous arithmetic problems such as the implementation of a procedure for generating a 1-1 Primitive Recursive correspondence between N^2 and N .

Introduction

Two years ago, I was selected for giving a short course on Recursion Theory and basic concepts on Computability. The participants were Math teachers and Math students who were familiarized, at least, with some fundamentally notions about Logic and Computing Theory. This course was programmed for been seek in one week so although main of the contents must be summarized, these concepts should be presented in a special form so at the end of the course most of the participants would be interested in keeping work on Recursive Theory and Computability.

Towards principles of the 20th century, mathematicians began to worry for solving problems, in different branches of Mathematics, where a more precise concept of the algorithm idea was required.

Intuitively speaking, we can define an algorithm saying that it is a process, where no creative invention is involved, that searches for the solution of a given problem by a finite number of steps. However, this ambiguous idea was not sufficient if we were interested in solving theoretical problems such like the Hilbert's tenth problem (to give an algorithm which will tell of a given polynomial Diophantine equation with integer coefficients whether or not it has a solution in integers); or the well known *Decision Problem* to give a mechanical procedure to determine the truly of falsity of a sentence of the Predicate Calculus.

In 1931 Kurt Gödel published his famous Incompleteness theorems which resulted in the end of some important basic goals of Hilbert's Program. Gödel proved that there is no way of constructing a consistent first-order formal system so every true statement representable within the system are provable and that the consistency of such system cannot be proved with methods involved on it. As an important element in his demonstration, Gödel introduces the formal notion of a recursive function, which allows to define those functions whose values can be calculated by a machine.

Arithmetic Functions

A function f from \mathbb{N}^n to \mathbb{N} , where \mathbb{N}^n is the n -fold cartesian product of \mathbb{N} with itself is called *Arithmetic*. We can define new Arithmetic Functions from other by means of the following rules:

1. Composition

If f is an arithmetic function with m arguments and each of the g_1, g_2, \dots, g_m are functions of n arguments, the function

$$h(x_1, \dots, x_n) = f(g_1(x_1, \dots, x_n), \dots, g_m(x_1, \dots, x_n))$$

is obtained from f and g by composition.

2. Primitive Recursion

If g is a function with n arguments and h is a function with $n + 1$ arguments, then the function f defined as follows

$$\begin{aligned} f(x_1, \dots, x_n, 0) &= g(x_1, \dots, x_n) \\ f(x_1, \dots, x_n, y+1) &= h(x_1, \dots, x_n, y, f(x_1, \dots, x_n, y)) \end{aligned}$$

is obtained from g and h by Primitive Recursion.

3. Minimalization

If g is a function with $n + 1$ arguments, the function h defined as follows $h(x_1, \dots, x_n) = \mu_y [g(x_1, \dots, x_n, y) = 0]$, the smallest y such that $g(x_1, \dots, x_n, y) = 0$ is obtained from g by minimalization.

Here minimalization is restricted to those cases in which there is at least one y for any $\langle x_1, \dots, x_n \rangle$ such that $g(x_1, \dots, x_n, y) = 0$.

In general for the predicate $P(x_1, \dots, x_n, y)$, the notation $\mu_y P(x_1, \dots, x_n, y)$ indicates the least y such that $P(x_1, \dots, x_n, y)$ is true if there is any y .

We shall call the following arithmetic functions the *Initial functions*:

i) The Successor Function:

$$s(x) = x + 1$$

ii) The Zero Function

$$Z(x) = 0$$

iii) The Identical Function:

$$I_i^n(x_1, \dots, x_n) = x_i \quad (1 \leq i \leq n).$$

I didn't want the course had the traditional treatment, so I proposed to the audience that while we were defining different functions and concepts we should intend to design the algorithm that could represent it. In this way we could understand in a practical form why Recursive Functions are considered as a Universal Model of Computation.

The first try was for designing algorithms in Turing Machines to represent the Initial Functions and then being visualized with the help of the simulator Visual Turing 1.0. The successor function can be represented in a Turing Machine by means of this program:

$$\mathbf{M}_1 \quad \begin{array}{c|c} q_0 & \text{L } q_0 \\ q_0 \text{ B} & q_1 \end{array}$$

Here the alphabet is $\Sigma = \{ |, B \}$, where B represents a blank on the tape. An integer n can be represented by an unbroken string of $n+1$ symbols $|$ on an otherwise blank tape. The n -tuple $\langle x_1, \dots, x_n \rangle$ can be represented by n unbroken strings on an otherwise blank tape and separated each other by a blank cell.

Example 1

The number 3 can be represented on the tape as it is indicated in figure 1.

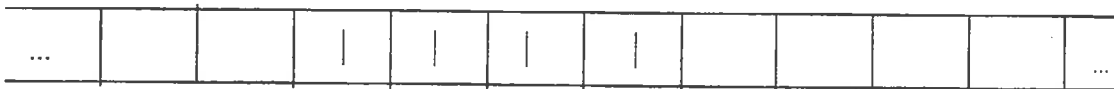


Figure 1. Representation of number 3 on the tape of a Turing Machine

Example 2

The expression $\langle 0, 2, 1 \rangle$ can be represented on the tape it is indicated in figure 2.

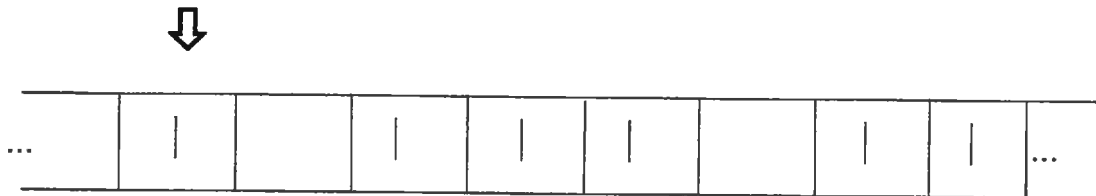


Figure 2. Representation of expression $\langle 0,2,1 \rangle$ on the tape of a Turing Machine.
The reading head appears scanning the leftmost symbol on the tape.

We used Visual Turing in order to visualize the transition of the reading head (Figure 3).

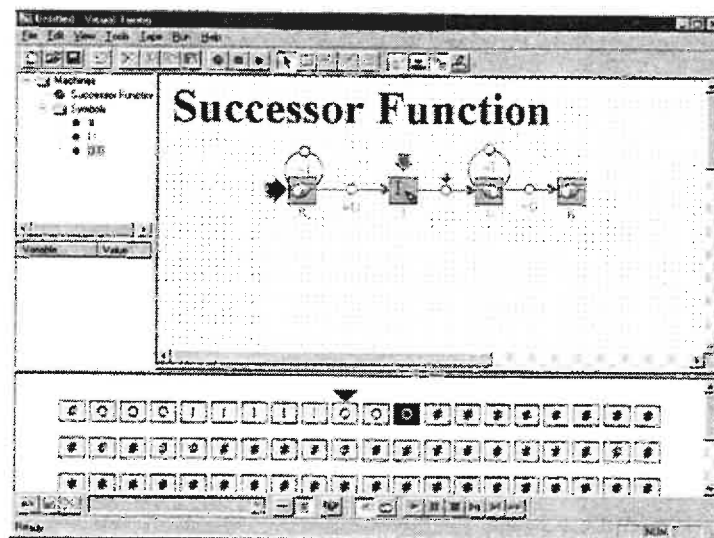


Figure 3. Using Visual Turing to visualize the transition of the reading head for successor function

We can define very simple functions with the help of Turing Machines, but we also desired to define algorithms for more complex functions, one more efficient computing language is required and here is where *DERIVE* is used.

Primitive Recursive Functions

An Arithmetical Function is said to be *Primitive Recursive* if it is one of the Initial Functions or can be obtained from them by finite applications of rules 1 - 2.

Examples 3-16 are Primitive Recursive Functions

Example 3: Predecessor Function:

$$\text{Pred}(x) = \begin{cases} \text{Pred}(0) = 0 \\ \text{Pred}(s(x)) = x \end{cases}$$

Example 4: Addition Function:

$$\text{add}(x, y) = \begin{cases} \text{add}(x, 0) = x \\ \text{add}(x, s(y)) = s(\text{add}(x, y)) \end{cases}$$

Example 5: Product Function:

$$\text{prod}(x, y) = \begin{cases} \text{prod}(x, 0) = 0 \\ \text{prod}(x, s(y)) = \text{add}(\text{prod}(x, y), y) \end{cases}$$

Example 6: Exponent Function:

$$x^y = \begin{cases} x^0 = 1 \\ x^{s(y)} = \text{prod}(x, x^y) \end{cases}$$

Example 7: Factorial Function:

$$x! = \begin{cases} 0! = 1 \\ s(x)! = \text{prod}(x, x!) \end{cases}$$

Example 8: Difference Modified Function:

$$x \dot{-} y = \begin{cases} x - y & \text{if } x > y \\ 0 & \text{if } x \leq y \end{cases}$$

Example 9: Signum Function:

$$\text{sg}(x) = \begin{cases} \text{sg}(0) = 0 \\ \text{sg}(s(x)) = 1 \end{cases}$$

Example 10: Contrary Signum Function:

$$\overline{\text{sg}}(x) = 1 \dot{-} \text{sg}(x)$$

Example 11: Absolute Difference Function:

$$|x - y| = \text{add}(x \dot{-} y, y \dot{-} x)$$

Example 12: Minimum Function (two arguments):

$$\text{minim}(x, y) = x \dot{-} (x \dot{-} y)$$

Example 13: Maximum Function (two arguments):

$$\text{maxim}(x, y) = \text{add}(y, x \dot{-} y)$$

Example 14. Remainder Function (remainder upon division of y by x):

$$\text{rem}(x, y) = \begin{cases} \text{rem}(x, 0) = 0 \\ \text{rem}(x, s(y)) = s[\text{prod}(\text{rem}(x, y), \text{sg}(|x - s(\text{rem}(x, y))|))] \end{cases}$$

#21: Example 5: Product Function

```

PROD(x, y) :=
  If y = 0
#22:      0
          ADDIT(x, PROD(x, PRED(y)))

```

#23: PROD(12, 13) = 156

#24: Example 6: Exponent Function

```

POT(x, y) :=
  If y = 0
#25:      1
          PROD(x, POT(x, PRED(y)))

```

#26: POT(3, 5) = 243

#27: Example 7: Factorial Function

```

FACT(x) :=
  If x = 0
#28:      1
          PROD(x, FACT(PRED(x)))

```

#29: FACT(5) = 120

#30: FACT(6) = 720

#31: takes 5 seconds, FACT(10) exhausts memory!?

#32: Example 8: Difference Modified Function

```

DIFF(x, y) :=
  If y = 0
#33:      x
          PRED(DIFF(x, PRED(y)))

```

#34: DIFF(10, 4) = 6

#35: DIFF(4, 10) = 0

#36: Example 9: Signum Function

```

SIG(x) :=
  If x = 0
#37:      0
          1

```

#38: Example 10: Contrary Signum Function

#39: SIG_(x) := DIFF(1, SIG(x))

#40: [SIG_(10), SIG_(0)] = [0, 1]

#41: Example 11: Absolute Difference Function

#42: DIFF_ABS(x, y) := ADDIT(DIFF(x, y), DIFF(y, x))

#43: DIFF_ABS(23, 11) = 12

#44: DIFF_ABS(11, 23) = 12

#45: Example 12: Minimum Function of two arguments

#46: MINIM(x, y) := DIFF(x, DIFF(x, y))

#47: MINIM(123, 122) = 122

#48: Example 13: Maximum Function of two arguments

#49: MAXIM(x, y) := ADDIT(y, DIFF(x, y))

The *DERIVE* package has internal functions for each of examples 3-13, except maybe for 8 and 11. The objective was to be able of designing every algorithm without the support of those internal functions as well as it was possible. Examples 14-16 require some of the functions defined previously:

#50: Example 14: Remainder Function

```
REMAIND(x, y) :=
  If y = 0 ∨ MODS(y, x)
#51:      0
          S(PROD(REMAIND(x, PRED(y)), SIG(DIFF_ABS(x, S(REMAIND(x, PRED(y)))))))
#52:      [REMAIND(6, 20), REMAIND(5, 20), REMAIND(7, 20)] = [2, 0, 6]
```

#53: Example 15: Quotient Function

```
QUOT(x, y) :=
  If y = 0
#54:      0
          ADDIT(QUOT(x, PRED(y)), SIG(DIFF_ABS(x, S(REMAIND(x, PRED(y))))))
#55:      QUOT(6, 20) = 3
#56:      [QUOT(6, 20), QUOT(5, 20), QUOT(7, 20)] = [3, 4, 2]
```

General Recursive Functions

An arithmetic function f is said to be *General Recursive* (or simply *Recursive*) if f can be obtained by finite applications of rules 1-3. Reader can note that every Primitive Recursive Function is General Recursive. Examples 17 and 18 are recursive functions and can be defined with *DERIVE* as indicates expressions 44 and 46:

Example 17: $x/2$ – Function:

$$\frac{x}{2} = \mu_y [|\text{add}(y, y) - x| = 0]$$

Example 18: $\lceil \sqrt{x} \rceil$ = the greatest integer $\leq \sqrt{x}$:

$$\lceil \sqrt{x} \rceil = \mu_y [\overline{\text{sg}}(s(y) - x) = 0]$$

#60: Example 17: $x/2$ Function defined in terms of μ -operator

```
#61: MID(x) := MIN(VECTOR(IF(DIFF_ABS(ADDIT(y, y), x) = 0, y, x), y, 0, x))
#62:      [MID(20), MID(21), MID(22)] = [10, 21, 11]
#63: Example 18: Square Integer Function
#64: SQUAR(x) := MIN(VECTOR(IF(SIG(DIFF(PROD(S(y), S(y)), x)) = 1, y, x), y, 0, x))
#65:      [SQUAR(15), SQUAR(20), SQUAR(25)] = [3, 4, 5]
```

Recursive Predicates

A predicate $P(x_1, \dots, x_n)$ is an expression where variables $\langle x_1, \dots, x_n \rangle$ can be replaced by natural numbers $\langle a_1, \dots, a_n \rangle$, so it is possible to determine the truly or falsity of $P(a_1, \dots, a_n)$ e.g. the predicate $P(x, y) := x + y = 5$ is true for the pair $\langle 2, 3 \rangle$, but is false for $\langle 3, 4 \rangle$.

The *Characteristic Function of P* is an arithmetical function defined by:

$$C_P(a_1, \dots, a_n) = \begin{cases} 0 & \text{if } P(a_1, \dots, a_n) \text{ is true} \\ 1 & \text{if } P(a_1, \dots, a_n) \text{ is false} \end{cases}$$

A predicate $P(x_1, \dots, x_n)$ is recursive if its characteristic function is recursive. Table 1 contains some examples of Recursive Predicates.

Predicate	Description	Characteristic Function
Example 19: $x \mid y$	x divides to y	$\text{divi}(x, y) = \text{sg}(\text{rem}(x, y))$
Example 20: $x < y$	x is less than y	$\text{min}(x, y) = \overline{\text{sg}}(y - x)$
Example 21: $x = y$	x is equal to y	$\text{eq}(x, y) = \overline{\text{sg}}(x - y)$
Example 22: $\text{Pr}(x)$	x is a prime number	$P(x) = \text{sg}[(D(x) - 2) + \overline{\text{sg}}(x - 1) + \overline{\text{sg}}(x - 0)]$

Table 1. Examples of Recursive Predicates.

Using some Recursive Functions defined above with *DERIVE* we have programmed the following Characteristic Functions:

```
#69: Example 19:
#70: DIUI(x, y) := SIG(REMAIND(x, y))
#71: [DIUI(3, 15), DIUI(4, 15)] = [0, 1]
#72: Example 20:
#73: MENOR(x, y) := SIG(DIFF(y, x))
#74: [MENOR(11, 17), MENOR(17, 17), MENOR(23, 17)] = [0, 1, 1]
#75: Example 21:
#76: EQUAL(x, y) := SIG(DIFF_ABS(x, y))
#77: [EQUAL(5, 5), EQUAL(5, 6)] = [0, 1]
#78: Example 22:
#79: BE_PRIME(x) := SIG(ADDIT(ADDIT(DIFF(DIUS(x), 2), SIG(DIFF_ABS(x, 1))),
    SIG(DIFF_ABS(x, 0))))
#80: VECTOR(BE_PRIME(u), u, [3, 4, 5, 6, 14, 19]) = [0, 1, 0, 1, 1, 0]
```

Now we can define a Recursive Function in order to determine the n^{th} prime number starting from 0:

$$\text{Prime}(n) = \begin{cases} \text{prime}(0) = 2 \\ \text{prime}(s(n)) = \mu_{y \leq \text{prime}(n)} (\text{prime}(n) < y \wedge P(y)) \end{cases}$$

This function can be programmed with *DERIVE* as follows:

```
PRI(x) :=
  If x = 0
#81: 2
  MIN(VECTOR(IF(PRI(PRED(x)) < y ^ BE_PRIME(y) = 0, y, PRI(PRED(x)) + 1), y, 0,
    PRI(PRED(x))! + 1))
#82: PRI(3) = 5
```

Finally, we shall construct a function that generates a Primitive Recursive 1-1 correspondence between \mathbb{N}^2 and \mathbb{N} .

Pairs of natural numbers can be enumerated on a list as follows:

$\underbrace{(0,0)}_{1^{\text{st}} \text{ group}}, \underbrace{(0,1),(1,0),(1,1)}_{2^{\text{nd}} \text{ group}}, \underbrace{(0,2),(2,0),(1,2),(2,1),(2,2)}_{3^{\text{th}} \text{ group}}, \underbrace{(0,3),(3,0),(1,3),(3,1),(2,3),(3,2),(3,3)}_{4^{\text{th}} \text{ group}}, \dots$

Here pairs in the k^{th} group (starting from 1) are ordered in this form:

$$(0, k+1), (k+1, 0), (k+1, 1), (1, k+1), \dots, (k, k+1), (k+1, k), (k+1, k+1).$$

If $x = y$, then (x, y) is the $(x+1)^2$ pair of the $(x+1)^{\text{th}}$ group. On the other hand, if $x < y$, then (x, y) appears before (y, x) and it is the $(2x+1)^{\text{th}}$ pair of the $(y+1)^{\text{th}}$ group. The function $\sigma(x, y)$ determines the place of (x, y) on the list. The pair $(0, 0)$ occupies the 0^{th} place on the list:

$$\sigma(x, y) = (\text{sg}(x - y)) \cdot (x^2 + 2y + 1) + (\overline{\text{sg}}(x - y)) \cdot (y^2 + 2x)$$

```
#85:  σ(x, y) := ADDIT(PROD(SIG(DIFF_ABS(x, y)), ADDIT(ADDIT(PROD(x, x), PROD(2, y)), 1)),
      PROD(SIG_(DIFF_ABS(x, y)), ADDIT(PROD(y, y), PROD(2, x)))))
```

```
#86:                                     σ(3, 3) = 15
```

The functions $\alpha(n)$ and $\beta(n)$ such that $\alpha(\sigma(x, y)) = x$ and $\beta(\alpha(\sigma(x, y))) = y$ are defined below and determine the inverse function $\psi(n) = (\alpha(n), \beta(n))$ which denotes the n^{th} pair on the list:

$$\alpha(n+1) = \begin{cases} \beta(n) & \text{if } \alpha(n) < \beta(n) \\ \beta(n) + 1 & \text{if } \alpha(n) > \beta(n) \\ 0 & \text{if } \alpha(n) = \beta(n) \end{cases}$$

$$\beta(n+1) = \begin{cases} \alpha(n) & \text{if } \alpha(n) \neq \beta(n) \\ \alpha(n) + 1 & \text{if } \alpha(n) = \beta(n) \end{cases}$$

This correspondence can be programmed in *DERIVE5* as follows:

```
#87:  [α(x) :=, β(x) :=]
```

```
      α(x) :=
      If x = 0
      0
      If α(PRED(x)) < β(PRED(x))
#88:      β(PRED(x))
      If α(PRED(x)) > β(PRED(x))
      S(β(PRED(x)))
      0
```

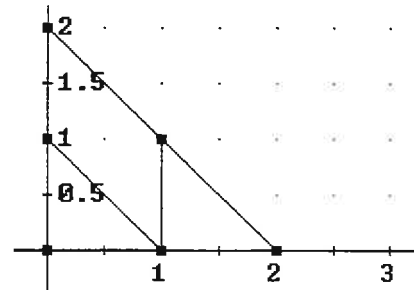
```
      β(x) :=
      If x = 0
      0
#89:      If α(PRED(x)) = β(PRED(x))
      S(α(PRED(x)))
      α(PRED(x))
```

```
#90:  ψ(x) := [α(x), β(x)]
```

```
#91:                                     ψ(10) = [3, 0]
```

We can try `VECTOR(ψ(n), n, 0, 5)` for obtaining the first six pairs on the list represented on the matrix and then plot them in the 2D-Plot Window.

$$\#93: \text{VECTOR}(\psi(n), n, 0, 5) = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \\ 0 & 2 \\ 2 & 0 \end{bmatrix}$$



Conclusion

There is a large number of applications for the functions defined in this paper. Many of them are very useful in trying to perceive the intuitive connection between Computable Functions and Recursive Functions specially for the student that faces for the first time a rigorous course about Formal Number Theory. At the end of the course everybody was more than happy and illustrated with using a simple but powerful tool in Mathematics work such as *DERIVE* in order to complement theoretical and practical concepts in one important branch of mathematics such as Recursion Theory.

Acknowledge

Special thanks to Dr. Iván Castro for encourage the using of *DERIVE* as fundamental part in mathematics courses in Javeriana University with his *Seminary on Mathematics Assisted by Computer* and also to Dr. Guillermo Arias for introducing me in studying Recursion Theory.

Biographic Details

Nelson Urrego P. is Professor of the Mathematics Department in Pontificia Universidad Javeriana in Bogotá, Colombia. His interest fields are Theory of Computation and Computing with DNA. Professor Urrego is candidate to PhD degree in Pedagogical Sciences in Enrique José Varona Pedagogic University in Havana, Cuba.

Nelson Urrego P.

Department of Mathematics

School of Sciences

Pontificia Universidad Javeriana

nurrego@javeriana.edu.co

Carrera 7 No 43-82 Edif. Carlos Ortiz, Oficina 512

Bogotá Colombia

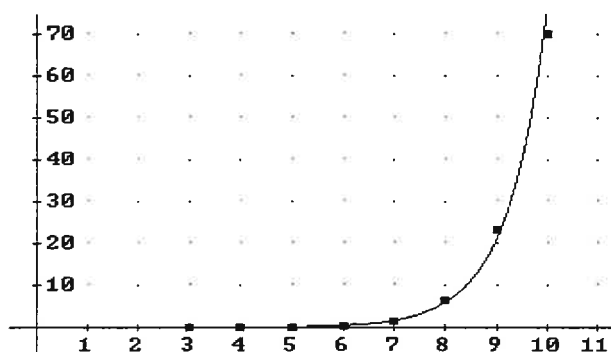
References

Boolos G. S. & Jefferey R.C. (1993) *Computability and Logic*. Cambridge University Press, 3th edition.

Castro I, (1992) *Cómo hacer Matemáticas con DERIVE*. Reverté. Bogotá.

Mendelson E. (1979) *Introduction to Mathematical Logic*. Van Nostrand, 3th edition.

Urrego N, (2000) *Some applications of Post and Turing Machines in Mathematics Teaching*. Proceedings of the 4th International Conference DERIVE TI 89/92, Liverpool John Moores University.



It is nice to plot Computation Time [sec] versus x of function $\psi(x)$ and find an appropriate regression line.

Exponential regression seems to fit:

$$1.2789 \cdot x \\ 0.00021482 \cdot e^x$$

Josef

Dynamic Algebra

Dr. René Hugelshofer, Heerbrugg, Switzerland

Cabri Geometre and Geometer Sketchpad showed us the astonishing effects that can be produced by dynamical change of the base objects. A classical problem leads to new perceptions and visual thinking (e.g. the way an ellipse can be transformed via a parabola to a hyperbola).

There is a lack of interesting problems for the use of CAS-calculators (CAS = Computer Algebra System). Parameterisation is an easy way to create attractive and challenging problems derived from classical ones. Such problems will lead students to a new way of thinking, similar to the geometrical approach described above.

I would like to demonstrate the method with the following examples. Examples of this kind can be found in any classical exercise book.

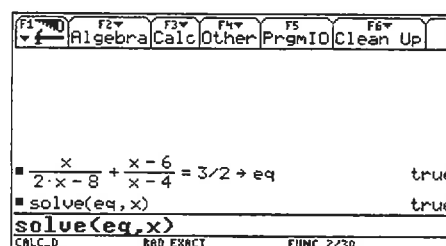
Classical problem 1: Fractional equation

Solve the equation: $\frac{x}{2x-8} + \frac{x-6}{x-4} = \frac{3}{2}$

Store the equation in eq and solve it with `solve(eq, x)`!

$$\text{SOLVE}\left(\frac{x}{2 \cdot x - 8} + \frac{x - 6}{x - 4} = \frac{3}{2}, x\right) = \text{true}$$

$$\text{SOLUTIONS}\left(\frac{x}{2 \cdot x - 8} + \frac{x - 6}{x - 4} = \frac{3}{2}, [x]\right) = [[e1]]$$



Result: TRUE. All $x \in \mathbb{R}$ are solutions, or nearly all. The CAS-calculator doesn't check the domain of the equation.

Dynamization of the classical problem

- a) Replace 4 with a ! Give an interpretation of the result!

The result is the Boolean expression $x = 6$ or $a = 4$ (means $x = 6$ or $a = 4$ and x arbitrary). Without first solving the classical problem, students can hardly understand it. The interpretation of such expressions helps to improve logical thinking.

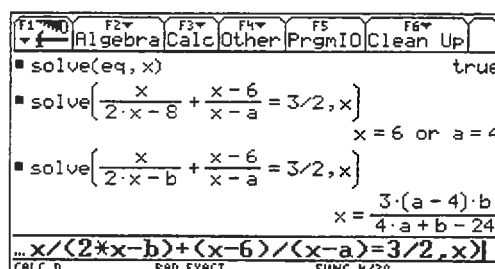
- b) In addition replace in (a) furthermore 8 with b !

The result $x = \frac{3b(a-4)}{b+4(a-6)}$ seems to be incomplete and leads to a discussion of the reliability of

the solver. Indeed for $a = 4$ the solution is 0 independent of b (compare with task (a)). This is a nice motivation to investigate the situation without technology.

CAS-calculators often give incomplete or even false solutions, especially with fractional equations.

DERIVE shows only x as result.



- c) Replace in the classical problem $\frac{3}{2}$ with $3!$

Result: FALSE.

If you solve the equation by hand you get $x = 4$. In this case the calculator considered the domain of definition of the equation.

$$\text{SOLVE} \left(\frac{x}{2 \cdot x - 8} + \frac{x - 6}{x - 4} = 3, x \right) = \text{false}$$

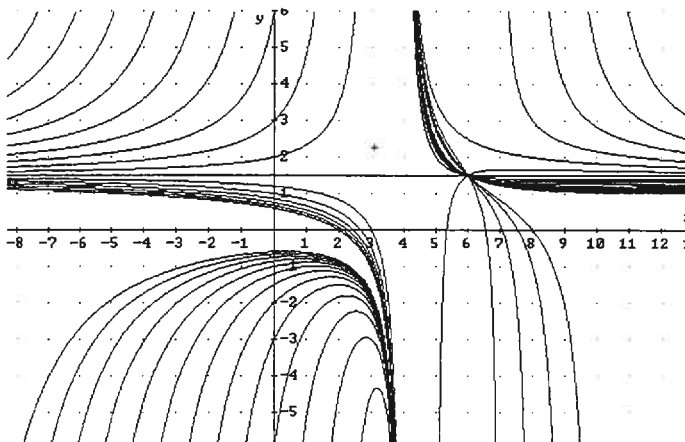
- d) In (c) replace 4 with $a!$

The result $x = \frac{5a + 4 \pm \sqrt{25a^2 - 248a + 592}}{6}$ is impressive and would be a hard piece of work without technology.

Further questions are possible: For which a will we have one, two or no solution(s)?

Result: $25a^2 - 248a + 592 = 0 \Leftrightarrow a = \frac{148}{25}$ or $a = 4$. As we know from (c) $a = 4$ isn't a solution. Once more the Solver arrives at an incomplete result.

- (e) When I read this example I had the idea to plot the family of curves resulting from task (a)'s right hand side taking a as parameter.



One could ask the students about a common property of all the curves and its connection to the solution of the equation.

Or give another task: Try to change the equation that you will have $x = 10$ independent on the choice for a !

I wrote this comment to René and his answer was very true: "Josef, as you can see, Dynamization has infected you!"

Classical problem 2: Heron's formula

(From Richard G. Brown: *Advanced Mathematics*, page 6, exercise 32)

Find the area of the triangle with vertices $A=(-13/2)$, $B=(5/17)$ and $C=(22/-4)$, using Heron's formula.

Solution: Define Heron's formula for the area of a triangle as a function:

$$\sqrt{s \cdot (s - a) \cdot (s - b) \cdot (s - c)} \quad | \quad s = \frac{a + b + c}{2} \rightarrow \text{heron}(a, b, c)$$

Test the Formula with $a = 3, b = 4, c = 5$!

Store the vertices: $[-13, 2] \rightarrow a, [5, 17] \rightarrow b, [22, -4] \rightarrow c$

`heron(norm(b-a), norm(c-b), norm(a-c));` gives the *Result*: 316.5

Note: The variables a, b, c and even s are local variables. They are not overwritten by the definition of the vertices.

The above exercise doesn't require a CAS-calculator. You can also do it with a small program.

Dynamization of the classical problem

- Is it possible to find a triangle with sides $x, x+1, x+2$ (arithmetical sequence) and area 10?
(*Result*: $x = 4.018$ or $x = -6.018$)
- What is the biggest (smallest) area, you can replace 10 in (a), to get a value of x ? (Individual exploration!). *Result*: It works for all areas!
- A "Heronic Triangle" is a triangle whose sides and areas are natural numbers. Find all heronic triangles whose sides are less or equal than 20.

Solution: Write a program to list all possible triangles.

You can imagine a lot of other questions, e.g. about isosceles or equilateral triangles or triangles with a given ratio of its sides.

(So the students experimented with `heron(x, x, 2x)` and investigated the factor c in

`heron(x, x, x) = c*heron(x, x, x/n)`
with $n = 1, 2, 3, \dots$)

I took René's challenge and wrote a DERIVE5 program which shows all possible true triangles. $(2, 2, 4, 0)$ is not on the list and each possible combination of sides has only one occurrence.

(What a "herioc" program for the "heronic" triangles!)

Josef

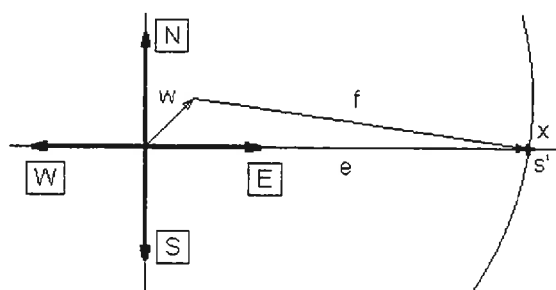
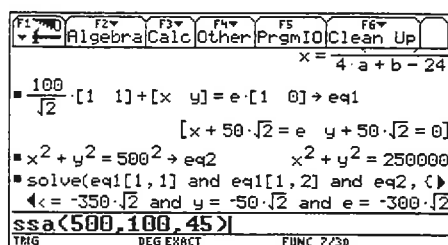
3	4	5	6
4	13	15	24
5	5	6	12
5	5	8	12
5	12	13	30
6	8	10	24
8	15	17	60
9	10	17	36
9	12	15	54
10	10	12	48
10	10	16	48
10	13	13	60
13	14	15	84
15	15	18	108
16	17	17	120

Classical problem 3: A Vector Equation

An aeroplane holds course east with an unknown effective velocity e . The wind blows in direction north-east with $w = 100$ mph. The aeroplane is flying with a velocity of $f = 500$ mph.

What is the maximum effective velocity of the plane and which course should the pilot hold? (flight direction vector $\vec{f} = [x, y]$)

Solution: You can take either a geometrical or an algebraic approach as follows.



As you can see in the edit line of the TI, the students can also take the trigonometric approach.

Result: $[x,y] = [350\sqrt{2}, -50\sqrt{2}]$, $e = 400\sqrt{2} \approx 565.7$ (and the second "solution in the opposite direction with $[x,y] = [-350\sqrt{2}, -50\sqrt{2}]$ and $e = -400\sqrt{2}$).

Dynamization of the classical problem

- a) The velocity of the wind and the plane will now be replaced by parameters w and f . As the wind force changes, the direction vector of the plane has to change, too, in order to hold a constant effective speed. The formulas will allow an automatic flight control.

Solution: Rewrite the two respective equations and solve again for x, y and e

$$\text{SOLVE} \left(\text{APPEND} \left[\frac{w}{\sqrt{2}} \cdot [1, 1] + [x, y] = e \cdot [1, 0], [x^2 + y^2 = f^2] \right], [x, y, e] \right)$$

$$\left[\begin{aligned} x &= \frac{\sqrt{2} \cdot \sqrt{(2 \cdot f^2 - w^2)}}{2} \wedge y = -\frac{\sqrt{2} \cdot w}{2} \wedge e = \frac{\sqrt{2} \cdot \sqrt{(2 \cdot f^2 - w^2)}}{2} + \frac{\sqrt{2} \cdot w}{2}, x = - \\ &\frac{\sqrt{2} \cdot \sqrt{(2 \cdot f^2 - w^2)}}{2} \wedge y = -\frac{\sqrt{2} \cdot w}{2} \wedge e = \frac{\sqrt{2} \cdot w}{2} - \frac{\sqrt{2} \cdot \sqrt{(2 \cdot f^2 - w^2)}}{2} \end{aligned} \right]$$

You can ask questions like: Which is the minimal flight velocity, allowing the pilot to hold the eastern course.

- b) A further generalization is to replace the wind direction vector $[1,1]$ by $[a,b]$. If you like you can describe $[a,b]$ by the angle of track.

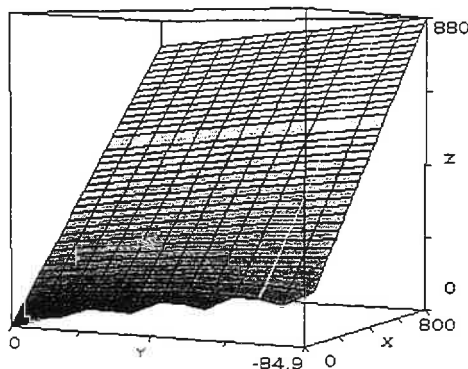
Solution: $[a,b] + [x,y] = e[1,0] \rightarrow \text{eq1, eq2 stays unchanged, } a^2 + b^2 = w^2 \rightarrow \text{eq3}$

Result: The students must analyse the solutions and choose the most appropriate one. This should improve their abstract thinking ability.

$$\begin{aligned} &\text{solve}(a+x=e \text{ and } b+y=0 \text{ and } x^2+y^2=f^2 \Rightarrow \\ &x = -\sqrt{f^2 - b^2} \text{ and } y = -b \text{ and } a^2 + b^2 = w^2 \Rightarrow \\ &w = \sqrt{a^2 + b^2} \text{ and } e = a - \sqrt{f^2 - b^2} \text{ or } x = \end{aligned}$$

(Caution: the output depends on the "mood" of the calculator. It can also be given in a much bulkier form!)

Students are now motivated to start a flight simulation project (at least in an ideal world).



Another idea is to interpret the solution of task a) as parameter representation of a surface with parameters f and w . Plot this surface with DERIVE (or DPGraph).

Can you find the point belonging to solution (a)?

Are the parameter lines really straight lines (as they are looking like)?

Classical problem 4: Transversals

Given are two skewed straight lines g and h . Insert a horizontal transversal of length 5! At which height do you have to place the transversal?

$$g: (x/y/z) = (-8/5/7) + t(-6/3/4)$$

$$h: (x/y/z) = (0/1/-5) + s(1/2/4)$$

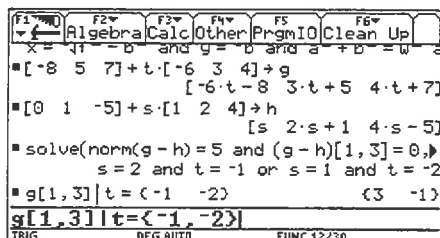
Solution: Store the above vector equations of the lines in g and h !

The problem can be expressed by two conditions:

(I) $|\overline{GH}| = 5$ (CAS: $\text{norm}(g-h) = 5$)

(II) The 3rd component of \overline{GH} is 0 (CAS: $(g-h)[1,3] = 0$)

The solutions to the system of equations (I) and (II) are given in the figure.



Calculation with paper and pencil is rather hard work. However it requires only basic algebraic manipulations, which reduce the problem to a quadratic equation. With the calculator it can be done by pressing a few keys.

Dynamization of the classical problem

This problem can be generalized using the same equations but replacing values by parameters, e.g.:

- a) Replace the length 5 in condition (I) by a parameter c . What is the shortest horizontal connection

between the straight lines g and h ? *Result:* $c = \frac{5\sqrt{2}}{2}$

- b) Omit condition (II) in the classical problem! The transversal of length 5 glides on the two straight lines. What is the possible range of the variables t and s for a solution?

Result: The extreme values are $t = \frac{5(-12 \pm \sqrt{21})}{41}$ and $s = \frac{48 \pm 5\sqrt{61}}{41}$

- c) Omit condition (II) again, but replace 5 with c (as in a)). Determine the endpoints and the length of the shortest distance between g and h ! Can you find another approach as given in b)?

Result: Shortest distance $c = \frac{20\sqrt{41}}{41}$ for $t = -\frac{300}{205} = -\frac{60}{41}$ and $s = \frac{240}{205} = \frac{48}{41}$

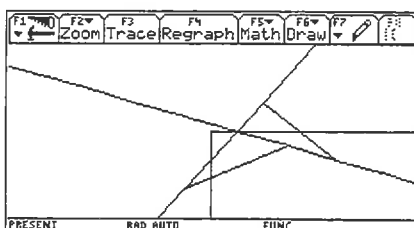
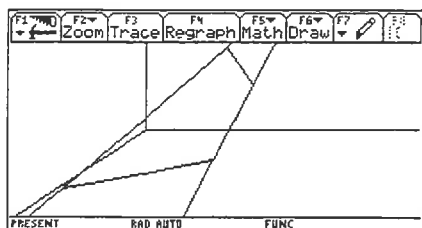
This is just the mean value of the extrem values of the solutions found in b).

Now substitute for s and t in the vector equations of g and h !

- d) Replace in the classical problem one component of the first vector in g by a parameter c (e.g. c instead of 7). Find the value of c , for which g and h intersect!

Solution: $\text{solve}((g-h)[1,1]=0 \text{ and } (g-h)[1,2]=0, \{s, t\})$ *Result:* $s=0$ and $t=-4/3$
 $(g-h)[1,3]=0 | s=0 \text{ and } t=-4/3 \text{ gives } c = 1/3$

- e*) Project for interested students: In problem b) vary parameter t within the possible range. Describe the other endpoint of the gliding transversal using the same parameter (two possibilities). Visualize the gliding transversal geometrically (by programming or animation in a geometry program). Or produce a graphic representation of any solution using a tool like $\text{pres}()$ *



An Oblique View and a Top View.

* On request from the Editor

Classical problem 5: Apollonius sphere

Given are the points $A=(4/-5/11)$, $B=(1/-0.5/2)$ and the line $l: (x/y/z) = (0/-13/0) + t (3/8/1)$.

Find point P on the line l , so that $|AP| : |BP| = 2:1$!

Solution: The problem can be expressed by two conditions:

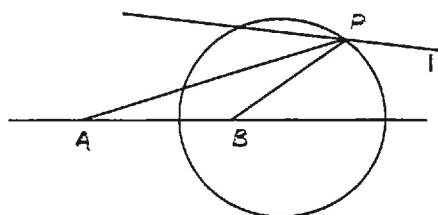
$$(I) \quad \frac{|AP|}{|BP|} = 2 \quad (II) \quad P \in l$$

First store the points A and B as a and b and the line as l

Solve equation (I): `solve(norm(l-a)/norm(l-b) = 2, t)`. (Result: $t = 1$ or $t = 2$)

Substitute for t in l : $1 \mid t = \{1, 2\}$

Solutions: $P_1 = (3/-5/1)$, $P_2 = (6/3/2)$



Dynamization of the classical problem

Again we generalize by replacing given constants by parameters e.g.:

a) Replace l by c in the direction vector of the line l .

For which parameters c will exist one, two or no solution(s)?

Give a geometrical interpretation!

Result: Solution of the equation I: $t_{1,2} = \frac{c - 112 \pm \sqrt{-147c^2 - 224c + 17}}{c^2 + 73}$

One solution: $c = 2.759$ or $c = -4.290$

Two solutions: $-4.290 < c < 2.759$

No solution: $c < -4.290$ or $c > 2.759$

b) Replace in the line l of (a) the vector $(0/-13/0)$ by $(0/1/-1)$ and give a geometrical interpretation!

Result: $t_{1,2} = \frac{\pm 7}{\sqrt{c^2 + 73}}$ (2 solutions, symmetrical with respect to the point $(0/1/-1)$)

Interpretation: $(0/1/-1)$ must be the center of the sphere.

c) Replace in the original classical problem the vector $(0/-13/0)$ by $(0/30/c)$.

Result: No solution! (The set of lines given by c never intersects the sphere.)

d) Let us omit condition (II) in the original problem

Result: Equation of the Apollonius sphere $x^2 + (y-1)^2 + (z+1)^2 = 49$ (compare with problem (b)).

You can also introduce parameters in A and B and make investigations on the harmonic points A , B and the intersection point of the Apollonius sphere with l

Classical problem 6: Center of a circle

Find the circumscribed circle for the points $A=(-3/1)$, $B=(1/-1)$, $C=(3/-2)$.

Solution: The problem can be expressed by two conditions:

$$(I) \quad |AM| = |BM| \quad (II) \quad |BM| = |CM|$$

```

■ solve(norm(m-a)=norm(m-b) and norm(m-b)
x = 5689913.3858 and y = 11379827.2869
...d norm(m-b)=norm(m-c),{x,y}}
Warning: More solutions may exist

```

```

■ solve((norm(m-a)=norm(m-b))^2 and (norm(m-b)=norm(m-c))^2,{x,y})
false
Warning: Operation might introduce false solutions

```

The first results for x and y are a very large numbers and the calculator gives the warning: "More solutions may exist". The two equations (I) and (II) are containing roots, which can be eliminated by squaring: Then we get the correct result "false", which means that there is no intersection point at all. The points are on a straight line.

```

F1 F2 F3 F4 F5 F6
Algebra Calc Other PrgmIO Clean Up
[3 -2.1] → C [3 -2.1000]
■ solve(norm(m-a)=norm(m-b) and norm(m-b)
x = 947731942395. and y = 1.8599e12
■ (norm(m-a)=norm(m-b) and norm(m-b)=norm(m-c))
x = 2.1302e12 and y = 4.2527e12
■ solve((norm(m-a)=norm(m-b))^2 and (norm(m-b)=norm(m-c))^2,{x,y})
x = -1581/40 and y = -1541/20
...norm(m-b)=norm(m-c))^2,{x,y}}
PRESENT RAD EXACT FUNC 23/30

```

Replacing -2 with -2.1 in C leads to an intersection point.

But the result depends on the calculation mode!

Although the last result seems to be correct, the calculator gives the warning: "... might introduce false solutions"

Dynamization of the classical problem

- a) Replace -2 in C with a constant k and calculate the center of the circle again!

Result: $x = \frac{k^2 - 4k + 3}{4(k+2)}$ and $y = \frac{k^2 + 11}{2(k+2)}$; Case $k=-2$ is not dealt with by the calculator.

Note: In this problem it would probably be better to start with $C=(3/k)$ and to ask the students to analyse special cases including geometrical interpretation.

- b) Which interpretation can be given for $(\text{norm}(m-a))^2 - (\text{norm}(m-b))^2 = 0$?
(and for $(\text{norm}(m-b))^2 - (\text{norm}(m-c))^2 = 0$) \rightarrow *perpendicular bisectors of \overline{AB} and \overline{BC}* .
- c) Introduce a factor f in condition (I): $|AM| = f|BM|$. The equation (I) represents an Apollonius circle as in the problem above.

Result in simplified form: $(1-f^2)x^2 + (2f^2+6)x + (1-f^2)y^2 + (-2f^2-2)y - 2f^2 + 10 = 0$

It is a rewarding exercise to transform this equation to the normal form (with or without technology):

$$\left(x + \frac{f^2+3}{1-f^2}\right)^2 + \left(y - \frac{f^2+1}{1-f^2}\right)^2 = \frac{20f^2}{(1-f^2)^2} = \text{radius}^2$$

You can ask questions like: What happens with the center and the radius, if f tends to 1 or 0?
Describe a straight line as limit of a circle.

You can also generalize the classical problem in three dimensions: Given four points A, B, C and D . Calculate the center M and radius of the sphere through the given points.

Final note:

This method of dynamization by parametrization works with about one third of all classical mathematical problems and gives us a quick way to produce problems for CAS calculators.

The method shows a way to progress from skill-oriented problems to thinking-oriented problems.

Parallelkurven und Flatterbandkurven

Josef Rolfs, Friesoythe, Josef.Rolfs@t-online.de

Zu Pfingsten im Jahre 1977 hörte ich auf der ersten T³-Pfingsttagung in Münster einen Vortrag von Herrn Wunderling zu Parallelkurven. Seine Überlegungen wurden im DNL #31, p31-p34, mithilfe von *DERIVE* veröffentlicht. In meinem Mathematikunterricht (Leistungskurs) hatten alle Schüler(innen) den TI 92 zur Verfügung, und so stürzte ich mich zur Vernetzung von Analysis und Analytischer Geometrie auf dieses Thema. Dabei standen immer parametrisierte Darstellungen im Vordergrund.

Jede Funktion f mit dem Term $f(x)$ und $D(f)$ kann parametrisiert werden durch

$$\vec{x}(t) = \begin{pmatrix} t \\ f(t) \end{pmatrix} \text{ mit } t \in D(f).$$

Ein Tangentenvektor, der in die Richtung steigender t -Werte („nach rechts im üblichen x - y -Koordinatensystem“) weist, ergibt sich aus der Ableitung

$$\dot{\vec{x}}(t) = \begin{pmatrix} 1 \\ f'(t) \end{pmatrix} \text{ mit } t \in D(f).$$

Somit ergibt sich ein Normalenvektor, der „nach oben“ weist, aus

$$\vec{n}(t) = \begin{pmatrix} -f'(t) \\ 1 \end{pmatrix} \text{ mit } t \in D(f).$$

Es folgt für den Normaleneinheitsvektor, der „nach oben“ weist, die Darstellung

$$\vec{n}_e(t) = \frac{\vec{n}(t)}{|\vec{n}(t)|} \text{ mit } t \in D(f).$$

Es lassen sich also die Parallel- oder Abstandskurven vektoriell beschreiben durch

$$\vec{x}_p(t) = \vec{x}(t) + d \cdot \vec{n}_e(t) \text{ mit } t \in D(f)$$

Für $d > 0$ verlaufen sie hier oberhalb der vorgegebenen Kurve und für $d < 0$ unterhalb.

In 1977 at the first T³-Whitsun Conference in Münster I heard W. Wunderling's talk on Parallel Curves. He published his ideas in DNL#31. In my math teaching (performance course) all students had a TI-92 at their disposal. So I took this object to combine Calculus and Analytical Geometry. In all our considerations we focused on parameterized representations.

In the boxes given above one can find parameter representations of a function $f(x)$, its tangent vector (which shows to the right for increasing t -values), its perpendicular vector (which shows in the "upper direction"), its perpendicular unit vector and finally using all these vectors you can give the vector definition of Parallel or Distance Curves. (For $d > 0$ they run above the given curve, for $d < 0$ they run below.)

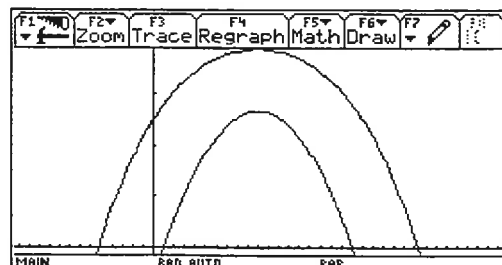
Bsp: Darstellung für die Kurve mit dem Term $f(x) = \frac{1}{3}(x-5)^2 + 4$

Parallelkurve im Abstand 2

Parallel Curve for distance $d = 2$

Fig. 1: Graph von f mit Parallelkurve ($d=2$)

Fig. 1: Graph of f with Parallel Curve ($d = 2$)



Auf der Krümmungskreisseite der Kurve kann es zu Sonderformen („Rückläufern“) kommen. Hier müssen Teile der sich ergebenden Kurve herausgenommen werden, um noch Abstandskurven zu erhalten. Der „Restbogen“ weist jedoch dann einen Knick auf.

Special forms can appear on the side of the osculation circle ("Back Runners"). Parts of the resulting curve must be removed to obtain Distance Curves. Then the remaining rest shows a sharp turn.

Abstandskurven im Abstand 2 (Distance Curves with $d = 2$)

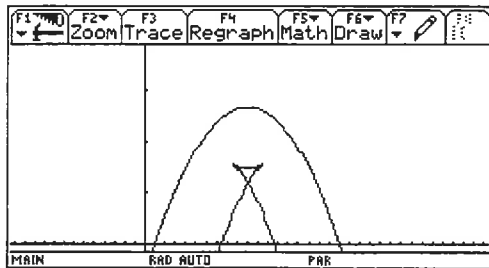


Fig. 2: Graph von f mit Parallelkurve ($d = -2$)

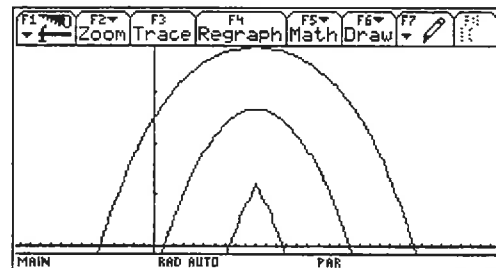


Fig. 3: Graph von f mit Abstandskurven ($d = 2$ und $d = -2$)

Aus der obigen Frage entwickelte sich im Anschluss eine Untersuchung der Parallel- oder Abstandskurven bei der Normalparabel, um diese Sonderformen deuten zu können. „Parallelkurven“ gibt es nur auf der krümmungskreisfreien Seite der Normalparabel und auf der Krümmungskreisseite für $d \leq r_{\min} = 1/k_{\max} = 1/2$ mit $k_{\max} = k(0) = 2$. Für alle anderen „Abstandskurven“ ergeben sich Sonderformen mit Spitzen. Diese liegen auf der **Evolute** der Ausgangskurve, die hier eine semikubische oder auch Neillsche Parabel ist.

The question from above led to an investigation of the Parallel- or Distance Curve of the parabola $y = x^2$ to find an interpretation of the special forms:

„Parallel Curves“ only exist „outside“ of the curve and „inside“ only for

$$d \leq r_{\min} = 1/k_{\max} = 1/2 \text{ with } k_{\max} = k(0) = 2$$

*For all other Distance Curves we receive special forms containing one or more sharp bends, which are lying on the **evolute** of the given function graph (which is in our case the Semicubic or Neill's Parabola)*

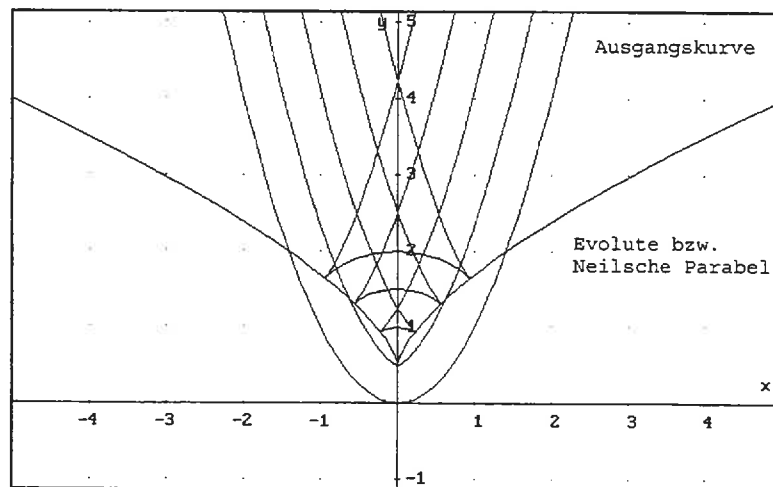


Fig. 4

Für die Parabel mit der Gleichung $y = x^4$ ergeben sich ähnliche Sonderformen. Auch hier liegen die Spitzen auf der zugehörigen **Evolute** der Parabel. Rechnerisch ergibt sich hier

$$r_{\min} = \frac{1}{k_{\max}} = \frac{9}{98} \sqrt[5]{7^5} \approx 0.464799.$$

Figure 5 shows $y = x^4$ together with its evolute and some Parallel Curves.

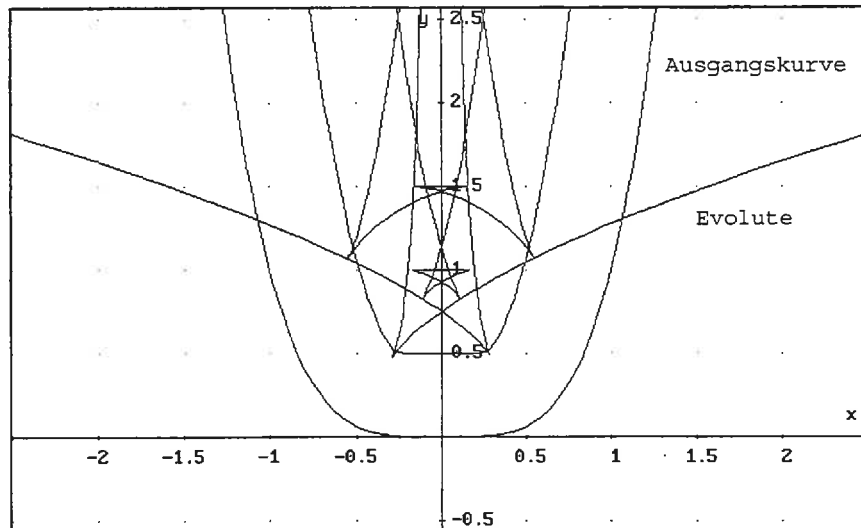


Fig. 5

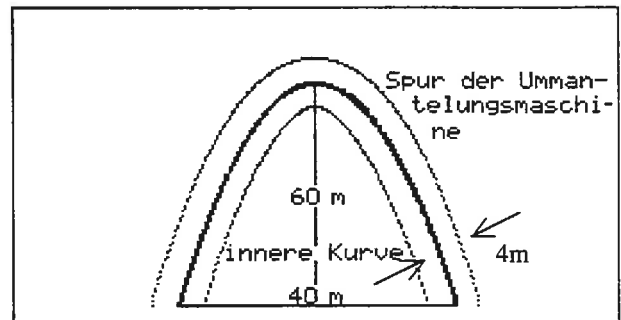
Abituraufgabe:

Wahrzeichen in Parabelform

Endexamination task:

A Parabola as a Landmark

Das Wahrzeichen einer Stadt soll ein 60m hoher und am Fuße 40m breiter Parabelbogen werden. Um ihn herzustellen, wird ein Stahldraht in die Parabelform gebogen und dann mit einem 2m dicken Kunststoffmantel umhüllt. Zum Ummanteln wird dazu ein Zylinder mit Radius 2m genau in 2m Abstand am Draht entlang geführt und flüssiger Kunststoff eingefüllt, der sich sofort verfestigt.



Wähle ein geeignetes x - y -Koordinatensystem und ermittle den Parabelterm $f(x)$, der den Stahldrahtbogen beschreibt. Ermittle in den Schnittpunkten mit der x -Achse die Tangenten- und Normalengleichung. Wie lautet in einem beliebigen Punkt $P(t/f(t))$ des Bogens die zugehörige Normalengleichung?

- Bestimme die Ortsvektoren der Kurve, die die Ummantelungsmaschine steuert.
- Für die Optik ist auch die innere Kurve wichtig. Wie lauten die Ortsvektoren der Kurvenpunkte?
- Ein anderer Architekt möchte den Draht mit einem dickeren Mantel umgeben. Leite in Abhängigkeit vom Zylinderradius d die Ortsvektoren der inneren Kurvenpunkte her. Für welche $d > 0$ wird die y -Koordinate der inneren Kurve genau einmal maximal? Deute das Resultat im Zusammenhang mit der Maximalkrümmung der Ausgangskurve.

The government of a town intends to erect a new landmark in form of a parabolic bend with 60m height and a base of 40m. To produce this monument a steel wire is bent which will be surrounded by a coat of synthetic material with 2m diameter. For jacketing the wire a cylinder of radius 2m will glide in a distance of 2m along on the steel wire and it will be filled with immediately hardening liquid synthetic material.

- Choose an appropriate x-y-coordinate system and find the function term describing the parabola. Find the equations of the tangents and the perpendicular lines in the zeros. What is the equation of the perpendicular line in an arbitrary point $P(t / f(t))$ of the graph?
- Find the position vector of the curve steering the coating machine.
- What is the parameter form of the "inner curve"?
- Another architect prefers a thicker surrounding. Derive the parameter form of the inner curve in dependence of the cylinder radius d . Which $d > 0$ gives a unique maxim y-coordinate of the inner curve? Give an interpretation of the result in connection with the maximum bend of the parabola.

In der nächsten Aufgabe werden diese Überlegungen wieder aufgegriffen, und zwar im Zusammenhang mit den sogenannten Flutterbandkurven der Länge l , die vektoriell beschrieben werden durch:

$$\vec{x}_f(t) = \vec{x}(t) - l \cdot \frac{\dot{\vec{x}}(t)}{|\dot{\vec{x}}(t)|}, t \in D(f).$$

In the next task we will come back to these considerations in connection with so called "Flutter or Flighty Ribbon Curves of length l , which have a vector form:

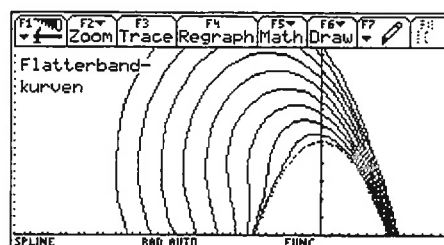


Fig. 6 Eine Schar von Flutterbändern

Problem: "Flutterband" "Flutter Ribbon"

Ein Tangentenstück der Länge l gleitet auf der Parabel $y = x^2$ entlang. Untersuchen Sie die Kurve, die der Endpunkt der Strecke durchläuft.

A tangent segment of length l glides along the parabola $y = x^2$. Find the locus of the endpoint of the segment.

Lösung: Ortsvektor zum Parabelpunkt P (position vector of P) $\vec{p} = \begin{pmatrix} t \\ t^2 \end{pmatrix}$

Tangentenvektor in P (tangent vector in P) $\vec{t} = \begin{pmatrix} 1 \\ 2t \end{pmatrix}$

Tangenteneinheitsvektor in P (unit tangent vector in P) $\vec{t}_e = \frac{1}{\sqrt{1+4t^2}} \begin{pmatrix} 1 \\ 2t \end{pmatrix}$

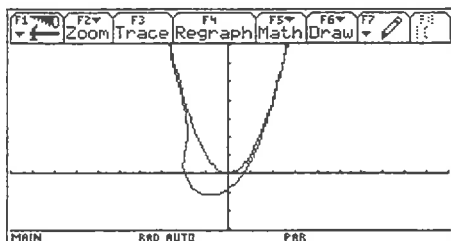
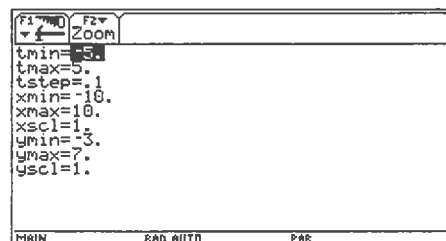
Hat das Tangentenstück die Länge l , so ergibt sich für den Ortsvektor \vec{x} des "Flutterbandes" im Punkt X die **Vektor-** und weiters die **Parameterdarstellung**

The **vector form** and the **parameter form** of the "Flighty Ribbon" with a ribbon of length l are given by:

$$\vec{x}(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \vec{p} - l \cdot \vec{t}_e = \begin{pmatrix} t \\ t^2 \end{pmatrix} - \frac{l}{\sqrt{1+4t^2}} \begin{pmatrix} 1 \\ 2t \end{pmatrix}$$

$$x(t) = t - \frac{l}{\sqrt{1+4t^2}} \quad \text{und} \quad y(t) = t^2 - \frac{2lt}{\sqrt{1+4t^2}}$$

Über die Befehlsfolge **[MODE]**, **PARAMETRIC** **[ENTER]**, **[ENTER]** werden nun im **[Y=]**- Editor die Koordinatengleichungen eingegeben.

Fig. 7 Flutterband für $l = 2$ 

Windows - Einstellungen

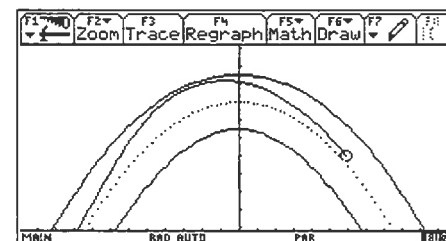
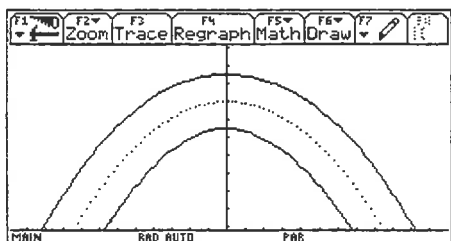
Kombinierte Aufgabe: Straßenprobleme

Combined Task: Roadproblems

Gegeben sei die Mittellinie einer Straße durch eine Funktion $f(x) = 50 - \frac{x^2}{20}$

Die 6 m breite Allee wird im Abstand von 3m von der Mittellinie durch zwei Baumreihen begrenzt.

Ein Lkw hat einen starren T-Träger zu transportieren, der über die hintere Anhängerachse 8m mittig hinausragt. Der Mittelpunkt der hinteren Anhängerachse bewegt sich dabei direkt über der Mittellinie der Straße. Somit beschreibt der Trägerendpunkt eine Flutterbandkurve



Man erkennt hier, dass der Schwertransport „die Kurve schafft“. Es stellte sich hier sofort die Frage nach der maximalen T-Trägerlänge, mit der diese Kurve noch passiert werden kann. Diese Frage konnten wir nur graphisch bzw. approximativ lösen. Gibt es hier auch eine algebraische oder geometrisch-konstruktive Lösung?

Eine andere Frage, die sich stellte und mit oben verwandt ist, ist diese:

Welcher Punkt der „Flutterbandkurve“ hat von der „Trägerkurve(Ausgangskurve)“ den größten Abstand und wie groß ist dieser?

The center line of a road is given by the function $f(x) = 50 - \frac{x^2}{20}$. The road is bounded by two rows of trees in a distance of 3m of the center line.

A truck has to transport a T-beam which exceeds the truck's back side by 8m. The center of the truck's rear axle moves along the center line of the road. Can the truck pass the bend in the road without having contact with the trees?

One can recognize by inspection only that the truck masters the transport. Immediately the question for the maximum possible length for the beam was posed. We were only able to find an answer graphically and numerically. Is there a algebraic and/or geometric solution?

Another question, which is related to the above one is:

Which point of the flutter curve has the maximum distance from the center line and how much is it?

(Solution for the end examination task "Landmark" can be found in: Knechtel & Weiskirch, Abituraufgaben mit Graphikrechner und Taschencomputern, Teil 1, Schroedel Verlag Hannover 2001)

Mistakes in Representing Graphs on the TI-92? Rounding Errors with TI-InterActive?

Thomas Himmelbauer, Vienna

Let's take the following "nice" function $y_1(x)$:

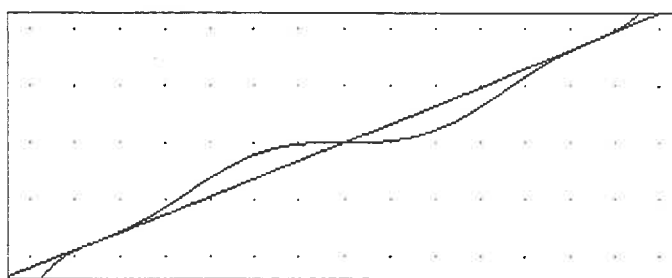
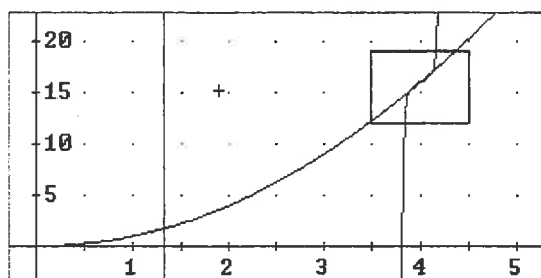
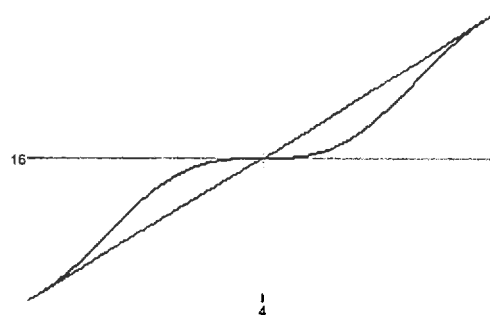
$$\begin{aligned} & \frac{750000000000}{1771561} \cdot x^8 - \frac{16000000000000}{1771561} \cdot x^7 + \frac{111969750000000}{1771561} \cdot x^6 + \frac{36000000000}{14641} \cdot x^5 - \\ & - \frac{4481451597372500}{1771561} \cdot x^4 + \frac{28664253071800000}{1771561} \cdot x^3 - \frac{85946307513840000}{1771561} \cdot x^2 + \\ & + \frac{130886209591680000}{1771561} \cdot x - \frac{81746656184775024}{1771561} \end{aligned}$$

That nobody takes care about my mental condition I'd like to explain how I got this function. I wanted to build a polynomial function which deviates in a small region around $x = 4$ from the standard parabola $y = x^2$. The deviation should be so small that in standard settings for the plot windows one couldn't recognize a difference in the graphs, but the 1st derivatives of both functions should be very-different. So I chose:

$$\begin{array}{lll} y_1(4) = 16 & y_1(3.89) = 3.89^2 & y_1(4.11) = 4.11^2 \\ y_1'(4) = 0 & y_1'(3.89) = 2 \cdot 3.89 & y_1'(4.11) = 2 \cdot 4.11 \\ y_1''(4) = 0 & y_1''(3.89) = 0 & y_1''(4.11) = 0 \end{array}$$

I solved the system with *DERIVE* and plotted the resulting function together with the parabola with TI-InterActive (right).

See below the two graphs of $y_1(x)$ plotted with *DERIVE5* using different scales:

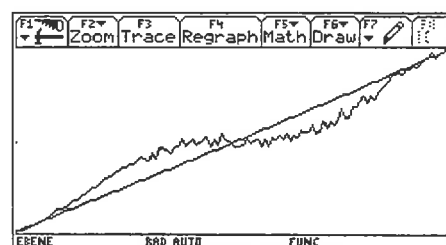


Now compare the plot on the TI-92:

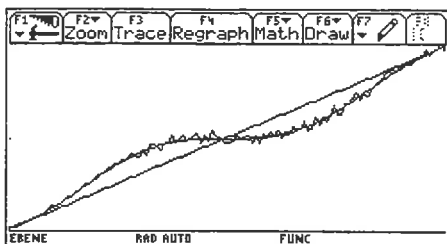
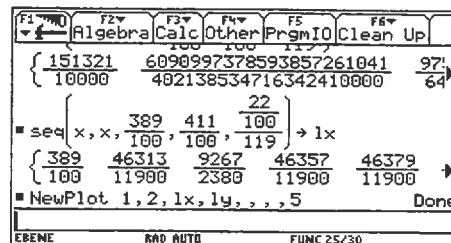
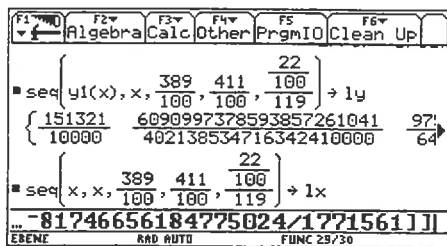
So we could raise the question how such a presentation can be possible?

First of all let's inspect the WINDOW-settings:

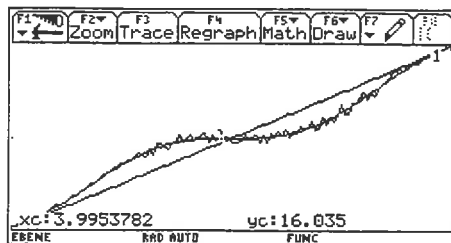
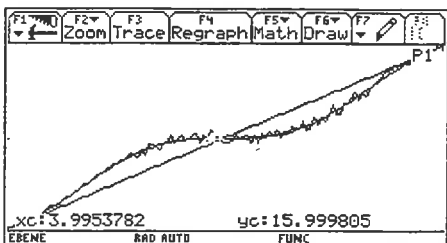
```
xmin=3.89
xmax=4.11
xscl=16.
ymin=15.1321
ymax=16.8921
yscl=16.
xres=1.
```



I checked the function values, producing the respective lists and plotted the results.

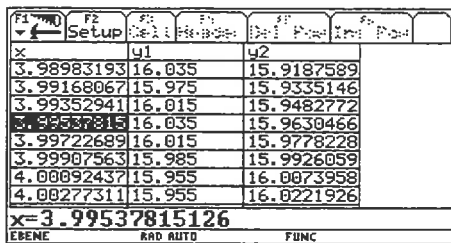
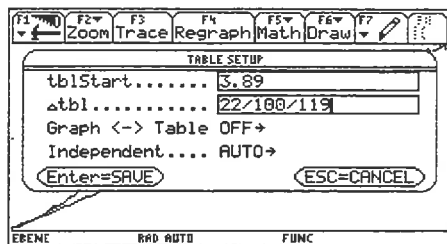


As you can see, the plotted points connected are showing the graph correctly.

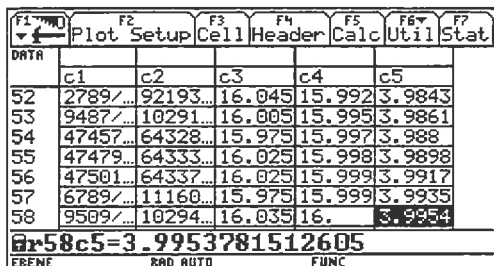


It is interesting to compare the function values of the $y_1(x)$ and of the plot for $x = 3.9953782$: 15.999805 (plot) versus 16.035 (function). Is the mistake caused by incorrect roundings?

I became curious and inspected the table: The table values are matching the function values of the function graph.



Finally I produced value tables using the Data/Matrix Editor:



In column c1 you can find the exact x-values:

$$c1 = \text{seq}(x, x, 389/100, 411/100, 22/11900),$$

in c5 the approximated column c1:

$$c5 = \text{approx}(c1)$$

c2 shows the exact function values:

$$c2 = y1(c1)$$

c3 shows the function values of the approximated x-values from c5: $c3 = y1(\text{approx}(c1))$ – here I found 16.035 – and in column c4 I show the approximated exact function values from c2: $c4 = \text{approx}(c2)$. And in this column you can see 15.99804621835 (data for the plotted points)!

p38	FACTORS, FACTORS & many messages	D-N-L#43
-----	----------------------------------	----------

According to the TI-92 the exact value for $x = 3.99537815112605$ $x = \frac{9509}{2380}$ and its exact function value is $\frac{10294620778195837345}{643421655546147856}$ with its approximation 15.9998046218. This is corresponding with the value belonging to the plot of the two lists.

I checked these values with *DERIVE5* and TI-InterActive as well.
The results are collected in the following summary.

	TI-92	TI-InterActive	Derive 5
y1(9509/2380)	$\frac{10294620778195837345}{643421655546147856}$	$\frac{10294620778195837345}{643421655546147856}$	$\frac{10294620778195837345}{643421655546147856}$
approx(y1(9509/2380))	15.9998046218	15.99974823	15.99980462
y1(approx(9509/2380))	16.035	15.99974823	15.99980462

Whereas in *DERIVES* and in TI-InterActive the order of the approximation process does not influence the result (correct plotting of the function graph) this is not so on the TI-92.

Finally we must notice that *DERIVE* / TI-92 and TI-InterActive are producing show different rounded results.

Factors, Factors and many messages in the *DERIVE* News

From Philippe (Belgium):

Hello, How can I put in a variable the last factor (without exponent) of a factorisation?

*484 should give 11 ($484 = 2^2 * 11^2$)*

Regards, Philippe.

From Johann (Austria):

Johann Wiesenbauer had a very short reply:

The following function

`Philippe(n) := firstfirstreversefactorsn`

should do the trick for any $n > 1$.

Cheers, Johann

From Wim (England):

Despite the bracketing, I'm still not getting anywhere with it. Is the age of my version of *DERIVE* (4.0) to blame? It does not recognise the "first" function.

From Ignacio (Spain):

Yes, it is. I have version 5 and 4.11. V4.11 doesn't recognise "first" neither "reverse".

From Wim again:

Philippe,

Am I correct in thinking that you are looking for a function $f(n)$ which will give you the largest prime factor of any natural number > 1 ? If so, the following will do:

$$g(n) := (\text{SELECT}(\text{PRIME}(k) \wedge \text{MOD}(n, k) = 0, k, n, 2, -1))_1$$

$$g(484) = 11$$

$$g(48434) = 397$$

I am not getting anywhere with Johann's expression and wonder if it has to be amended.

Finally from Aleksey (Russia?)

Hi,

All Derivers!

On reading very interesting mails (Factors etc) I propose the function emulating the FACTORS function of *DERIVE5xx*: It works with *DERIVE* 3.11 and later.

$\text{FACTORS5}(n) := \text{VECTOR}([\text{EXP}(j \text{ SUB } 1), j \text{ sub } 2], j, \text{FACTORS}(\text{FACTOR}(\text{VECTOR}(\text{LN}(k), k, \text{FACTORS}(\text{FACTOR}(n))))))$

$$\text{FACTORS5}(484) = \begin{bmatrix} 2 & 2 \\ 11 & 2 \end{bmatrix}$$

It is again Johann (Austria):

Nice,

unfortunately it will fail, if n has a prime factor of multiplicity 1, which is extremely likely for big numbers (try $n = 22$, instead of $n = 484$). Hence, you should use the following modification:

$\text{FACTORS6}(n) := \text{VECTOR}([\text{EXP}(j \text{ SUB } 1), j \text{ SUB } 2/2], j, \text{FACTORS}(\text{FACTOR}(\text{VECTOR}(\text{LN}(k), k, \text{FACTORS}(\text{FACTOR}(n^2))))))$

$$\text{FACTORS6}(22) = \begin{bmatrix} 2 & 1 \\ 11 & 1 \end{bmatrix}$$

$$\text{FACTORS6}(484) = \begin{bmatrix} 2 & 2 \\ 11 & 2 \end{bmatrix}$$

Now it's my turn, Josef:

Preparing this DNL I found out that $\text{FACTOR6}(n)$ does not work with *DERIVE5*. So I tried to adapt it for the "5", too. It is an example how to change *DERIVE4* solutions for *DERIVE5*. In some cases routines treating vectors and matrices must be changed.

I'd like to direct your attention how Johann edited his function – without typing the parantheses for the functions. *DERIVE5* is clever enough to set the parantheses by its own!!

$\text{FACTORS6_}(n) := \left[\text{VECTOR} \left(\text{EXP}(j_{1,1}), j, \text{FACTORS}(\text{FACTOR}(\text{VECTOR}(\text{LN}(k), k, \right.$

$\left. \text{FACTORS}(\text{FACTOR}(n^2)))) \right] \text{DIM}(\text{FACTORS}(\text{FACTOR}(n))), 1$

$$\text{FACTORS6_}(484) = 11$$

$$\text{FACTORS6_}(22) = 11$$

$$\text{FACTORS6_}(48434) = 397$$

Titbits from Algebra and Number Theory (20)

by Johann Wiesenbauer, Vienna

In my last column I dealt with the set S_n of permutations on the set $\{1, 2, \dots, n\}$ for some natural number n . In particular, I introduced a number of useful operations for these sets S_n , which form groups w.r.t. the composition of functions that play an important role in many applications both in mathematics and natural sciences. In fact, those symmetric groups and their subgroups can simplify things a lot, if the objects under consideration show any symmetries.

Before giving an example of this sort (from recreational mathematics, for a change!), let me list all functions from my last contribution that will be used in the following along with some supplements. (This will by the way also give me the opportunity to remove some small, but harmless "inconsistencies" in those listings.)

The first two transform a permutation of S_n given in standard notation into the disjoint cycle notation and vice versa.

```

toperm(p, n) :=
  Prog
  If p = []
    RETURN [1, ..., n]
  p := APPEND(VECTOR(APPEND(u_, FIRST(u_), 0), u_, p))
  p := SELECT(II(p_) > 0, p_, [DELETE(p, -1), REST(p)]^1)
  Loop
  If n = 0
    RETURN SORT(p)^12
  If  $\neg$  MEMBER?(n, FIRST(p^1))
    p := ADJOIN([n, n], p)
  n := -1

tocycles(p, c_, p_ := [], s_) :=
  Prog
  If VECTOR?(FIRST(p))
    p := p^12
  s_ := {1, ..., DIM(p)}
  Loop
  If s_ = {}
    RETURN REVERSE(p_)
  c_ := ITERATES(p^1k_, k_, MIN(s_))
  If DIM(c_) > 2
    p_ := ADJOIN(DELETE(c_, -1), p_)
  s_ := s_ \ MAP_LIST(c_^1k_, k_, {1, ..., DIM(c_)})

```

And here a small example:

```
toperm([[1, 5, 4], [3, 6]], 7) = [5, 2, 6, 1, 4, 3, 7]
```

```
tocycles([5, 2, 6, 1, 4, 3, 7]) = [[1, 5, 4], [3, 6]]
```

Let's make a convention for the following. Whenever a function dealing with permutations expects the input in disjoint cycle notation we will use only small letters for its name and only capital letters otherwise. The same goes for the output as regards permutations. Here is the first example of this sort which defines the composition of two permutations.

```
MULPERMS(a, b) := VECTOR(a
                        b^1k_
                        , k_, 1, DIM(a))
```

```

mulperms(a, b, n_) :=
  Prog
  If a = []  $\vee$  b = []
    RETURN APPEND(a, b)
  n_ := MAX(APPEND(APPEND(a), APPEND(b)))
  tocycles(MULPERMS(toperm(a, n_), toperm(b, n_)))

```

```
mulperms([[1, 6], [2, 3, 4]], [[1, 2], [3, 4, 5]]) = [[1, 3, 2, 6], [4, 5]]
```


And now two auxiliary functions (mind you!) that will make the following programs easier to understand. The first determines the order of a permutation p , i.e. the least positive exponent e such that the power p^e w.r.t. composition is the identity mapping, whereas the second orders the permutations in a given list s according to their orders and the number of elements in $\{1,2,\dots,n\}$ that are not mapped onto themselves.

```

order(p) :=
  If p = []
    1
  LCM(VECTOR(DIM(c_), c_, p))

sortperms(s) :=
  Prog
    s := VECTOR(ADJOIN([order(v_), DIM(APPEND(v_))], v_), v_, s)
    s := VECTOR(REST(v_), v_, SORT(s))

```

In particular, we can use sortperms(s) for a more readable form of symgroup(n) in my last "Titbits".

```

symgroup(n, k_, n_ := 2, s_ := [[1]], t_) :=
  Loop
    If n_ > n
      RETURN sortperms(VECTOR(tocycles(v_), v_, s_))
    k_ := n_
    t_ := []
    Loop
      If k_ = 0 exit
      t_ := APPEND(t_, VECTOR(INSERT(n_, v_, k_), v_, s_))
      k_ := k_ - 1
    s_ := t_
    n_ := n_ + 1

```

The most important new function however is permgroup(s) that computes for a list s of permutations the subgroup generated by those permutations in an appropriate S_n and comes in handy on many occasions. (Note that by a wellknown theorem of algebra, one can get all finite groups by considering the subgroups of all symmetric groups S_n !)

```

PERMGROUP(g, g_, s_, t_) :=
  Prog
    s_ := MAP_LIST(g+k_, k_, {1, ..., DIM(g)})
    Loop
      t_ := s_
      g_ := g
      Loop
        If g_ = [] exit
        s_ := s_ u MAP_LIST(MULPERMS(u_, FIRST(g_)), u_, s_)
        g_ := REST(g_)
      If s_ = t_
        RETURN SORT(s_)

permgroup(g, n_) :=
  Prog
    n_ := MAX(APPEND(APPEND(g)))
    g := VECTOR(toperm(p_, n_), p_, g)
    sortperms(VECTOR(tocycles(p_), p_, PERMGROUP(g)))

```

Do you know the subgroup of S_4 that is generated by the two cycles $[1,2,3,4]$ and $[1,2]$? Have a look at the following example:

```

permgroup [ [1, 2, 3, 4]
            [1, 3] ]

[ [], [[1, 3]], [[2, 4]], [ 1 2 ], [ 1 3 ], [ 1 4 ], [[1, 2, 3, 4]], [[1, 4, 3, 2]] ]

VECTOR[order(p_), p_, permgroup [ [1, 2, 3, 4]
                                   [1, 3] ] ] = [1, 2, 2, 2, 2, 2, 4, 4]

```

Keeping my promise from the start of this column, I'll now show you how these functions can be used to solve a celebrated problem of recreational mathematics, namely the so-called n queens problem. Just in case you have never heard of it, this is the problem to place n queens on a chess board with n^2 squares so that no queen is on the line of attack of any other queen. This problem was posed in 1848 by Max Bezzel in the "Berliner Schachzeitung" for the classical case $n=8$ and solved for this case by Franz Nauck in 1850. (It kept even Gauß busy for some time, who didn't succeed in finding all 92 solutions though.)

Using the search engine Altavista and the key words "queens problem" I got no less than 1754 hits. In particular, there were all sorts of programs where one could solve the n queens problem for a given n via Internet. Here comes (surprise, surprise!) yet another solution, this time using the powerful programming language of Derive. What might be special about my program is the fact that it is iterative rather than recursive and uses breadth first search (BFS) rather than depth first search (DFS).

To begin with let's consider the simpler case $n=4$. (By the way, it is easy to see that there are no solutions at all for $n=2$ and $n=3$. On the other hand, it can be proved that there is always a solution for $n>3$.) First we number the squares of the board in the following way:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

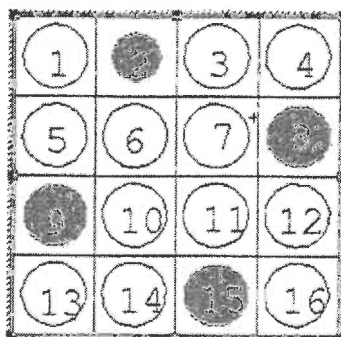
Since in a final solution for obvious reasons each row and each column of the board contains exactly one queen, we may w.l.o.g. put the first queen onto the first row, the second onto the second row and so on. Let's assume that we put the first queen say on square 2. As indicated in the following a picture (produced in Derive!) a number of squares are no longer available for subsequent queens then.

1	2	*3	4
5	6	7	8
9	10	11	12
13	14	15	16

In fact, there is only one square left in the second row to place the second queen there.

1	2	3	4
5	6	7	*8
9	10	11	12
13	14	15	16

Similarly it is clear that the third and the fourth queen must be put on the squares 9 and 15, respectively, leading to the following solution:



How can we get this solution (and of course all remaining ones) using Derive? Obviously the first thing we need is a routine that determines for a specific queen all squares that are in a line of attack as regards that queen. This could be programmed in the following way:

```
remove(p, s, n, x_, y_) :=
  Prog
    s := s \ {FLOOR(p - 1, n) * n + 1, ..., FLOOR(p - 1, n) * n + n}
    s := s \ {p, p + n, ..., n^2}
    x_ := MOD(p - 1, n)
    y_ := n - FLOOR(p - 1, n) - 1
    s := s \ {p, p + n - 1, ..., p + MIN(x_, y_) * (n - 1)}
    s := s \ {p, p + n + 1, ..., p + MIN(n - x_ - 1, y_) * (n + 1)}
```

Here p is the number of the square occupied by the queen, s is the set of squares that are still to be considered and n refers to the size of the board. For example, we get

```
remove(2, {1, ..., 16}, 4) = {8, 9, 11, 13, 15, 16}
```

as indicated in one of the pictures above.

The next routine has as first input a list l whose elements are of the form [u,s], where u is a list representing the selection of queens so far and s is the set of squares that are still “free”, i.e. available for the next queen. In practice, l represents the nodes in our search tree that have a fixed distance (=level) from the root of the tree. As the name indicates, the output of this routine is simply the corresponding list for the next level.

```
nextlevel(l, n, q_, s_, t_, u_, v_ := []) :=
  Loop
    If l = []
      RETURN REVERSE(v_)
    u_ := FIRST(FIRST(l))
    s_ := FIRST(REST(FIRST(l)))
    l := REST(l)
    Loop
      If s_ = {} exit
      q_ := MIN(s_)
      If q_ > (DIM(u_) + 1) * n exit
      t_ := remove(q_, s_, n)
      v_ := ADJOIN([APPEND(u_, [q_]), t_], v_)
      s_ := REST(s_)
```

Let's consider our example above again using these routines by computing all nodes of the first 2 levels of our search tree starting with [[[], {1,...,16}]]

```
nextlevel([[[[]], {1, ..., 16}]], 4) =
```

[1]	{7, 8, 10, 12, 14, 15}
[2]	{8, 9, 11, 13, 15, 16}
[3]	{5, 10, 12, 13, 14, 16}
[4]	{5, 6, 9, 11, 14, 15}

$$\text{nextlevel} \left[\begin{bmatrix} [1] & \{7, 8, 10, 12, 14, 15\} \\ [2] & \{8, 9, 11, 13, 15, 16\} \\ [3] & \{5, 10, 12, 13, 14, 16\} \\ [4] & \{5, 6, 9, 11, 14, 15\} \end{bmatrix}, 4 \right] = \begin{bmatrix} [1, 7] & \{14\} \\ [1, 8] & \{10, 15\} \\ [2, 8] & \{9, 13, 15\} \\ [3, 5] & \{12, 14, 16\} \\ [4, 5] & \{11, 14\} \\ [4, 6] & \{15\} \end{bmatrix}$$

It should be clear by now that we have simply to apply these routine n times and leave out the set components then in order to get all solutions.

```
n_queens(n) := (ITERATE(nextlevel(1_, n), 1_, [[[], {1, ..., n^2}]], n)) COL 1
```

$$\text{n_queens}(4) = \begin{bmatrix} 2 & 8 & 9 & 15 \\ 3 & 5 & 12 & 14 \end{bmatrix}$$

```
FIRST(n_queens(8)) = [1, 13, 24, 30, 35, 47, 50, 60]
```

```
DIM(n_queens(8)) = 92
```

Check the first solution as an example!

In the case $n=4$ it is easy to see that the second solution is symmetric to the first one., i.e. there is essentially 1 solution. What is the corresponding number for the classical case $n=8$? Here is where our theory of permutation groups comes into play. In particular, the dihedral group D_4 , i.e. the 8-element group of symmetries of a square is needed, which can be generated by a rotation by 90 degrees and any reflection. To get it we only have to set up these two permutations a, b of $\{1, 2, \dots, n^2\}$ and call our routine PERMGROUP($[a, b]$). For a specific solution $[a_1, \dots, a_n]$ of the n queens problem the “orbit” of all symmetric solutions is then obtained by sorting the lists $[f(a_1), \dots, f(a_n)]$ in ascending order for all f in PERMGROUP($[a, b]$). Of course, what we are after is a set of representatives of all those orbits. Do you think that Derive can find them? You bet! On my PC the computation of all 12 “essentially different” solutions for $n=8$ took only 18.4s (vs. 18s for the computation of all 92 solutions) meaning that all those grouptheoretic computations have been done in fractions of a second! Due to lack of space I leave further experimenting to you. Have a good time! (j.wiesenbauer@tuwien.ac.at)

```
n_queens(n, f := true, a_, b_, p_, s_, t_ := []) :=
  Prog
  s_ := (ITERATE(nextlevel(s_, n), s_, [[[], {1, ..., n^2}]], n)) COL 1
  If ~ f
    RETURN s_
  a_ := ITERATES(IF(u_ > n^2 - n, MOD(u_ - 1, n), u_ + n), u_, n, n^2 - 1)
  b_ := ITERATES(IF(MOD(u_, n), u_ - 2*n + 1, u_ + 1), u_, n^2 - n + 1, n^2 - 1)
  p_ := PERMGROUP([a_, b_])
  s_ := MAP_LIST(s_↓k_, k_, {1, ..., DIM(s_)})
  Loop
  If s_ = {}
    RETURN SORT(t_)
  f := FIRST(SORT(s_ \ REST(s_)))
  t_ := ADJOIN(f, t_)
  f := VECTOR(SORT(v_), v_, p_ COL f)
  s_ := s_ \ MAP_LIST(f↓k_, k_, {1, ..., DIM(f)})
  s_ := REST(s_)
```

$$\text{n_queens}(5) = \begin{bmatrix} 1 & 8 & 15 & 17 & 24 \\ 4 & 6 & 13 & 20 & 22 \end{bmatrix} \quad (0.37s)$$

```
DIM(n_queens(8, false)) = 92 (18s)
```

```
DIM(n_queens(8)) = 12 (18.4s)
```