THE DERIVE - NEWSLETTER #103

ISSN 1990-7079

THE BULLETIN OF THE DERIVE USER GROUP + CAS-TI

Contents:

September 2016

TIME 2018?

TIME Conferences have a long and successful history. Starting 1992 in Krems, Austria, as *International School on the Didactics of Computer Algebra* with only 22 presenters the series of conferences developed further and became a meeting place of educators and CAS-enthusiasts from all over the world.

The list of the places of the following events is long: Düsseldorf, Plymouth, Honolulu, Bonn, Särö (Sweden), Gettysburg, Gösing (Austria), Liverpool, Vienna, Montreal, Dresden, Buffelspoort (South Africa), Malaga, Tartu (Estonia), Krems and finally we came together this year in Mexico City.

The Conference objective was extended form using only DERIVE to the use of handheld symbolic calculators (TI-92, Voyage 200 and TI-NspireCAS), to Dynamic Geometry, to Spreadsheet applications and has become now open for all forms of technology which can be used in math education.

It would be a pity if the TIMES of TIME should have ended with the last conference in Mexico.

So we would like to ask if any institution or organisation is willing to organize TIME 2018?

DUG and organizers of past conferences will support you as much as possible.

We would be very glad and happy to receive any suggestions or proposals. If you have any questions, then please come back to me. You know my email-address.

Best regards
Josef
On behalf of all TIMERS

Just a reminder: You can download the Proceedings of all former conferences. Go to http://rfdz.ph-noe.ac.at/acdca/acdca-conferences.html.

Dear DUG Members,

A warm welcome to Newsletter #103. Full of new ideas back from TIME 2016 in Mexico City I tried to be more or less in time with this issue.

First of all, I'd like to thank the organizers of TIME 2016 for their work. We had great days in your country. I will intend to present a selection of presentations in the next DNLs. I think that now is the right time and occasion to give a "**Call for TIME 2018**". Please notice our message on the information page. I will send this request to the delegates of the last TIME-conferences, too. So it might happen that you will receive it twice!

It would be great if we could proceed with our series of CAS-centred conferences.

Marlene Torres-Skoumal from the Vienna International School sent a mail. I promised to publish it in our Newsletter. You will find her message in the User Forum, page 3. If you are interested in the devices, then please contact her. Many thanks, Marlene.

In this DNL you will find the English version of Wolfgang Alvermann's task for final exam presented in DNL#102. I must apologize if one or the other translation of the technical expressions related to metal treatments is not correct. Nevertheless, I hope that the article is well understandable.

Wolfgang sent a challenge – not for the CAS, because calculation is easy and does not need any technology assistance. The problem was how to plot the result. This was not so difficult with DERIVE (applying my favourite command VECTOR) but caused serious problems for TI-Nspire. It would have been not so complicated with Voyage 200 because of its capability for programming graphics. This cannot be done with Nspire? The solution is – once again – LUA. See the results in the "Pizza"-contribution. Thanks to Steve Arnold from DownUnder for his more than valuable support.

I know from my time as teacher that cryptography is interesting for students. Especially now in times of smartphone, facebook, e-banking, … it concerns our daily life more and more. Hill Encryption is a great example to combine matrix calculation with number theory (modular arithmetic). You will find a procedure of an easier algorithm for the good old TI-92 / Voyage 200. You may try transferring the Hill Cipher to TI-NspireCAS, too.

And then we have two problems which I found in the IBDG (Information Pages for Geometry). Gerhard Schröpfer has taken care for the "Problems' Corner" since many years. Some of his problems are not only a geometric challenge, but also an algebraic one. Contacting Gerhard we found out that we know each other since 53 years (serving in the army 1963). Please see my additional comments for the second problem on page 32.

Best regards until next time

Josef

Download all *DNL*-DERIVE- and TI-files from

http://www.austromath.at/dug/

All four parts of "CAS for Physics Examples" are now ready for download from

http://www.acdca.ac.at/

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

Editor: Mag. Josef Böhm
D´Lust 1, A-3042 Würmla, Austria
Phone: ++43-(0)660 3136365
e-mail: nojo.boehm@pgv.at

Contributions:
Please send all contributions to the Editor. Non-English speakers are encouraged 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* & CAS-*TI Newsletter* will be.

Next issue: December 2016

Preview: Contributions waiting to be published

Some simulations of Random Experiments, J. Böhm, AUT, Lorenz Kopp, GER
Wonderful World of Pedal Curves, J. Böhm, AUT
Tools for 3D-Problems, P. Lüke-Rosendahl, GER
Simulating a Graphing Calculator in *DERIVE*, J. Böhm, AUT
Graphics World, Currency Change, P. Charland, CAN
Cubics, Quartics – Interesting features, T. Koller & J. Böhm, AUT
Logos of Companies as an Inspiration for Math Teaching
Exciting Surfaces in the FAZ / Pierre Charland´s Graphics Gallery
BooleanPlots.mth, P. Schofield, UK
Old traditional examples for a CAS – What´s new? J. Böhm, AUT
Where oh Where is It? (GPS with CAS), C. & P. Leinbach, USA
Mandelbrot and Newton with *DERIVE*, Roman Hašek, CZK
Tutorials for the NSpireCAS, G. Herweyers, BEL
Some Projects with Students, R. Schröder, GER
Dirac Algebra, Clifford Algebra, D. R. Lunsford, USA
A New Approach to Taylor Series, D. Oertel, GER
Henon & Co; Find your very own Strange Attractor, J. Böhm, AUT
Rational Hooks, J. Lechner, AUT
Statistics of Shuffling Cards, H. Ludwig, GER
Charge in a Magnetic Field, H. Ludwig, GER
Factoring Trinomials, D. McDougall, CAN
Using DERIVE to extend an Exam Question, J. Griffiths, UK
Selected Lectures from TIME 2016

and others

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

Marlene's offer:

Hi Josef,

The VIS has about 20 TI-92s that are in good working order, but they would like to get rid of them since no one is using them. Do you know of a school or organization that could make good use of them?

Thanks, and hope that all is good with you. Would love to see you again!

Marlene (marlenes@aon.at)

Francisco Marcelo Fernández, La Plata, Argentina

Dear Josef,

I am attaching a Derive file with a problem I had manipulating simple algebraic equations. I hope you find what is wrong with it.

Best regards,

Marcelo

I am trying to do some simple algebraic manipulations of equations.
I multiply this expression by its denominator

$$\#1:\quad k = \frac{k\cdot 1\cdot(\beta\cdot \mathrm{COS}(k\cdot 1) + 2\cdot k\cdot 1\cdot \mathrm{SIN}(k\cdot 1))}{2\cdot k\cdot 1\cdot \mathrm{COS}(k\cdot 1) - \beta\cdot \mathrm{SIN}(k\cdot 1)}$$

$$\#2:\quad \left(k = \frac{k\cdot 1\cdot(\beta\cdot \mathrm{COS}(k\cdot 1) + 2\cdot k\cdot 1\cdot \mathrm{SIN}(k\cdot 1))}{2\cdot k\cdot 1\cdot \mathrm{COS}(k\cdot 1) - \beta\cdot \mathrm{SIN}(k\cdot 1)}\right)\cdot(2\cdot k\cdot 1\cdot \mathrm{COS}(k\cdot 1) - \beta\cdot \mathrm{SIN}(k\cdot 1))$$

and I obtain the desired result

$$\#3:\quad 2\cdot k^2\cdot \mathrm{COS}(k) - k\cdot\beta\cdot \mathrm{SIN}(k) = k\cdot(\beta\cdot \mathrm{COS}(k) + 2\cdot k\cdot \mathrm{SIN}(k))$$

However, If I try to do exactly the same with the square of the expression

$$\#4:\quad \left(k = \frac{k1\cdot(\beta\cdot \mathrm{COS}(k1) + 2\cdot k1\cdot \mathrm{SIN}(k1))}{2\cdot k1\cdot \mathrm{COS}(k1) - \beta\cdot \mathrm{SIN}(k1)}\right)^2$$

$$\#5:\quad k^2 = \frac{k1^2\cdot((\beta^2 - 4\cdot k1^2)\cdot \mathrm{COS}(k1)^2 + 4\cdot\beta\cdot k1\cdot \mathrm{SIN}(k1)\cdot \mathrm{COS}(k1) + 4\cdot k1^2)}{(2\cdot k1\cdot \mathrm{COS}(k1) - \beta\cdot \mathrm{SIN}(k1))^2}$$

$$\#6:\quad \left(k^2 = \frac{k\cdot 1^2\cdot((\beta^2 - 4\cdot k\cdot 1^2)\cdot \mathrm{COS}(k\cdot 1)^2 + 4\cdot\beta\cdot k\cdot 1\cdot \mathrm{SIN}(k\cdot 1)\cdot \mathrm{COS}(k\cdot 1) + 4\cdot k\cdot 1^2)}{(2\cdot k\cdot 1\cdot \mathrm{COS}(k\cdot 1) - \beta\cdot \mathrm{SIN}(k\cdot 1))^2}\right)\cdot(\sim$$

$$2\cdot k\cdot 1\cdot \mathrm{COS}(k\cdot 1) - \beta\cdot \mathrm{SIN}(k\cdot 1))^2$$

```
for some reason unknown to me the denominator is not cancelled

          4         2       3                        2  2        2
#7:    4·k ·COS(k)  − 4·k ·β·SIN(k)·COS(k) + k ·β ·SIN(k)  =


                 2  2       4               2          2              3       ~
         k·(4·k ·β ·COS(k)  + 8·k·β·(2·k  + 2·k − β )·SIN(k)·COS(k)  + (16·~
        ──────────────────────────────────────────────────────────────────────~
                                                                              ~
                                                                            (2~

          3       2  2     4          2        2              2               ~
        k  − 16·k ·β  + β )·SIN(k) ·COS(k)  + 4·k·β·(β  − 4·k)·SIN(k)·COS(k~
        ──────────────────────────────────────────────────────────────────────~
                                  2                                           ~
        ·k·COS(k) − β·SIN(k))                                                 ~

                2       4
        ) + 4·k·β ·SIN(k) )
        ────────────────────
```

DNL:

Dear Marcelo,

interesting problem. I could not resolve it in the direct way.
But there is some work around can be done:

(1) Writing the expression in another order or
(2) Substituting for sin(k1) and cos(k1) – e.g. α and β, doing the manipulation and then resubstituting.

Then it works. See file bug_simp.dfw.
Anyhow, I will put your problem into the next DNL.

Best regards
Josef

Wolfgang Pröpper, Nürnberg, Germany

Dear Josef,

Just very short:

The link given in your DUG-message from 5 September 2016 does not work.

You gave "~/DUG", it should read "~/dug".

Hope you are well. Thanks for the extra work and best regards to you and Noor,
Yours Wolfgang

DNL:

Dear Wolfgang,

many thanks for your notice. It's great to have DUG members who read the DNL so carefully. It was my fault. I didn't consider that the links are case sensitive. I corrected my signature immediately.

Best regards

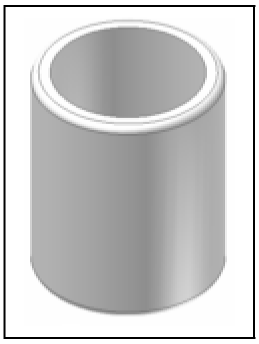
N & J

A Problem for the Final Exam

Wolfgang Alvermann, Germany

Newton's Law of Cooling → Regression / ODE

In the production of work pieces in metal-working steels are used which must have certain material properties. These properties can be influenced among others by selected heat treatments. The following heat treatments shall be applied:



1. Heating to a certain initial temperature.
2. Chilling (Immersion hardening in an oil bath in order to increase the temper of the steel).

Investigate the heat treatment of a hull which has the temperature of the environment at the beginning of the heat treatment.

a) During heating the hull in an oven with temperature of ϑ = 1000°C), the temperature of the steel is measured at certain times:

t in seconds [s]	100	200	400	800	1000	1500
ϑ in [°C]	237	406	640	867	920	977

For illustration visualize the heating process.

Sketch the run of the temperature using the given values in an appropriate coordinate system.

Describe the kind of the increase of temperature.

Find the function which describes this increase by an appropriate exponential regression. Explain the steps of solution.

b) The heating can be approximated by the following equation:

$$\vartheta(t) = 1000 - 980 \cdot e^{-0.0025t}$$

Calculate the environment temperature when heating is starting.

For the following chilling process of the hull an initial temperature of $\vartheta_A = 800°C$ is needed.

Investigate how many hulls can be heated in the stove per hour.

For complete through heating of the hull the change of temperature at the end of the heating process must not be greater than $v_\vartheta = 0.6\frac{°C}{s}$.

Check if this specification is met for $\vartheta_A = 800°C$.

For the following parts of the problem you can assume an initial temperature of $\delta_A = 800°C$ und an oil bath temperature of $\vartheta_U = 20°C$.

c) After heating the hull is chilled down in an oil bath from ϑ_A to ϑ_U. The cooling process can be described by a family of functions ϑ_k with $\vartheta_k(t) = \vartheta_U + (\vartheta_A - \vartheta_U) \cdot e^{-k \cdot t}$ (Newton's Law of Cooling) with parameter *k* depending on the kind of used oil.

We have:	t	: time in s
	$\vartheta(t)$	: temperature in °C at time *t*
	k	: cooling coefficient in $\frac{1}{s}$

Describe the influence of parameters ϑ_U, ϑ_A and k on the form of the graph of ϑ_k.

Find the equation of the family of functions, which describes the chilling process.

The process for chilling to a temperature of 50 °C shall last exact 80 seconds. Calculate the respective cooling coefficient *k* in order to select the appropriate oil.

d) Newton's law of cooling is the solution of the differential equation $\vartheta_k'(t) = -k \cdot (\vartheta(t) - \vartheta_U)$.

Show that the family of functions $\vartheta_k(t) = \vartheta_U + (\vartheta_A - \vartheta_U) \cdot e^{-k \cdot t}$ is a solution of this DE.

Heat energy stored in the hull during heating is given by the formula: $Q = m \cdot C \cdot \Delta\delta$

We have:	Q	: quantity of heat in *KJ* (*kJ*)	
	m	: mass of the work piece in kilogram (*kg*)	$m_{St} = 0.5\ kg$
	$\Delta\delta$	: change of temperature in Kelvin (*K*)	
	C	: specific capacity of heat in $\left(\frac{kJ}{kg \cdot K}\right)$	

$$C_{St} = 0.47 \frac{kJ}{kg \cdot K}$$

During chilling to temperature of environment heat energy *Q* stored in the hull caused by its heating is transmitted to the oil bath. Oil must be cooled when the heat energy emitted per hull is greater than 200 *kJ*. Decide if the oil must be cooled.

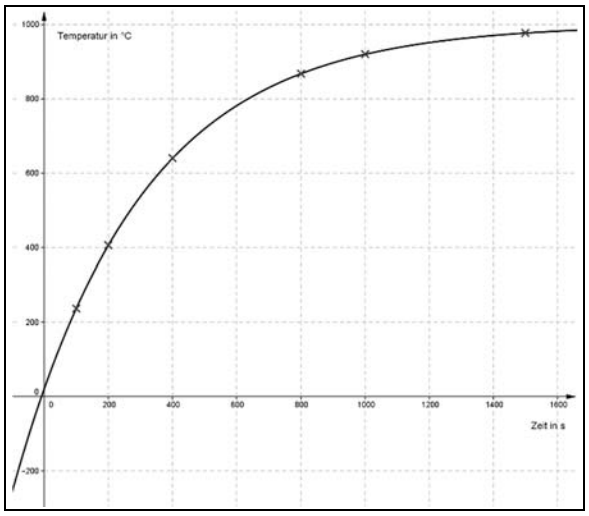
At the other hand heat energy can be interpreted as area under the cooling curve.

Then we have: $Q(t) = A \cdot \int_0^{t_{ab}} \left(\vartheta_k(t) - \vartheta_U(t)\right) dt,$ with $\vartheta_{0.0407}(t) = 20 + 780 \cdot e^{-0.0407 \cdot t}$.

Show that the coefficient for the hull is given by *A* = 0.00956 $\frac{KJ}{K \cdot s}$ and explain a way of calculation which can be performed without use of any CAS. Calculate the amount of heat energy which is transmitted within the first 50 seconds.

Solution a)

Sketch of the run of temperature / Description



Heating shows a limited growth because temperature starts increasing strongly, then the increase goes down and finally tends to the saturation value 1000°C.

Calculation of the respective function using regression

We extend the table by the row *1000-*δ in order to perform the regression of form $\vartheta(t) = a \cdot b^t$.

t in seconds [s]	100	200	400	800	1000	1500
ϑ in [°C]	237	406	640	867	920	977
1000 - ϑ	763	594	360	133	80	23

CAS delivers values *a* = 979.88 and *b* = 0.9975 from rows *t* and *1000-*ϑ.

Hence we receive the result

$\vartheta(t) = 1000 - 979.88 \cdot 0.9975^t$ which gives written with base *e*:

$$\boxed{\vartheta(t) \approx 1000 - 980 \cdot e^{-0.0025 \cdot t}}$$

Solution b)

Calculation of the environment temperature and number of hulls

Environment temperature is approximately 20 °C [$\vartheta(0) = 20$]

Equation for finding the number of hulls:

$800 = 1000 - 980 \cdot e^{-0.0025 \cdot t} \Rightarrow t = 635.69\,s$

Approximately 5 hulls can be heated per hour.

$$\frac{3600\,s}{635.69\,s} \approx 5.66$$

Finding the change of temperature:

$\vartheta'(t) = 2.45 \cdot e^{-0.0025 \cdot t} \Rightarrow \vartheta'(635.69) = 0.5 \Rightarrow v_f = 0.5 \frac{°C}{s}$

The required specification is met.

Solution c)

Influence of the parameters

ϑ_U: An increase of ϑ_U shifts the graph of ϑ_k upwards, a decrease downwards.

ϑ_A: An increase of ϑ_A shifts the intersection point of the graph with the *t*-axis upwards, a decrease downwards.

k: An increase of *k* compresses the graph ϑ_k of in *t*-direction, a decrease stretches the graph.

Equation of the family of functions / Cooling coefficient

$\vartheta_U = 20°C,\ \vartheta_A = 800°C$ substituted in $\vartheta_k(t) = \vartheta_U + (\vartheta_A - \vartheta_U) \cdot e^{-k \cdot t}$ gives

$\vartheta_k(t) = 20 + 780 \cdot e^{-k \cdot t}$

$\vartheta_k(80) = 50 \Rightarrow 50 = 20 + 780 \cdot e^{-80 \cdot k}$
$k = 0.0407$

The value of the cooling coefficient is $k = 0.0407 \frac{1}{s}$

Solution d)

Proof 1 of the differential equation

Substitution of the function and its derivative into the DE

$$\left.\begin{aligned} \vartheta_k(t) &= \vartheta_U + (\vartheta_A - \vartheta_U) \cdot e^{-k \cdot t} \\ \vartheta_k{}'(t) &= -k \cdot (\vartheta_A - \vartheta_U) \cdot e^{-k \cdot t} \end{aligned}\right\} \text{Substitution gives } -k \cdot (\vartheta_A - \vartheta_U) \cdot e^{-k \cdot t} = -k \cdot \left(\vartheta_U + (\vartheta_A - \vartheta_U) \cdot e^{-k \cdot t} - \vartheta_U\right)$$

Proof 2 of the differential equation

Solving the DE by separation of the variables

$$\left.\begin{aligned} \vartheta_k{}'(t) &= \frac{d\vartheta_k}{dt} = -k \cdot (\vartheta_k(t) - \vartheta_U) \\ \int \frac{d\vartheta_k}{\vartheta_k(t) - \vartheta_U} &= -\int k \cdot dt \\ \ln(\vartheta_k(t) - \vartheta_U) &= -k \cdot t + const \\ \vartheta_k(t) - \vartheta_U &= e^{-k \cdot t + C} = e^{-k \cdot t} \cdot \underbrace{e^{Const}}_{=C} \\ \vartheta_k(t) &= \vartheta_U + C \cdot e^{-k \cdot t} \end{aligned}\right\} \text{The initial condition for the cooling process is given by } \vartheta_k(0) = \vartheta_A \Rightarrow C = \vartheta_A - \vartheta_U$$

Hence: $\boxed{\vartheta_k(t) = \vartheta_U + (\vartheta_A - \vartheta_U) \cdot e^{-k \cdot t}}$ q.e.d. Proof 1 is preferable (in my opinion, of course).

Decision about oil cooling

$$Q = m \cdot C \cdot \Delta\vartheta = 0.5\,kg \cdot 0.47 \frac{kJ}{kg \cdot {}^\circ C} \cdot (800 - 20)^\circ C = 183.3\,kJ$$

In the hull heat energy of 183.3 kJ < 200 kJ is stored, so it is not necessary to cool the oil.

Proof of A without CAS

$$Q(t) = A \cdot \int_0^{t_{ab}} (\vartheta_{0.0407}(t) - 20)\,dt \Rightarrow$$

$$183.3 = A \cdot \lim_{t_{ab}\to\infty} \int_0^{t_{ab}} 780 \cdot e^{-0.0407 \cdot t} dt$$

$$183.3 = A \cdot \lim_{t_{ab}\to\infty} \left[-19165 \cdot e^{-0.0407 \cdot t}\right]_0^{t_{ab}}$$

$$= A \cdot \lim_{t_{ab}\to\infty} \left[-19165 \cdot e^{-0.0407 \cdot t_{ab}} - (-19165 \cdot e^{-0.0407 \cdot 0})\right]$$

$$= 19165 \cdot A$$

$$A = 0.00956 \frac{kJ}{K \cdot s}$$

The amount of heat transmitted during the first 50 seconds

$$Q(50) = 0.00956 \cdot \int_0^{50} (\vartheta_{0.0407}(t) - 20)\,dt = 159.27$$

An amount of 159.27 kJ is transmitted during the first 50 seconds.

Comments

In Lower Saxonian Upper Secondary Schools and in Central Exams of Vocational Gymnasiums it is demanded to provide professional oriened tasks from economy or engineering using a graphing calculator or a CAS. It is difficult to find real problems of this kind in the textbooks, so it remains the task for the teacher or teams of teachers to develop own problems which are close to reality at one hand side and which can be offered successfully to the students (and teachers, of course) on the other side.

In the problem given above I borrowed some ideas from heat-treating iron materials. Among others one must differ

- ***Make red hot*** → slow heating, keeping the temperature and slow cooling to reduce created tensions in the material
- ***Harden*** → heating and keeping the respective temperature, chilling in oil, water or air (steel becomes very hard and brittle), draw the temper (heating to a special temperature in order to reduce brittleness and fragility)
- More treatments like ***Quench and temper, Nitride hardening …***

This problem is from the Lower Saxonian Central Exam 2012 for Vocational Gymnasiums.

I took the problem, the solution in the form shown above is my work!

W. A.

Cryptography using Matrices – Hill Encryption

Josef Böhm, Würmla, Austria

Several years ago I produced two papers concerning working with matrices, both of them in German: "Matrizenrechung mit DERIVE" (2001) and several years later "Matrizenrechung mit dem TI-92"[1]. I presented some applications. In Brian Denton's book "Learning linear Algebra through DERIVE" [2] I found a nice example which might motivate students.

In "Mathematics Beyond the Numbers" (Gilbert & Hatcher, 2000) [3] I met "Hill Encryption" the first time and was enjoyed about the combination of modular arithmetic and matrix calculation. Searching in the web I came across Mike May's website [4] containing a more extended paper "*The Hill Cipher*"on this encryption method combined with a Maple worksheet.

In the following I will refer to these papers and provide procedures for DERIVE and TI-NspireCAS as well.

1 Matrix Encryption [3]

Let's assume that we would like to send the message "I like working with matrices more and more" – we will keep it secret because the message is too strange and we don't like to make it public!

In order to make the procedure as simple as possible we use upper case letters only and don't use any special characters. So our message changes to

"**ILIKEWORKINGWITHMATRICESMOREANDMORE**".

We replace the characters by their position numbers in the alphabet (A = 1, B = 2, …, Z = 26) and receive the following sequence of numbers:

9, 12, 9, 11, 5, 23, 15, 18, 11, 9, 14, 7, … 13, 15, 18, 5.

Then we have to rewrite this sequence as a sequence of 2×2 matrices:

$$\begin{pmatrix} 9 & 12 \\ 9 & 11 \end{pmatrix}, \begin{pmatrix} 5 & 23 \\ 15 & 18 \end{pmatrix}, \begin{pmatrix} 11 & 9 \\ 14 & 7 \end{pmatrix}, \ldots, \begin{pmatrix} 15 & 18 \\ 5 & 0 \end{pmatrix}$$

If necessary, then we fill up the last matrix with zeros.

Now we choose a "key-matrix" for encryption, e.g. $key = \begin{pmatrix} 3 & 4 \\ 5 & 2 \end{pmatrix}$ and multiply the sequence of matrices given above one after the another by the *key*. This results in a new sequence of matrices given the encoded message:

$$\begin{pmatrix} 63 & 80 \\ 63 & 79 \end{pmatrix}, \begin{pmatrix} 75 & 141 \\ 85 & 136 \end{pmatrix}, \begin{pmatrix} 89 & 55 \\ 92 & 53 \end{pmatrix}, \ldots, \begin{pmatrix} 65 & 54 \\ 55 & 36 \end{pmatrix}$$

We will not send the sequence of matrices but write the elements of them as a sequence of numbers:

63, 80, 63, 79, 75, 141, 85, 136, … 65, 54, 55, 36.

The important fact is that same letters in the plain text are not encoded as same numbers.

This message can be sent open; the key-matrix must be sent hidden to the addressee.

In my papers for DERIVE and TI-92 I added some tasks:

a) Accomplish encoding the full message.

b) Can you find the "recipe" for decoding the message?

c) Do you think that enciphering (= encoding) the message will become safer when using two key matrices C_1 and C_2?

d) Encode a self-chosen plain text using a 3×3 key matrix, hand in the encoded message and the key matrix to a colleague and let him/her decode the text.

e) What is important for choosing the key matrix?

f) Finally, try to perform encoding and decoding using a CAS. You can reproduce the procedure stepwise as shown above or even better, write programs for encoding and decoding.

2 Matrix Encryption performed with a Voyage 200 (or TI-92)

It has been a long time since my last V200 application was presented in the DNL. Believe it or not, I like to work with my little assistant even now in 2016. In times of TI-92 I provided a script `crypt.txt` to perform this matrix encryption.

```
:Cryptography
C:"MATRIZENRECHNUNGMACHTMIRIMMERMEHRSPASZ"→plain
 :Function tonumb(message) converts the string to a list of numbers
C:tonumb(plain)→newplain
 :number of characters
C:dim(newplain)
 :For a 2x2-key matrix two zeros must be added:
C:augment(newplain,{Ø,Ø})→newplain
C:
 :the key matrix c:
C:[3,4;2,3]→c
C:c*[newplain[⁻3+4k],newplain[⁻2+4k];newplain[⁻1+4k],newplain[4k]]|k=1→cd1
 :simply copy changing only two characters.
 :.....
C:c*[newplain[⁻3+4k],newplain[⁻2+4k];newplain[⁻1+4k],newplain[4k]]|k=1Ø→cd1Ø
 :Matrices cd1 - cd1Ø represent the encoded message.
C:c^⁻1*cd1
 :.....
C:c^⁻1*cd1Ø
 :gives a list
C:{13,1,2Ø,18,9,26,5,14,18,5,3,8,14,21,14,7,13,1,3,8,2Ø,13,9,18,9,13,13,5,18
,13,5,8,18,19,16,1,19,26,Ø,Ø}→numbs
 :tochar(list) converts the list of numbers back to a string (plain).
C:tochar(numbs)
C:delvar c,cd1,cd2,cd3,cd4,cd5,cd6,cd7,cd8,cd9,cd1Ø,zahlen,urtext,neu-
text,numbs,plain,newplain
```

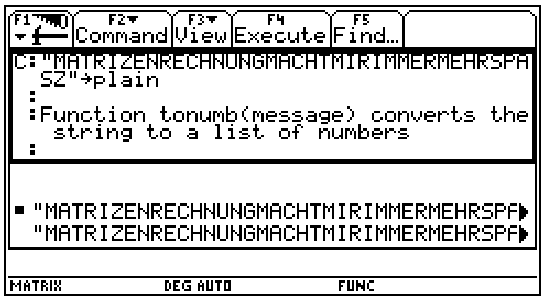
Only two short functions are used: `tonumb(string)` and `tochar(list of numbers)`.

The German script is provided as `krypt.txt` and the two functions are named `wandle(string)` and `rewandle(liste)`.

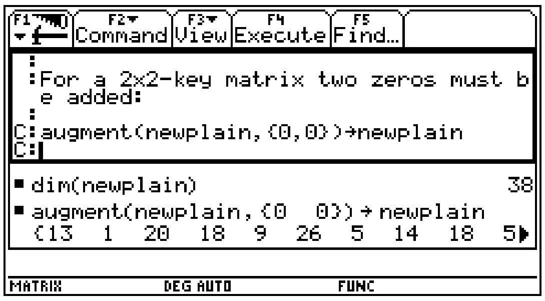
```
tonumb
(txt)
Func
Local  ordtxt,i
newList(dim(txt))→ordtxt
For  i,1,dim(txt)
ord(mid(txt,i,1))-64→ordtxt[i]
EndFor
```

```
tochar
(lis)
Func
Local mess,i
""→mess
For i,1,dim(lis)
mess&char(lis[i]+64)→mess
EndFor
mess
EndFunc
```

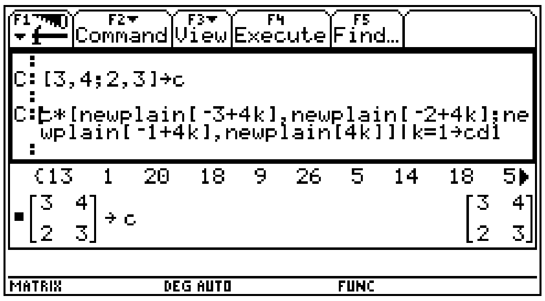
I enjoyed producing V200 screen shots showing parts of the script and its realization in the Home Screen. (Remember pressing F4 executes the command of the script.)



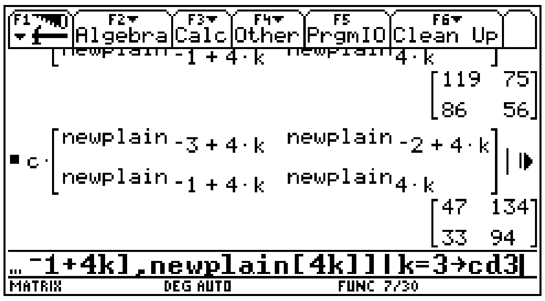
```
C:"MATRIZENRECHNUNGMACHTMIRIMMERMEHRSPA
  SZ"→plain
 :
 :Function tonumb(message) converts the
   string to a list of numbers
 :
■ "MATRIZENRECHNUNGMACHTMIRIMMERMEHRSPA▶
  "MATRIZENRECHNUNGMACHTMIRIMMERMEHRSPA▶
MATRIX         DEG AUTO          FUNC
```



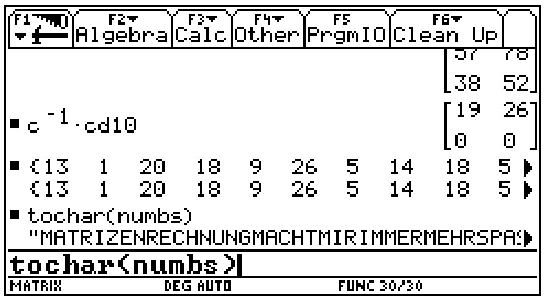
```
 :
 :For a 2x2-key matrix two zeros must b
  e added:
 :
C:augment(newplain,{0,0})→newplain
C:
■ dim(newplain)                        38
■ augment(newplain,{0  0})→newplain
  {13  1  20  18  9  26  5  14  18  5▶
MATRIX         DEG AUTO          FUNC
```



```
 :
C:[3,4;2,3]→c
 :
C:c*[newplain[-3+4k],newplain[-2+4k];ne
  wplain[-1+4k],newplain[4k]]|k=1→cd1
 :
 {13  1  20  18  9  26  5  14  18  5▶
■[3 4]→c                          [3 4]
 [2 3]                            [2 3]
MATRIX         DEG AUTO          FUNC
```



```
                                      [119  75]
                                      [86   56]
■ c·[newplain_{-3+4·k}  newplain_{-2+4·k}] |k=3
    [newplain_{-1+4·k}  newplain_{4·k}   ]
                                      [47  134]
                                      [33   94]
…-1+4k],newplain[4k]]|k=3→cd3
MATRIX         DEG AUTO          FUNC 7/30
```



```
                                      [57  78]
                                      [38  52]
                                      [19  26]
■ c^-1·cd10                           [0    0]
■ {13  1  20  18  9  26  5  14  18  5 ▶
  {13  1  20  18  9  26  5  14  18  5 ▶
■ tochar(numbs)
  "MATRIZENRECHNUNGMACHTMIRIMMERMEHRSPAS▶
tochar(numbs)
MATRIX         DEG AUTO          FUNC 30/30
```

This is not a program. Later I will present a more general program for TI-NspireCAS performing encryption and decryption. Let us first investigate how to proceed with DERIVE:

3 Matrix Encryption performed with DERIVE

It should be easy to follow the procedure.

#1: mess := ILIKEWORKINGWITHMATRICECMOREANDMORE

#2: key := $\begin{bmatrix} 3 & 4 \\ 2 & 5 \end{bmatrix}$

#3: DIM(mess) = 35

#4: mess1 := NAME_TO_CODES(mess)

#5: mess1 := [73, 76, 73, 75, 69, 87, 79, 82, 75, 73, 78, 71, 87, 73, 84, 72, 77, 65, 84, 82, 73, 67, 69, 67, 77, 79, 82, 69, 65, 78, 68, 77, 79, 82, 69]

#6: mess2 := APPEND(VECTOR(v − 64, v, mess1), [0])

#7: mess2 := [9, 12, 9, 11, 5, 23, 15, 18, 11, 9, 14, 7, 23, 9, 20, 8, 13, 1, 20, 18, 9, 3, 5, 3, 13, 15, 18, 5, 1, 14, 4, 13, 15, 18, 5, 0]

#8: mess3 := VECTOR$\left(\begin{bmatrix} \text{mess2}_{4\cdot k + 1} & \text{mess2}_{4\cdot k + 2} \\ \text{mess2}_{4\cdot k + 3} & \text{mess2}_{4\cdot k + 4} \end{bmatrix}, k, 0, 8\right)$

#9: mess3 := $\left[\begin{bmatrix} 9 & 12 \\ 9 & 11 \end{bmatrix}, \begin{bmatrix} 5 & 23 \\ 15 & 18 \end{bmatrix}, \begin{bmatrix} 11 & 9 \\ 14 & 7 \end{bmatrix}, \begin{bmatrix} 23 & 9 \\ 20 & 8 \end{bmatrix}, \begin{bmatrix} 13 & 1 \\ 20 & 18 \end{bmatrix}, \begin{bmatrix} 9 & 3 \\ 5 & 3 \end{bmatrix}, \begin{bmatrix} 13 & 15 \\ 18 & 5 \end{bmatrix}, \begin{bmatrix} 1 & 14 \\ 4 & 13 \end{bmatrix}, \begin{bmatrix} 15 & 18 \\ 5 & 0 \end{bmatrix}\right]$

#10: mess4 := VECTOR(key·v, v, mess3)

#11: mess4 := $\left[\begin{bmatrix} 63 & 80 \\ 63 & 79 \end{bmatrix}, \begin{bmatrix} 75 & 141 \\ 85 & 136 \end{bmatrix}, \begin{bmatrix} 89 & 55 \\ 92 & 53 \end{bmatrix}, \begin{bmatrix} 149 & 59 \\ 146 & 58 \end{bmatrix}, \begin{bmatrix} 119 & 75 \\ 126 & 92 \end{bmatrix}, \begin{bmatrix} 47 & 21 \\ 43 & 21 \end{bmatrix}, \begin{bmatrix} 111 & 65 \\ 116 & 55 \end{bmatrix}, \begin{bmatrix} 19 & 94 \\ 22 & 93 \end{bmatrix}, \begin{bmatrix} 65 & 54 \\ 55 & 36 \end{bmatrix}\right]$

#12: coded := APPEND(APPEND(mess4))

#13: coded := [63, 80, 63, 79, 75, 141, 85, 136, 89, 55, 92, 53, 149, 59, 146, 58, 119, 75, 126, 92, 47, 21, 43, 21, 111, 65, 116, 55, 19, 94, 22, 93, 65, 54, 55, 36]

#14: dec1 := VECTOR$\left(\begin{bmatrix} \text{coded}_{4\cdot k + 1} & \text{coded}_{4\cdot k + 2} \\ \text{coded}_{4\cdot k + 3} & \text{coded}_{4\cdot k + 4} \end{bmatrix}, k, 0, 8\right)$

#15: dec1 := $\left[\begin{bmatrix} 63 & 80 \\ 63 & 79 \end{bmatrix}, \begin{bmatrix} 75 & 141 \\ 85 & 136 \end{bmatrix}, \begin{bmatrix} 89 & 55 \\ 92 & 53 \end{bmatrix}, \begin{bmatrix} 149 & 59 \\ 146 & 58 \end{bmatrix}, \begin{bmatrix} 119 & 75 \\ 126 & 92 \end{bmatrix}, \begin{bmatrix} 47 & 21 \\ 43 & 21 \end{bmatrix}, \begin{bmatrix} 111 & 65 \\ 116 & 55 \end{bmatrix}, \begin{bmatrix} 19 & 94 \\ 22 & 93 \end{bmatrix}, \begin{bmatrix} 65 & 54 \\ 55 & 36 \end{bmatrix}\right]$

#16: unkey := key^{-1}

```
#17:   unkey := 
```
$$\begin{bmatrix} \frac{5}{7} & -\frac{4}{7} \\ -\frac{2}{7} & \frac{3}{7} \end{bmatrix}$$

```
#18:   dec2 := VECTOR(unkey·v, v, dec1)

#19:   dec2 :=
```
$$\left[\begin{bmatrix} 9 & 12 \\ 9 & 11 \end{bmatrix}, \begin{bmatrix} 5 & 23 \\ 15 & 18 \end{bmatrix}, \begin{bmatrix} 11 & 9 \\ 14 & 7 \end{bmatrix}, \begin{bmatrix} 23 & 9 \\ 20 & 8 \end{bmatrix}, \begin{bmatrix} 13 & 1 \\ 20 & 18 \end{bmatrix}, \begin{bmatrix} 9 & 3 \\ 5 & 3 \end{bmatrix}, \begin{bmatrix} 13 & 15 \\ 18 & 5 \end{bmatrix}, \begin{bmatrix} 1 & 14 \\ 4 & 13 \end{bmatrix}, \begin{bmatrix} 15 & 18 \\ 5 & 0 \end{bmatrix}\right]$$

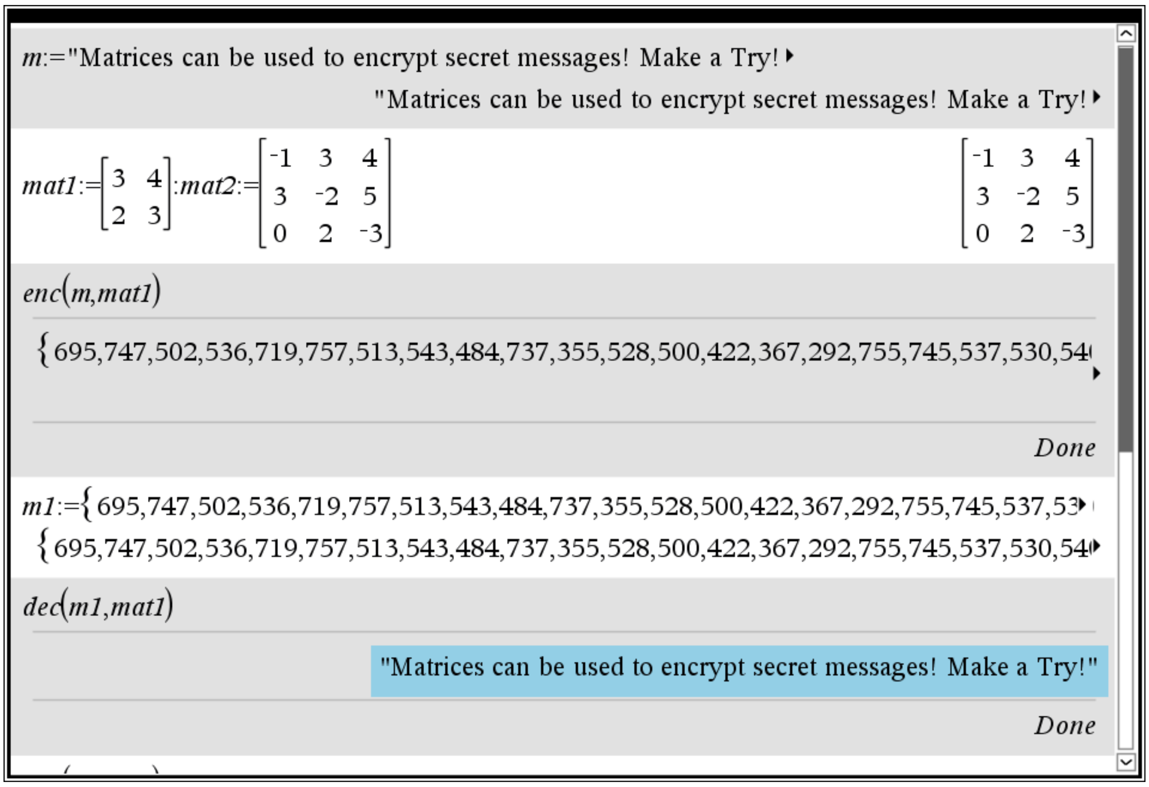
```
#20:   dec3 := APPEND(APPEND(dec2))

#21:   dec3 := [9, 12, 9, 11, 5, 23, 15, 18, 11, 9, 14, 7, 23, 9, 20, 8, 13, 1, 20, 18, 9, 3,
          5, 3, 13, 15, 18, 5, 1, 14, 4, 13, 15, 18, 5, 0]

#22:   dec4 := CODES_TO_NAME(VECTOR(v + 64, v, dec3))

#23:   dec4 := ILIKEWORKINGWITHMATRICECMOREANDMORE@
```

Additional tasks:

Is it possible to work with ASCII-Codes and include lower case letters and special characters?
Can we convert the encoded message to a string (text)?

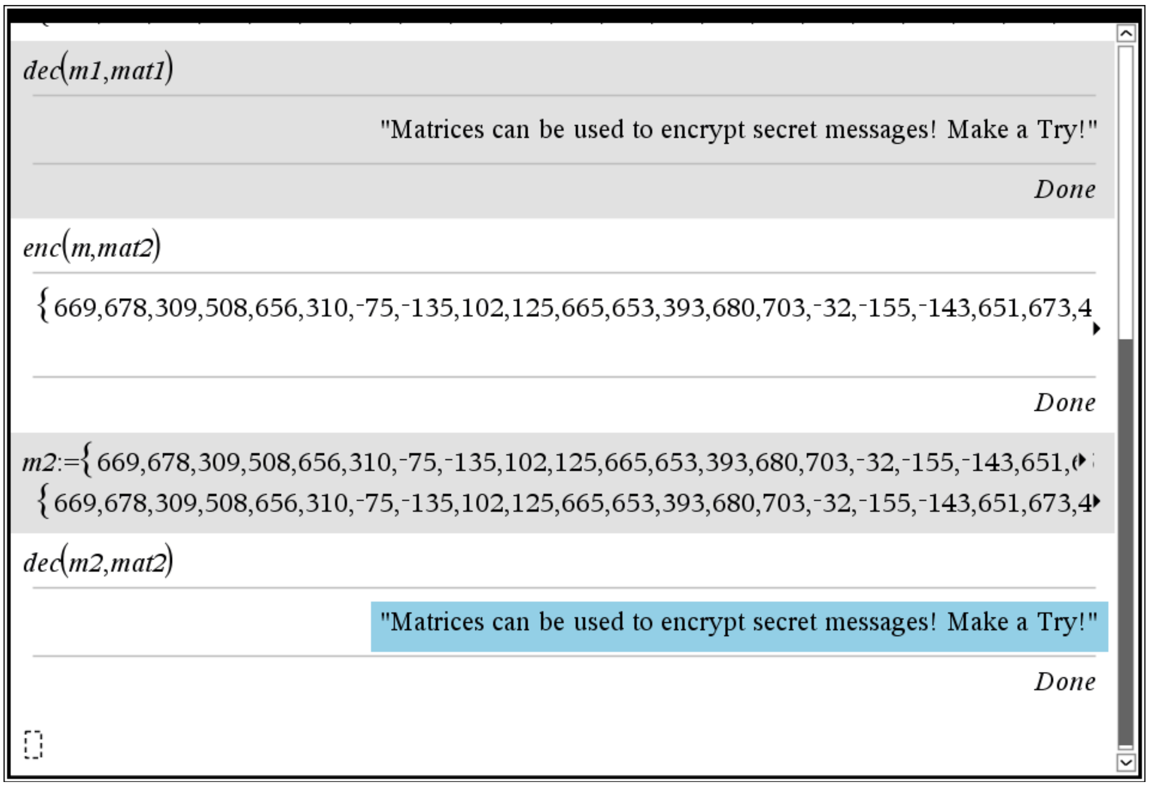
4 Matrix Encryption programs for TI-NspireCAS

See the screen shots followed by the programs:



```
m:="Matrices can be used to encrypt secret messages! Make a Try!▸
                    "Matrices can be used to encrypt secret messages! Make a Try!▸
```
$mat1:=\begin{bmatrix} 3 & 4 \\ 2 & 3 \end{bmatrix}:mat2:=\begin{bmatrix} -1 & 3 & 4 \\ 3 & -2 & 5 \\ 0 & 2 & -3 \end{bmatrix}$ $\quad\begin{bmatrix} -1 & 3 & 4 \\ 3 & -2 & 5 \\ 0 & 2 & -3 \end{bmatrix}$

```
enc(m,mat1)
 {695,747,502,536,719,757,513,543,484,737,355,528,500,422,367,292,755,745,537,530,54
                                                                              Done
m1:={695,747,502,536,719,757,513,543,484,737,355,528,500,422,367,292,755,745,537,53▸
 {695,747,502,536,719,757,513,543,484,737,355,528,500,422,367,292,755,745,537,530,54▸
dec(m1,mat1)
                    "Matrices can be used to encrypt secret messages! Make a Try!"
                                                                              Done
```



```
dec(m1,mat1)
"Matrices can be used to encrypt secret messages! Make a Try!"
Done
enc(m,mat2)
{669,678,309,508,656,310,⁻75,⁻135,102,125,665,653,393,680,703,⁻32,⁻155,⁻143,651,673,4▸
Done
m2:={669,678,309,508,656,310,⁻75,⁻135,102,125,665,653,393,680,703,⁻32,⁻155,⁻143,651,(▸
{669,678,309,508,656,310,⁻75,⁻135,102,125,665,653,393,680,703,⁻32,⁻155,⁻143,651,673,4▸
dec(m2,mat2)
"Matrices can be used to encrypt secret messages! Make a Try!"
Done
```

m1 and *m2* are the messages which can be sent open, *mat1* and *mat2* are the respective coding matrices which must be kept secret.

As you can see, we encrypt full text messages using arbitrary $n\times n$ (invertible) matrices. It is still not possible to send the encoded message as a string – which would be nicer than the sequence of numbers.

```
Define enc(mess,matr)=
Prgm
:Local dm,n_mess,zers,nm,dmt
:n_mess:={}:cr_mess:={}
:dm:=dim(mess): dmt:=dim(matr)[1]
:m_z:=mod(dm,dmt^(2))
:For i,1,dm
:n_mess:=augment(n_mess,{ord(mid(mess,i,1))})
:EndFor
:n_mess:=when(m_z≠0,augment(n_mess,seq(0,k,1,dmt^(2)−m_z)),n_mess)
:dm:=dim(n_mess)
:For i,0,((dm)/(dmt^(2)))−1
:listp:=mid(n_mess,1+i*dmt^(2),dmt^(2))
:nm:=list▸mat(listp,dmt)
:nm:=matr*nm
:cr_mess:=augment(cr_mess,mat▸list(nm))
:EndFor
:Disp cr_mess
:EndPrgm
```

```
Define dec(mess,matr)=
Prgm
:Local dm,n_mess,nm,dmt,im,txt
:dm:=dim(mess):n_mess:={}:txt:=""
:dmt:=dim(matr)[1]
:im:=matr^(⁻1)
:For i,0,((dm)/(dmt^(2)))−1
:listp:=mid(mess,1+i*dmt^(2),dmt^(2))
:nm:=list▸mat(listp,dmt)
:nm:=im*nm
:n_mess:=augment(n_mess,mat▸list(nm))
:EndFor
:For i,1,dm
:txt:=txt&char(n_mess[i])
:EndFor
:Disp string(txt)
:EndPrgm
```

5 Hill Encryption 1

This encoding method was published by Lester S. Hill in 1929. The first explanation follows my resource [3].

Let’s make it easy for the beginning and assume that our message consists of the 26 upper case letters only like above and we will also use a 2×2 key matrix.

We take a short message: “LETSHAVEFUN”, which must be rewritten in form of two-letter blocks: {LE, TS, … FU, NZ}. We accomplish the last block with a Z. The letters A through Z are substituted by numbers 0 to 25. We choose four integers *a*, *b*, *c*, and *d*. Now we can encode one two-letter (P1,P2) block after the other giving (C1,C2) applying the following recipe:

$$\begin{pmatrix} C1 \\ C2 \end{pmatrix} = \begin{pmatrix} (a\cdot P1 + b\cdot P2)\bmod 26 \\ (c\cdot P1 + d\cdot P2)\bmod 26 \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} P1 \\ P2 \end{pmatrix} \bmod 26 .$$

Let’s call the matrix *M*.

This is easy done, but how to decipher the block? We try to solve the two equations given in the rows of the matrix for P1 and P2. We eliminate P2 and P1

$$\left.\begin{array}{l} C1 = a\cdot P1 + b\cdot P2 \bmod 26 \mid \cdot d \\ C2 = c\cdot P1 + d\cdot P2 \bmod 26 \mid \cdot(-b) \end{array}\right\}+ \qquad \left.\begin{array}{l} C1 = a\cdot P1 + b\cdot P2 \bmod 26 \mid \cdot(-c) \\ C2 = c\cdot P1 + d\cdot P2 \bmod 26 \mid \cdot a \end{array}\right\}+$$

$$d\cdot C1 - b\cdot C2 = (a\cdot d - b\cdot c)\cdot P1 \bmod 26 \qquad -c\cdot C1 + a\cdot C2 = (a\cdot d - b\cdot c)\cdot P2 \bmod 26$$

$$P1 = (d\cdot C1 - b\cdot C2)\cdot \mathrm{mod}^{-1}(a\cdot d - b\cdot c, 26) \bmod 26 \qquad P2 = (-c\cdot C1 + a\cdot C2)\cdot \mathrm{mod}^{-1}(a\cdot d - b\cdot c, 26) \bmod 26$$

$\mathrm{mod}^{-1}(x,z)$ is the inverse mod of *x* mod *z*.

So we can write the rule to obtain (P1, P2) from (C1, C2) as follows:

$$\begin{pmatrix} P1 \\ P2 \end{pmatrix} = \text{mod}^{-1}(\det(M), 26) \cdot \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} \text{mod}\, 26 = \text{mod}^{-1}(\det(M), 26) \cdot \text{adj}(M)\, \text{mod}\, 26.$$

adj(M) is the adjugate matrix of M (adjungierte Matrix). This is the transposed matrix of its cofactors.

DERIVE provides functions INVERSE_MOD(x,z) and ADJ(matrix) as well.

TI-NspireCAS does not provide the inverse mod, library linalgcas offers the function comatrix which gives the matrix of cofactors which must be transposed to have the adjugate matrix.

$$\text{ADJ(M)} = \text{DET(M)} \cdot \text{M}^{-1}$$

The determinant of the encrypting matrix must not have any common factors with the modular base (26).

We transfer the procedure to DERIVE:

```
#1:   [CaseMode := Sensitive, InputMode := Word]

#2:   plain := LETSHAVEFUNZ
```

$$\#3: \quad M := \begin{bmatrix} 12 & 3 \\ 5 & 6 \end{bmatrix}$$

```
#4:   plain1 := VECTOR(v - 65, v, NAME_TO_CODES(plain))

#5:   plain1 := [11, 4, 19, 18, 7, 0, 21, 4, 5, 20, 13, 25]
```

$$\#6: \quad \left(C := \text{MOD}\left(M \cdot \begin{bmatrix} 11 \\ 4 \end{bmatrix}, 26\right)\right) = C := \begin{bmatrix} 14 \\ 1 \end{bmatrix}$$

```
#7:   MOD(DET(M), 26) = 5

#8:   INVERSE_MOD(DET(M), 26) = 21

#9:   MOD(5·21, 26) = 1
```

Here you can see the property of the inverse mod.

$$\#10: \quad \text{INVERSE\_MOD(DET(M), 26)} \cdot \text{ADJOINT(M)} = \begin{bmatrix} 126 & -63 \\ -105 & 252 \end{bmatrix}$$

$$\#11: \quad (\text{revkey} := \text{MOD}(21 \cdot \text{ADJOINT(key)}, 26)) = \text{revkey} := \begin{bmatrix} 22 & 15 \\ 25 & 18 \end{bmatrix}$$

$$\#12: \quad \text{MOD}(\text{revkey} \cdot C, 26) = \begin{bmatrix} 11 \\ 4 \end{bmatrix}$$

So we are back with the first two numbers which are easy to convert to letters.

Now we are ready to send the whole message.

#13: $(\text{plain2} := \text{VECTOR}([\text{plain1}_i, \text{plain1}_{i+1}], i, 1, 11, 2)) = \text{plain2} := \begin{bmatrix} 11 & 4 \\ 19 & 18 \\ 7 & 0 \\ 21 & 4 \\ 5 & 20 \\ 13 & 25 \end{bmatrix}$

#14: $\text{coded1} := \text{APPEND}(\text{VECTOR}(\text{MOD}(\text{M}\cdot\text{plain2}_i, 26), i, 1, 6))$

#15: coded1 := [14, 1, 22, 21, 6, 9, 4, 25, 16, 15, 23, 7]

#16: mess := CODES_TO_NAME(VECTOR(v + 65, v, coded1))

#17: mess := OBWVGJEZQPXH

We send “OBWVGJEZQPXH”. Our partner has received the secret matrix `revkey` and can start decoding our message:

#1: mess := OBWVGJEZQPXH

#2: $\text{revkey} := \begin{bmatrix} 22 & 15 \\ 25 & 18 \end{bmatrix}$

#3: mess1 := NAME_TO_CODES(mess)

#4: mess2 := VECTOR(v − 65, v, mess1)

#5: $\text{mess3} := \text{VECTOR}([\text{mess2}_i, \text{mess2}_{i+1}], i, 1, 11, 2)$

#6: $\text{mess4} := \text{APPEND}(\text{VECTOR}(\text{MOD}(\text{revkey}\cdot\text{mess3}_i, 26), i, 1, 6))$

#7: mess4 := [11, 4, 19, 18, 7, 0, 21, 4, 5, 20, 13, 25]

#8: CODES_TO_NAME(VECTOR(v + 65, v, mess4))

#9: LETSHAVEFUNZ

6 Hill Encryption 2

This method is far away from being safe in our times. It can be attacked successfully even when using larger encoding matrices.

However, I like it because it is a very nice application of matrix calculation together with modular arithmetic.

We will extend the DERIVE procedure to make it more general (larger matrices and all characters). This will involve another concept, hexadecimal numbers according Mike May’s paper “The Hill Cipher”.

Our modular base is 127 (which is fortunately prime). Trying to use the full ASCII-Code we will come across technical problems – see later. The DERIVE realisation is following.

```
Hill_new.dfw

 #1:    CaseMode := Sensitive

 Short Cuts

 #2:    NTC(x_) := NAME_TO_CODES(x_)

 #3:    CTN(x_) := CODES_TO_NAME(x_)

 Number Base Conversion from DNL#29 updated

 #4:    REM(number, base) := MOD(number, base)

        SELCT(r) := [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F]
 #5:                                                                r + 1

 #6:    SEQU(number, base) := ITERATES(FLOOR(x, base), x, number)

        AUX(number, base) := VECTOR(SELCT(REM((SEQU(number, base)) , base)),
 #7:                                                                 k

          k, 1, DIMENSION(SEQU(number, base)) - 2)

 #8:    CONV(a, x_) := [REVERSE_VECTOR(AUX(a, x_))]

        HEX(a) :=
          If a = 0
             [0, 0]
 #9:         If a > 15
                (CONV(a, 16))↓1
                APPEND([0], (CONV(a, 16))↓1)

        HX(a) := APPEND(STRING((HEX(a)) ), STRING((HEX(a)) ))
 #10:                                 1                  2

        INVHX(a, a1, a2) :=
          Prog
            a1 := (NTC(a↓1))↓1 - 48
 #11:       a2 := (NTC(a↓2))↓1 - 48
            a1 := IF(a1 ≤ 9, a1, a1 - 7)·16
            a2 := IF(a2 ≤ 9, a2, a2 - 7)
            a1 + a2

                                      ┌ 3  8 ┐
 #12:   [HEX(56), HEX(13)] =          │      │
                                      └ 0  D ┘

 #13:   [HX(56), HX(13)] = [38, 0D]

 #14:   [INVHX(38), INVHX(0D)] = [56, 13]

 We will need redimension of a matrix (DNL#17)

        redim(m_, r_, c_) := VECTOR(VECTOR((APPEND(m_))                 , k, c_),
 #15:                                                   (i - 1)·c_ + k

          i, r_)
```

#1 to #15 are auxiliary function which will be needed.

Our message is a text from Wikipedia:

```
#16:   wiki ≔ In classical cryptography, the Hill cipher is a polygraphic
         substitution cipher based on linear algebra. Invented by Lester S.
         Hill in 1929, it was the first polygraphic cipher in which it was
         practical (though barely) to operate on more than three symbols at
         once. The following discussion assumes an elementary knowledge of
         matrices. (Wikipedia)

#17:   wiki2 ≔ NTC(wiki)

#18:   wiki2 ≔ [73, 110, 32, 99, 108, 97, 115, 115, 105, 99, 97, 108, 32,
         99, 114, 121, 112, 116, 111, 103, 114, 97, 112, 104, 121, 44, 32,
         116, 104, 101, 32, 72, 105, 108, 108, 32, 99, 105, 112, 104, 101,
```

I will not copy the full lists in the following:

The key–matrix given in May's paper:

#19: $$\text{key} := \begin{bmatrix} 1 & 2 & 3 \\ 0 & 5 & 7 \\ 0 & 0 & 11 \end{bmatrix}$$

and the respective dekey–matrix:

#20: $$\begin{bmatrix} 1 & 25 & 30 \\ 0 & 51 & 83 \\ 0 & 0 & 104 \end{bmatrix}$$

But, how to find it?

There are two ways to find the adjoint (= adjugate) matrix:

#21: $$\left[\text{key}^{-1}\cdot\text{DET(key)},\ \text{ADJOINT(key)}\right] = \left[\begin{bmatrix} 55 & -22 & -1 \\ 0 & 11 & -7 \\ 0 & 0 & 5 \end{bmatrix}, \begin{bmatrix} 55 & -22 & -1 \\ 0 & 11 & -7 \\ 0 & 0 & 5 \end{bmatrix}\right]$$

#22: $$\text{INVERSE\_MOD(DET(key), 127)}\cdot\text{ADJOINT(key)} = \begin{bmatrix} 5335 & -2134 & -97 \\ 0 & 1067 & -679 \\ 0 & 0 & 485 \end{bmatrix}$$

#23: $$\left(\text{dekey} := \text{MOD}\left(\begin{bmatrix} 5335 & -2134 & -97 \\ 0 & 1067 & -679 \\ 0 & 0 & 485 \end{bmatrix}, 127\right)\right) = \text{dekey} := \begin{bmatrix} 1 & 25 & 30 \\ 0 & 51 & 83 \\ 0 & 0 & 104 \end{bmatrix}$$

dekey is the matrix to decipher the encoded message. We encode and decode the first three characters and we can observe that the procedure works properly.

#24: $$\text{key}\cdot\begin{bmatrix}73\\110\\32\end{bmatrix}=\begin{bmatrix}389\\774\\352\end{bmatrix}$$

#25: $$\text{MOD}\left(\text{key}\cdot\begin{bmatrix}73\\110\\32\end{bmatrix},\ 127\right)=\begin{bmatrix}8\\12\\98\end{bmatrix}$$

#26: $$\text{MOD}\left(\text{dekey}\cdot\begin{bmatrix}8\\12\\98\end{bmatrix},\ 127\right)=\begin{bmatrix}73\\110\\32\end{bmatrix}$$

We add two spaces (ASCII = 32) to make the number of characters divisible by three and make the first try encoding our message using the given key-matrix.

```
#27:   DIM(wiki2) = 347

#28:   MOD(347, 3) = 2

#29:   wiki3 := APPEND(wiki2, [32])

       encode(mess, key_m, c_mess, d_, v_, m_) :=
         Prog
           c_mess := []
           d_ := (DIM(key_m))↓1
           v_ := VECTOR(k, k, 1, d_)
#30:       Loop
             If DIM(mess) = 0 exit
             m_ := MOD(key_m·mess↓v_', 127)'
             c_mess := APPEND(c_mess, m_)
             mess := DELETE(mess, v_)
           c_mess

#31:   wiki3_enc := encode(wiki3, key)

#32:   wiki3_enc := [8, 12, 98, 98, 76, 51, 25, 40, 12, 109, 98, 45, 64, 23,

         111, 58, 102, 6, 24, 43, 111, 125, 18, 1, 51, 63, 98, 119, 84, 95,

       decode(mess, key_c, c_mess, d_, v_, m_) :=
         Prog
           c_mess := []
           d_ := (DIM(key))↓1
           v_ := VECTOR(k, k, 1, d_)
#33:       Loop
             If DIM(mess) = 0 exit
             m_ := MOD(key_c·mess↓v_', 127)'
             c_mess := APPEND(c_mess, m_)
             mess := DELETE(mess, v_)
           CTN(c_mess)
```

decode is our routine for deciphering the encoded message. Will it work?

```
#34:  decode(wiki3_enc, dekey)

#35:  In classical cryptography, the Hill cipher is a polygraphic
        substitution cipher based on linear algebra. Invented by Lester S.
        Hill in 1929, it was the first polygraphic cipher in which it was
        practical (though barely) to operate on more than three symbols at
        once. The following discussion assumes an elementary knowledge of
        matrices. (Wikipedia)
```

Great!!

I produced a tool to generate $n \times n$ key-matrices together with the respective dekey-matrices. These tools don't appear in May's paper.

```
#36:  de_key(m) := MOD(ADJOINT(m)·INVERSE_MOD(DET(m), 127), 127)

      hillkeys(n, m_) :=
        Loop
          m_ := RANDOM_MATRIX(n, n, 50)
#37:      m_ := MOD(m_, 50)
          If DET(m_) ≠ 0
             RETURN [m_, MOD(ADJOINT(m_)·INVERSE_MOD(DET(m_), 127), 127)]
```

#38: hillkeys(3) = $\left[\begin{bmatrix} 47 & 7 & 44 \\ 37 & 5 & 0 \\ 22 & 10 & 37 \end{bmatrix}, \begin{bmatrix} 63 & 28 & 107 \\ 118 & 47 & 21 \\ 116 & 29 & 44 \end{bmatrix}\right]$

#39: de_key $\begin{bmatrix} 47 & 7 & 44 \\ 37 & 5 & 0 \\ 22 & 10 & 37 \end{bmatrix} = \begin{bmatrix} 63 & 28 & 107 \\ 118 & 47 & 21 \\ 116 & 29 & 44 \end{bmatrix}$

Our message must be sent as a list of numbers (wiki3_enc) which is not very handy. Converting back from ASCII-code to characters does not work because some code numbers are not accepted (0, 10, 13, 34, …) and others are strange to read. It could look like this:

```
8HbW'bD┬‼}pTK4N♬☼♀C=g¶(o'Uvzv-c◘}┘|  ┐Lz─┤oO∗b┴(zn?uZpj6

  b└pT6Cb  \└¶QbuY┤‼┴u|◀_D&  Q[I\─||_53g┘)b─$uA

  _7•_┐→-hbT|)C:∗bkE_Ib♀∗b~h_5=G-♀→^b┴^bS)3=%bt7Y─&  &T─sM♀↑>─YWN┤6_→M
  N,•b6

  bwT_♂→3QP}#ON[M3┐┘3B%b)-EYJ5  |•_#T_o%b-a─[Nkp─♀a~  c─♀-?b┴^b])3(z
  DJb%CN2Ib8HbW'b
```

These are the "technical problems"!

May proposes to convert the encoded message in blocks consisting of the hexadecimal representation of the numbers given in the encoded message. We use the tools provided to create the hexadecimal numbers (two digits!). So we need also the leading zeros for one-digit numbers!

```
#41:   encode_h(wiki3, key)

#42:   [08, 0C, 62, 62, 4C, 33, 19, 28, 0C, 6D, 62, 2D, 40, 17, 6F, 3A, 66,

          06, 18, 2B, 6F, 7D, 12, 01, 33, 3F, 62, 77, 54, 5F, 6E, 4F, 0C, 27,

          02, 62, 0A, 27, 59, 0D, 21, 6F, 4F, 3C, 7A, 44, 4A, 62, 17, 29, 2D,
```

Then I form 6 digit “words” which represent the encoded message. I don’t print the two functions here.

```
#43:   encode_hb(wiki3, key)

#44:   [080C62, 624C33, 19280C, 6D622D, 40176F, 3A6606, 182B6F, 7D1201,

          333F62, 77545F, 6E4F0C, 270262, 0A2759, 0D216F, 4F3C7A, 444A62,

          17292D, 222B6F, 7D1201, 125462, 08013E, 1B2D0C, 3F0006, 163743,
```

Let’s put all together forming the encoding function:

```
         hill_enc(mess, key_m, c_mess, v_mess, d_, v_, m_, i, bl) :=
           Prog
             mess := NTC(mess)
             [c_mess := [], v_mess := []]
             d_ := (DIM(key_m))↓1
             v_ := VECTOR(k, k, 1, d_)
             mess := APPEND(mess, VECTOR(32, k, d_ - MOD(DIM(mess), d_)))
             Loop
               If DIM(mess) = 0 exit
               m_ := MOD(key_m·mess↓v_`, 127)`
               c_mess := APPEND(c_mess, m_)
               mess := REST(mess, d_)
#46:         c_mess := VECTOR(HX(e), e, c_mess)
             Loop
               If DIM(c_mess) = 0 exit
               [i := 1, bl := ""]
               Loop
                 If i > d_ exit
                 bl := APPEND(bl, FIRST(c_mess))
                 c_mess := REST(c_mess)
                 i :+ 1
               v_mess := APPEND(v_mess, [bl])
             [i := 1, bl := ""]
             v_mess
```

We encode our message using the given key-matrix (#19):

```
#47:   wiki_coded := hill_enc(wiki, key)

#48:   wiki_coded := [080C62, 624C33, 19280C, 6D622D, 40176F, 3A6606, 182B6F,

          7D1201, 333F62, 77545F, 6E4F0C, 270262, 0A2759, 0D216F, 4F3C7A,

          444A62, 17292D, 222B6F, 7D1201, 125462, 08013E, 1B2D0C, 3F0006,
        hill_dec(mess, key_m, c_mess, v_mess, d_, el) :=
          Prog
            c_mess := ""
            Loop
              If DIM(mess) = 0 exit
              c_mess := APPEND(c_mess, FIRST(mess))
              mess := REST(mess)
            v_mess := []
#49:        Loop
              If DIM(c_mess) = 0 exit
              el := c_mess↓[1, 2]
              v_mess := APPEND(v_mess, [INVHX(el)])
              c_mess := DELETE(c_mess, [1, 2])
            d_ := (DIM(key_m))↓1
            mess := APPEND(MOD(key_m·redim(v_mess, DIM(v_mess)/d_, d_)', 127)')
            RETURN CTN(mess)

#50:    hill_dec(wiki_coded, dekey) = In classical cryptography, the Hill

          cipher is a polygraphic substitution cipher based on linear

          algebra. Invented by Lester S. Hill in 1929, it was the first
```

Encoding and decoding work. We will try a 5× 5 key-matrix:

#51: hillkeys(5) = $\left[\begin{bmatrix} 20 & 38 & 36 & 35 & 26 \\ 28 & 23 & 33 & 13 & 4 \\ 37 & 30 & 25 & 8 & 45 \\ 24 & 31 & 22 & 3 & 43 \\ 10 & 49 & 42 & 2 & 43 \end{bmatrix}, \begin{bmatrix} 10 & 103 & 125 & 73 & 70 \\ 4 & 101 & 73 & 71 & 18 \\ 36 & 52 & 124 & 99 & 102 \\ 17 & 19 & 12 & 78 & 113 \\ 31 & 81 & 88 & 106 & 71 \end{bmatrix}\right]$

#52: $\left[\text{key5} := \begin{bmatrix} 20 & 38 & 36 & 35 & 26 \\ 28 & 23 & 33 & 13 & 4 \\ 37 & 30 & 25 & 8 & 45 \\ 24 & 31 & 22 & 3 & 43 \\ 10 & 49 & 42 & 2 & 43 \end{bmatrix}, \text{dekey5} := \begin{bmatrix} 10 & 103 & 125 & 73 & 70 \\ 4 & 101 & 73 & 71 & 18 \\ 36 & 52 & 124 & 99 & 102 \\ 17 & 19 & 12 & 78 & 113 \\ 31 & 81 & 88 & 106 & 71 \end{bmatrix}\right]$

```
#53:  coded_mess5 := hill_enc(wiki, key5)

#54:  coded_mess5 := [6F6E070C72, 3E7A60291B, 247D591606, 105A04655C,

        14214C0606, 5C3B466F01, 1C6A097625, 4A01242135, 484C026F0E,

        5C72753F6F, 571B6F2971, 2D79387408, 2B37632158, 6F5E445B0C,
#55:  hill_dec(coded_mess5, dekey5) = In classical cryptography, the Hill
        cipher is a polygraphic substitution cipher based on linear
        algebra. Invented by Lester S. Hill in 1929, it was the first
        polygraphic cipher in which it was practical (though barely) to
        operate on more than three symbols at once. The following
        discussion assumes an elementary knowledge of matrices. (Wikipedia)
```

As you can see, there is no problem to decipher the message!!

References

Useful and interesting websites:

[4] http://math.slu.edu/~may/

https://en.wikipedia.org/wiki/Hill_cipher

http://www.dcode.fr/hill-cipher

http://crypto.interactive-maths.com/hill-cipher.html

http://www.nku.edu/~christensen/092mat483%20hill%20cipher.pdf

https://www3.amherst.edu/~tleise/Math272Spring2013/HillCode.pdf

Other resources are:

[1] Josef Böhm, *Matrizenrechnung mit DERIVE*
http://www.acdca.ac.at/material/t3/t3matrix.htm

[2] Brian H. Denton, *Learning Linear Algebra through DERIVE*, Prentice-Hall, 1995

[3] George T. Gilbert, Rhonda L. Hatcher, *Mathematics Beyond the Numbers*, Wiley, 2000

[5] Robert J. Hill, Thomas A. Keagy, *Elementary Linear Algebra with DERIVE*, Chartwell-Bratt, 1995

[6] Karsten Schmidt, Götz Trenkler, *Einführung in die Moderne Matrix Algebra*, Springer, 2006

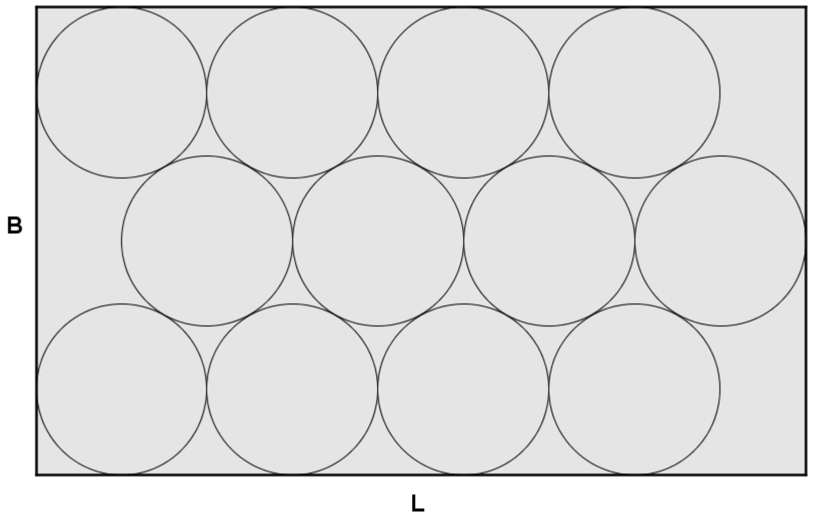
Mail from: Wolfgang Alvermann, Germany

Dear Josef,
on *Spiegel-Online* one can find regularly a mathematical problem of the week; one of them was motivation to produce a small problem for class room. It is attached.
Calculation is no problem at all, but what is the easiest way plotting the results?

Arranging Circles in a Rectangle

In accordance with a Spiegel-Online *Rätsel der Woche* (*Problem of the Week): Wenn Mathematiker Spaghetti kochen* (*When Mathematicians cook spaghetti*)[1] I produced the following problem for students:

In a pan (rectangle) 12 pizzas (radius r) are arranged like presented in the figure below:



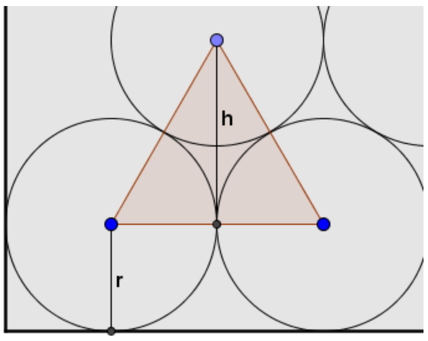
Tasks:

a) Find B and L dependending on the radius r of the pizzas!

b) Now take r/2 for the circles. How many pizzas (circles) can be arranged at most in the pan?

c) How does the sum of the areas of all pizzas change?

Solution a)



Read off: $\boxed{L = 9 \cdot r}$

For finding B we need the altitude in the equilateral triangle with side length 2r:

$$\left.\begin{array}{l} B = 2 \cdot (r+h) \\ h = r \cdot \sqrt{3} \end{array}\right\} \rightarrow \quad \boxed{B = 2 \cdot r \cdot \left(1 + \sqrt{3}\right)}$$

Solution b)

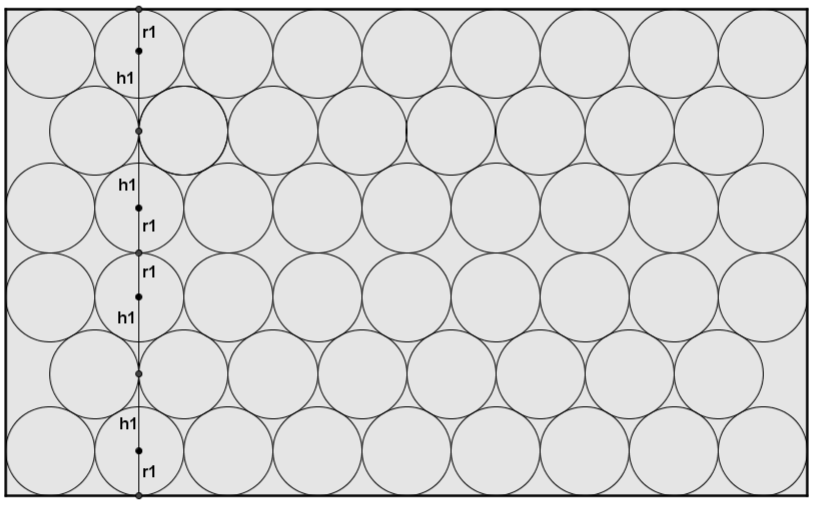
$r_1 = \frac{r}{2}$, i.e. that nine circles of $r_1 = \frac{r}{2}$ can be arranged in one row.

h_1 is now given by $h_1 = \frac{r}{2} \cdot \sqrt{3}$

For reaching the value of B from above we have to multiply both values by four.

$$B = 4 \cdot (r_1 + h_1) = 4 \cdot \left(\frac{r}{2} + \frac{r}{2} \cdot \sqrt{3}\right) = 2 \cdot r \cdot \left(1 + \sqrt{3}\right)$$

We need an arrangement formed by four rows of 9 pizzas and two rows of 8 pizzas in order to obtain the requested value for B.



This arrangement fulfills the conditions.

Solution c)

The total area of all pizzas in the first pan: $A_{tot} = 12 \cdot \pi \cdot r^2$

The total area of all small pizzas: $A_{tot} = 52 \cdot \pi \cdot \left(\frac{r}{2}\right)^2 = 13 \cdot \pi \cdot r^2$

The pizza area has increased by 8.33%.

Reference:

[1] http://www.spiegel.de/wissenschaft/mensch/vier-mathematiker-kochen-spaghetti-raetsel-der-woche-a-1109584.html 30.8.2016

Dear Wolfgang,

I could plot the pizzas using DERIVE. As you wrote, only the plotting is really of interest using technology.

Plotting the families of circles seems not to be so easy with TI-Nspire. I will try a LUA-script because unfortunately programming graphics (like with TI-92 and Voyage 200) has not been implemented (until now?).

Best regards
Josef

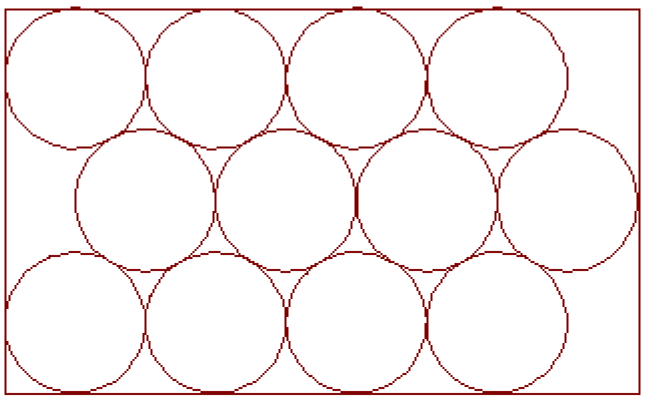
See first the DERIVE treatment:

One single VECTOR command delivers all the circles at once.

#1: $\text{pizzas\_12}(r) := \text{VECTOR}(\text{VECTOR}((x - r\cdot(2\cdot i - |j|))^2 + (y + j\cdot r\cdot\sqrt{3})^2 = r^2, i, 1, 4), j, 1, -1, -1)$

#2: $$\text{box}(r) := \begin{bmatrix} 0 & -r\cdot(1+\sqrt{3}) \\ 9\cdot r & -r\cdot(1+\sqrt{3}) \\ 9\cdot r & r\cdot(1+\sqrt{3}) \\ 0 & r\cdot(1+\sqrt{3}) \\ 0 & -r\cdot(1+\sqrt{3}) \end{bmatrix}$$

#3: [box(2), pizzas_12(2)]



Plotting the 52 pizzas is not so easy.

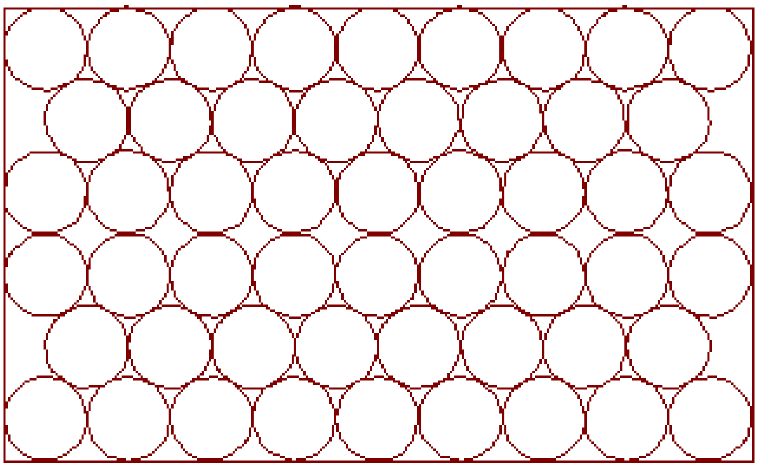
The circles are generated in two groups. Then we can put them together into one function (#6):

#4: $$\text{pizzas\_52\_1}(r) := \text{VECTOR}\left(\text{VECTOR}\left(\text{VECTOR}\left(\left(x - \frac{r}{2}\cdot(1 + 2\cdot i)\right)^2 + \left(y + \frac{(-1)^k\cdot r}{2}\cdot(1 + 2\cdot j\cdot\sqrt{3})\right)^2 = \frac{r^2}{4}, i, 0, 8\right), j, 0, 1\right), k, 1, 2\right)$$

#5: $$\text{pizzas\_52\_3}(r) := \text{VECTOR}\left(\text{VECTOR}\left(\left(x - \frac{r}{2}\cdot(2 + 2\cdot i)\right)^2 + \left(y + \frac{(-1)^j\cdot r}{2}\cdot(1 + \sqrt{3})\right)^2 = \frac{r^2}{4}, i, 0, 7\right), j, 1, 2\right)$$

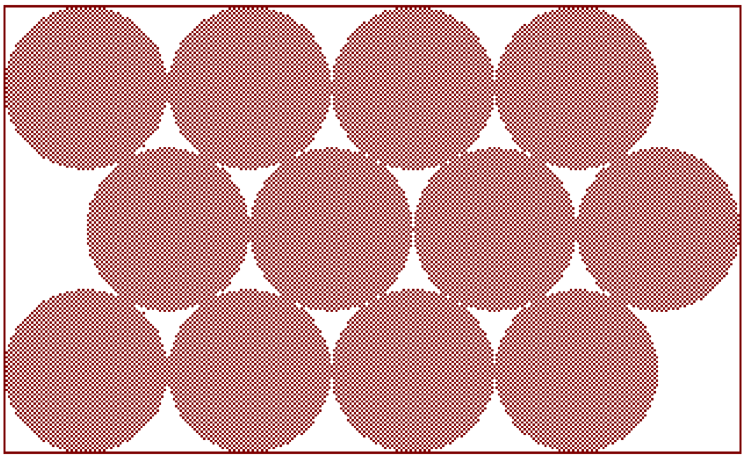
#6: pizzas_52(r) := APPEND(pizzas_52_1(r), pizzas_52_3(r))

#7: [pizzas_52(2), box(2)]



It seems to be that we forgot to have some toppings. Just replace the “=” by “≤” and you will have served the pizzas of your choice ☺.

```
#8:    f_pizzas_12(r) := VECTOR(VECTOR((x - r·(2·i - |j|))^2 + (y + j·r·√3)^2 ≤ r^2,

         i, 1, 4), j, 1, -1, -1)

#9:    [f_pizzas_12(2), box(2)]
```



But how to prepare the pizzas with TI-NspireCAS?

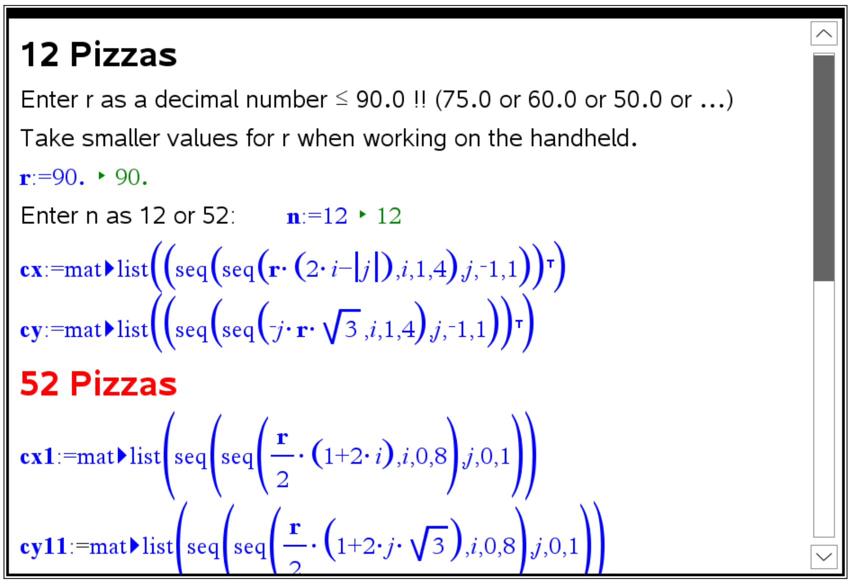
I will take a LUA recipe: The ingredients are provided in a Note Page. The “cooking” will be done in a LUA-stove.

I started with two Notes Pages, each of them connected with a LUA-script. So I had two Geometry Pages – one for 12 pizzas and the other one for 52 pizzas. It must be said that Steve Arnold gave very useful advice, specially how to use var.monitor and var.Change in the script. Thanks again Steve for your support.

I didn’t like the 2-Windows solution, so I put them together and worked with an if-construct in the script. And this was not too difficult.

The next page shows the results – the pizzas in a huge TI-NspireCAS-pan.

It is important to enter the radius as a decimal number. Otherwise you will fail. Then you have to enter r again (decimal number), open the script, Set Script, Focus Script.
Or you close the file without saving and reload it.



12 Pizzas

Enter r as a decimal number ≤ 90.0 !! (75.0 or 60.0 or 50.0 or ...)
Take smaller values for r when working on the handheld.

$\mathbf{r}:=90. \blacktriangleright 90.$

Enter n as 12 or 52: $\mathbf{n}:=12 \blacktriangleright 12$

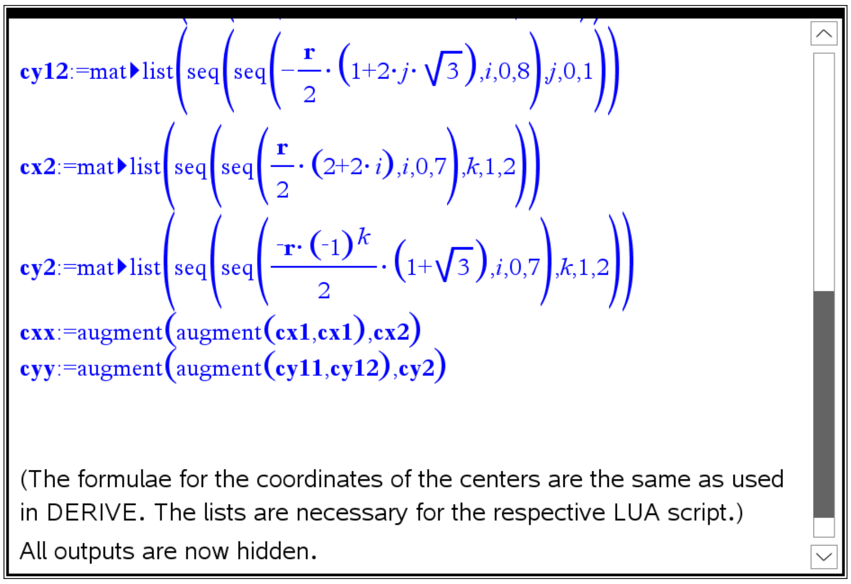
$\mathbf{cx}:=\text{mat}\blacktriangleright\text{list}\left(\left(\text{seq}\left(\text{seq}\left(\mathbf{r}\cdot\left(2\cdot i-|j|\right),i,1,4\right),j,^-1,1\right)\right)^T\right)$

$\mathbf{cy}:=\text{mat}\blacktriangleright\text{list}\left(\left(\text{seq}\left(\text{seq}\left(^-j\cdot\mathbf{r}\cdot\sqrt{3},i,1,4\right),j,^-1,1\right)\right)^T\right)$

52 Pizzas

$\mathbf{cx1}:=\text{mat}\blacktriangleright\text{list}\left(\text{seq}\left(\text{seq}\left(\frac{\mathbf{r}}{2}\cdot\left(1+2\cdot i\right),i,0,8\right),j,0,1\right)\right)$

$\mathbf{cy11}:=\text{mat}\blacktriangleright\text{list}\left(\text{seq}\left(\text{seq}\left(\frac{\mathbf{r}}{2}\cdot\left(1+2\cdot j\cdot\sqrt{3}\right),i,0,8\right),j,0,1\right)\right)$



$\mathbf{cy12}:=\text{mat}\blacktriangleright\text{list}\left(\text{seq}\left(\text{seq}\left(-\frac{\mathbf{r}}{2}\cdot\left(1+2\cdot j\cdot\sqrt{3}\right),i,0,8\right),j,0,1\right)\right)$

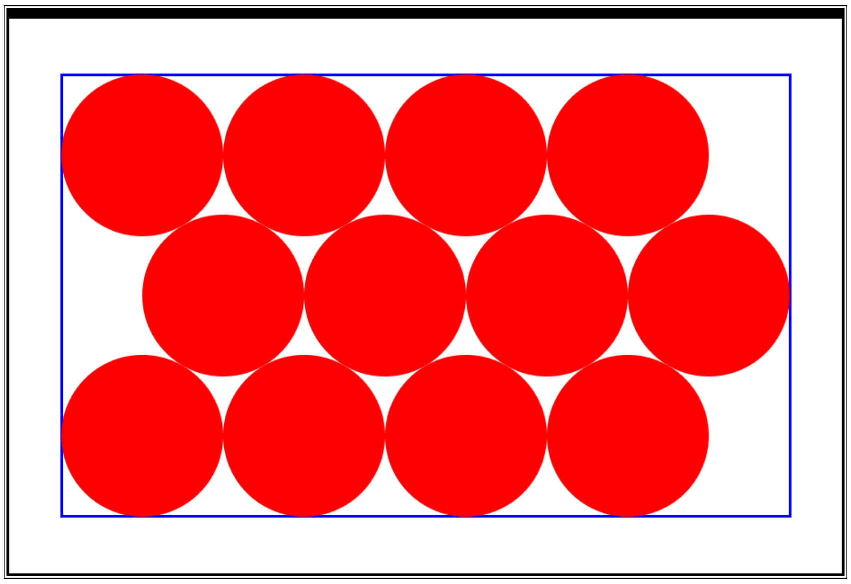
$\mathbf{cx2}:=\text{mat}\blacktriangleright\text{list}\left(\text{seq}\left(\text{seq}\left(\frac{\mathbf{r}}{2}\cdot\left(2+2\cdot i\right),i,0,7\right),k,1,2\right)\right)$

$\mathbf{cy2}:=\text{mat}\blacktriangleright\text{list}\left(\text{seq}\left(\text{seq}\left(\frac{^-\mathbf{r}\cdot\left(^-1\right)^k}{2}\cdot\left(1+\sqrt{3}\right),i,0,7\right),k,1,2\right)\right)$

$\mathbf{cxx}:=\text{augment}\left(\text{augment}\left(\mathbf{cx1},\mathbf{cx1}\right),\mathbf{cx2}\right)$

$\mathbf{cyy}:=\text{augment}\left(\text{augment}\left(\mathbf{cy11},\mathbf{cy12}\right),\mathbf{cy2}\right)$

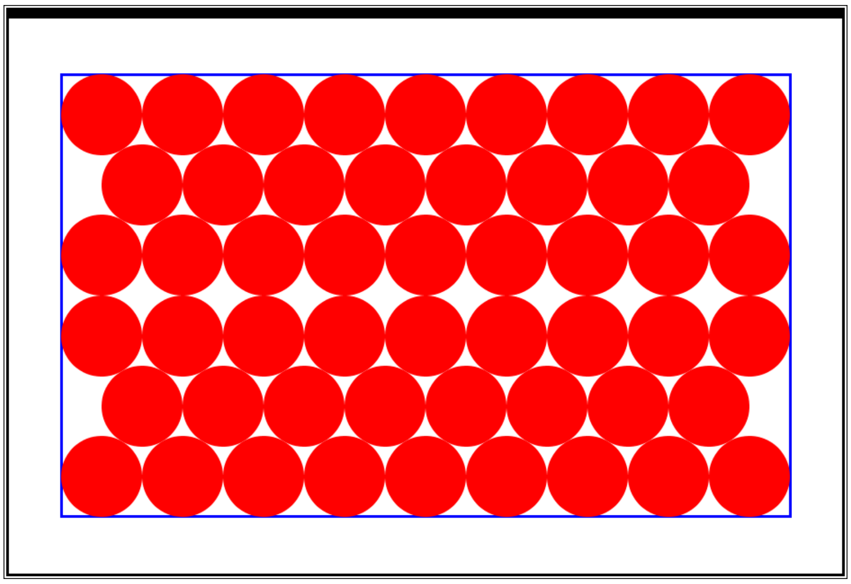
(The formulae for the coordinates of the centers are the same as used in DERIVE. The lists are necessary for the respective LUA script.)
All outputs are now hidden.



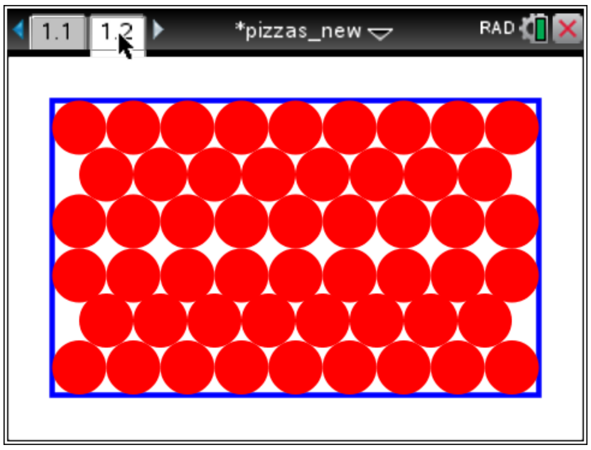
This is the LUA-script:

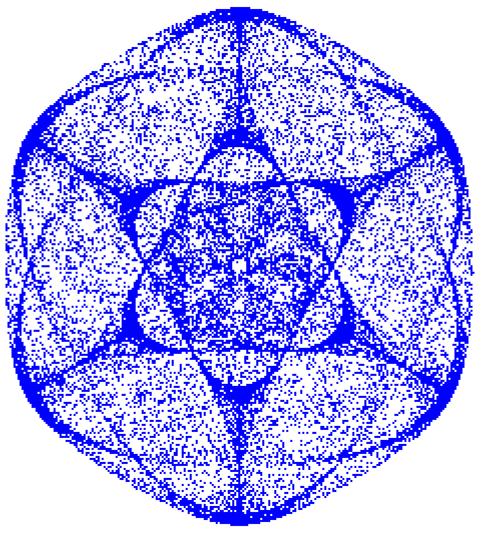
```
platform.apilevel='2.2'
local cxx=0; local cyy=0; local x0; local y0; local r3
local cx=0; local cy=0
function on.construction()
  h=platform.window:height(); w=platform.window:width()
end
var.monitor("n");var.monitor("cx");var.monitor("cy")
var.monitor("cxx"); var.monitor("cyy"); var.monitor("r")
function on.enterKey()
  platform.window:invaliditate()
end
function on.varChange()
  r=var.recall("r"); cxx=var.recall("cxx");cyy=var.recall("cyy")
  cx=var.recall("cx");cy=var.recall("cy");n=var.recall("n")
 --platform.window:invaliditate()
end
function on.resize(width,height)
  h=height; w=width; r=var.recall("r")
  cxx=var.recall("cxx"); cyy=var.recall("cyy")
  cx=var.recall("cx"); cy=var.recall("cy")
  n=var.recall("n")
end
cxx={}; cyy={}; cx={}; cy={}
 for i=1,52 do
   cxx[i]=0; cyy[i]=0
 end
 for i=1,12 do
   cx[i]=0; cy[i]=0
 end
function on.paint(gc)
   x0=w/2-9/2*r; y0=h/2; r3=math.sqrt(3)
   gc:setColorRGB(0,0,255); gc:setPen("medium","smooth")
   gc:drawRect(x0,y0-r*(1+r3),9*r,2*r*(1+r3))
   gc:setColorRGB(255,0,0)
   if n==52 then
     for i=1,52 do
       gc:fillArc(x0+cxx[i]-r/2,y0-cyy[i]-r/2,r,r,0,360)
     end
   else
     for i=1,12 do
       gc:fillArc(x0+cx[i]-r,y0-cy[i]-r,2*r,2*r,0,360)
     end
   end
end
```

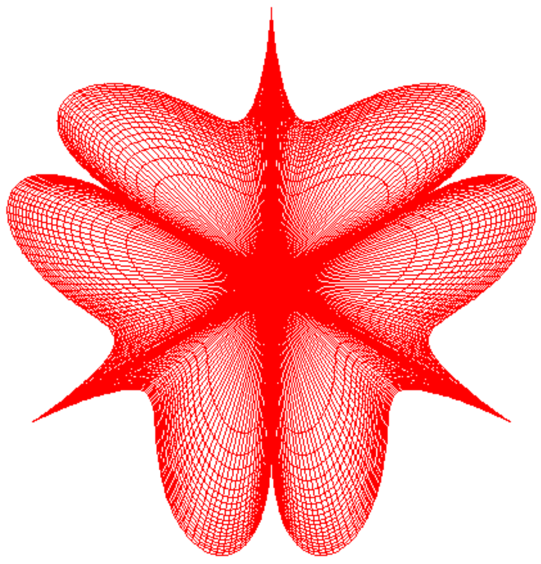
52 Pizzas with r = 90 (the pizzas have radius of 45) on the PC.



52 Pizzas with r = 30 on the handheld.







One of Golubitzy's symmetric attractors (mouthwatering for next DNLs?)

One of Norton Starr's Stars/Flowers

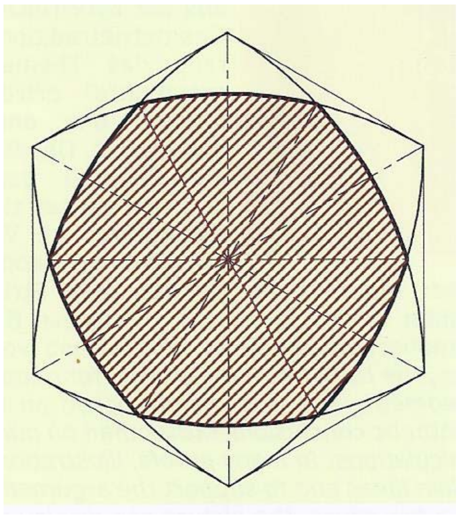
Two IBDG Problems – solved by CAS-Means

Josef Böhm, Würmla, Austria

Task 20.2 – medium

Two conguent REULEAUX-triangles are inscribed symmetrically in a regular hexagon with $r = 1$.

What is the area of the shaded area of intersection?



$$\#3:\quad \left[M1 := [1, 0],\ M2 := \left[\frac{1}{2}, \frac{\sqrt{3}}{2}\right],\ M3 := \left[-\frac{1}{2}, \frac{\sqrt{3}}{2}\right],\ M4 := [-1, 0],\ M5 := \left[-\frac{1}{2}, -\frac{\sqrt{3}}{2}\right],\ M6 := \left[\frac{1}{2}, -\frac{\sqrt{3}}{2}\right]\right]$$

$$\#4:\quad V := [M1, M2, M3, M4, M5, M6, M1]$$

$$\#5:\quad r := |M3 - M1|$$

$$\#6:\quad r = \sqrt{3}$$

$$\#7:\quad \text{VECTOR}\left(\left[V_{i,1} + r\cdot\text{COS}\left(\frac{3 + 2\cdot(i + t)}{6}\cdot\pi\right),\ V_{i,2} + r\cdot\text{SIN}\left(\frac{3 + 2\cdot(i + t)}{6}\cdot\pi\right)\right],\ i, 1, 6\right)$$

#7 generates the family of six arcs forming the figure. The “nice” thing is that for all arcs parameter *t* runs from 0 to 1.

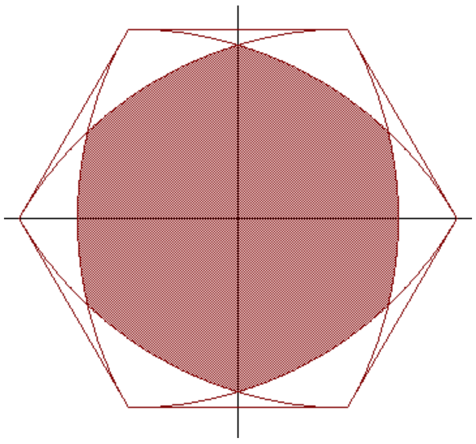
$$\#8:\quad \left[k1 := (x - V_{1,1})^2 + (y - V_{1,2})^2 \le r^2,\ k2 := (x - V_{2,1})^2 + (y - V_{2,2})^2 \le r^2\right]$$

$$\#9:\quad \left[k3 := (x - V_{3,1})^2 + (y - V_{3,2})^2 \le r^2,\ k4 := (x - V_{4,1})^2 + (y - V_{4,2})^2 \le r^2\right]$$

$$\#10:\quad \left[k5 := (x - V_{5,1})^2 + (y - V_{5,2})^2 \le r^2,\ k6 := (x - V_{6,1})^2 + (y - V_{6,2})^2 \le r^2\right]$$

$$\#11:\quad k1 \wedge k2 \wedge k3 \wedge k4 \wedge k5 \wedge k6$$

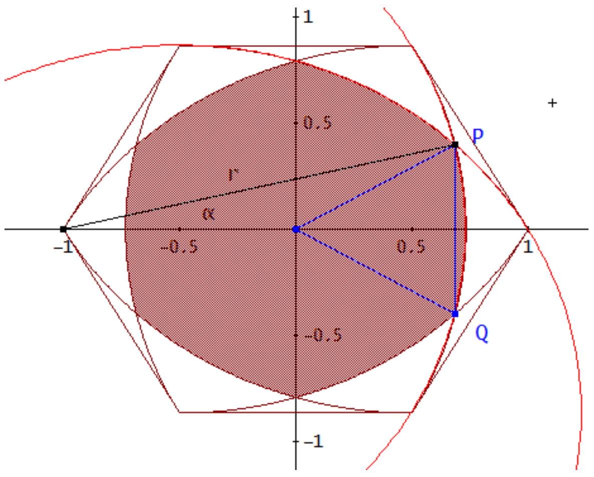
#11 plots the shaded area as intersection of the six circles defined in #8, #9 and #10..



#13: SOLUTIONS(k4_ ∧ k5_, [x, y]) = $\begin{bmatrix} \frac{\sqrt{33}}{4} - \frac{3}{4} & \frac{\sqrt{11}}{4} - \frac{\sqrt{3}}{4} \\ -\frac{\sqrt{33}}{4} - \frac{3}{4} & -\frac{\sqrt{11}}{4} - \frac{\sqrt{3}}{4} \end{bmatrix}$

#14: P := $\left[\frac{\sqrt{33}}{4} - \frac{3}{4}, \frac{\sqrt{11}}{4} - \frac{\sqrt{3}}{4}\right]$

#15: Q := $\left[\frac{\sqrt{33}}{4} - \frac{3}{4}, -\frac{\sqrt{11}}{4} + \frac{\sqrt{3}}{4}\right]$



#16: [[0, 0], P, Q, [0, 0]]

#17: [[-1, 0], P]

#18: $\left(\alpha := \mathrm{ATAN}\left(\frac{P_2}{1 + P_1}\right)\right) = \alpha := \frac{\mathrm{ATAN}\left(\frac{\sqrt{11}}{4}\right)}{3}$

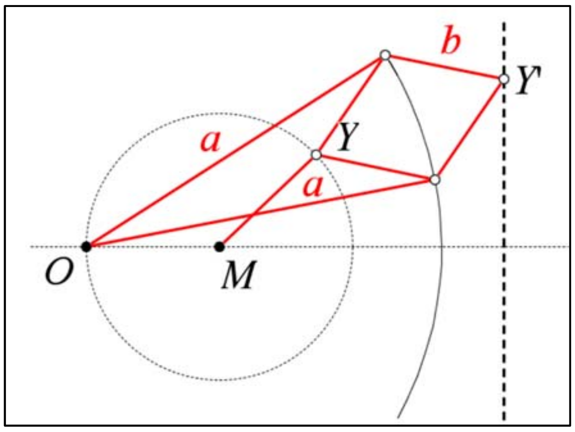
#19: $6\cdot\left(P_1\cdot P_2 + \frac{r^2}{2}\cdot(2\cdot\alpha - \mathrm{SIN}(2\cdot\alpha))\right) = 6\cdot\mathrm{ATAN}\left(\frac{\sqrt{11}}{4}\right) - \frac{3\cdot\sqrt{11}}{2} + \frac{3\cdot\sqrt{3}}{2}$

#20: $6\cdot\left(P_1\cdot P_2 + \frac{r^2}{2}\cdot(2\cdot\alpha - \mathrm{SIN}(2\cdot\alpha))\right) = 1.776747095$

Area = 1.77674

Task 20.3 – difficult

Which considerations are necessary to proof that the locus of *Y'* is a straight line when *Y* is moving on the circle *k*?



I worked through this task and – shame on me – I didn't realize that the very same construction was described and treated in DNL#36 (just revised) by R. Gossez and J. Sengier in their contribution "Peaucellier's Inversor" [3].

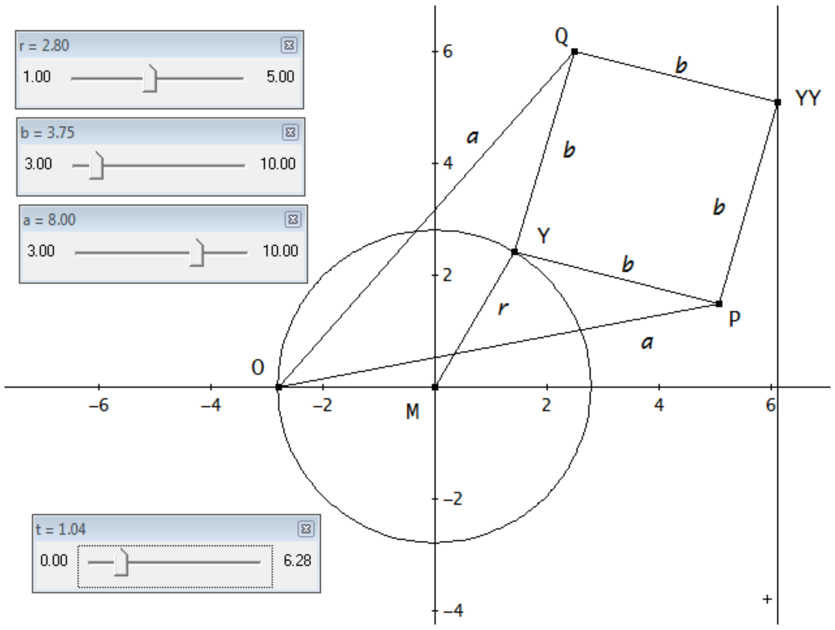
I left my treatment this issue because there are some other ideas for proof and representation including Phil Todd's remarks at the end of the article, thanks Phil. The TI-Nspire Dynamic Geometry-construction tracing Y' is right above. My DERIVE-proof is following.

```
#1:   CaseMode := Sensitive

                      2    2    2
#2:   circle1 := x  + y  = r

                                 2                    2    2
#3:   circle2 := (x - r·COS(t))  + (y - r·SIN(t))  = b

                        2    2    2
#4:   circle3 := (x + r)  + y  = a

#5:   Y := [r·COS(t), r·SIN(t)]

#6:   [O := [-r, 0], M := [0, 0]]

#7:   [M, Y]

#8:   sols := SOLUTIONS(circle2 ∧ circle3, [x, y])

      ⎡P := sols , Q := sols ⎤
#9:   ⎣          1          2⎦

      ⎡ Y  P ⎤
      ⎢      ⎥
      ⎢ Y  Q ⎥
#10:  ⎢      ⎥
      ⎢ O  P ⎥
      ⎢      ⎥
      ⎣ O  Q ⎦

                          2          2    2
#11:  circle4 := (x - P )  + (y - P )  = b
                       1          2

                          2          2    2
#12:  circle5 := (x - Q )  + (y - Q )  = b
                       1          2

#13:  sols2 := SOLUTIONS(circle4 ∧ circle5, [x, y])
```

In #13 we calculate the intersection points of two circles with radius *b*. YY is one of the intersection points. We hope to obtain point YY which is given as Y' in the sketch.

Then we try to find its locus.

As you can see, I introduce sliders for *r*, *a*, *b* and for the parameter *t* which navigates point Y on the circle.



#14: sols2 :=
$$\begin{bmatrix} \dfrac{a^2 - b^2 - 2\cdot r^2}{2\cdot r} & \dfrac{(a^2 - b^2)\cdot \mathrm{TAN}\left(\dfrac{t}{2}\right)}{2\cdot r} \\ \dfrac{r\cdot(3\cdot \mathrm{COS}(t)^3 + 5\cdot \mathrm{COS}(t)^2 + \mathrm{COS}(t) + \mathrm{SIN}(t)^4 - 1)}{\mathrm{COS}(t)^3 + 3\cdot \mathrm{COS}(t)^2 + 3\cdot \mathrm{COS}(t) + 1} & r\cdot \mathrm{SIN}(t) \end{bmatrix}$$

#15: Trigpower := Cosines

#16: sols2 :=
$$\begin{bmatrix} \dfrac{a^2 - b^2 - 2\cdot r^2}{2\cdot r} & \dfrac{(a^2 - b^2)\cdot \mathrm{TAN}\left(\dfrac{t}{2}\right)}{2\cdot r} \\ r\cdot \mathrm{COS}(t) & r\cdot \mathrm{SIN}(t) \end{bmatrix}$$

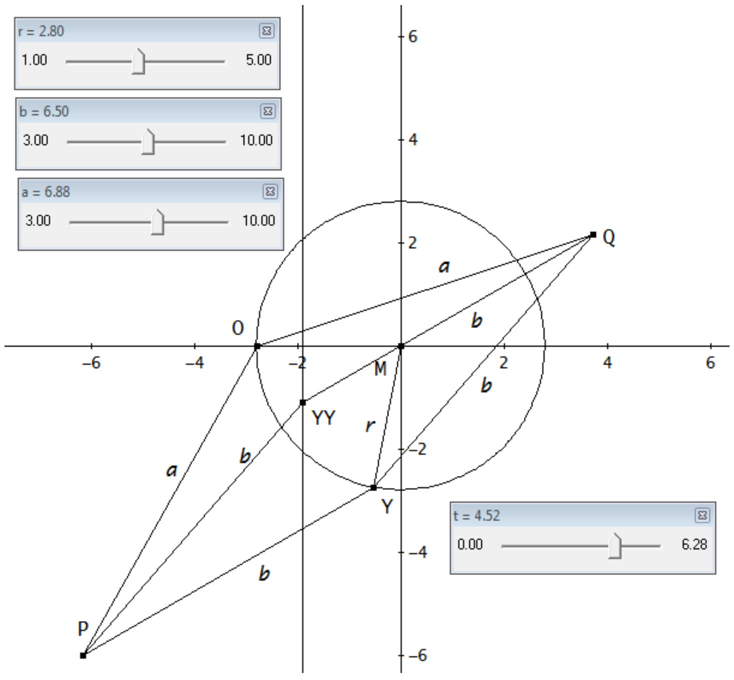
#17: YY :=
$$\left[\dfrac{a^2 - b^2 - 2\cdot r^2}{2\cdot r}, \dfrac{(a^2 - b^2)\cdot \mathrm{TAN}\left(\dfrac{t}{2}\right)}{2\cdot r}\right]$$

#18: [P, YY, Q]

#19:
$$\left[\dfrac{a^2 - b^2 - 2\cdot r^2}{2\cdot r}, \dfrac{(a^2 - b^2)\cdot \mathrm{TAN}\left(\dfrac{s}{2}\right)}{2\cdot r}\right]$$

Parameter *t* is reserved for the slider. So I substitute *s* for *t*.
Now I take -1 still hoping to obtain the requested result:
I was successful: this is the parameter representation (#19) of a straight line parallel to the *y*-axis.

It can be plotted (page 36). See here the case for $a < b$:



Moving the sliders enables some automation.

I like this interplay between algebra and geometry. At TIME 2006 in Dresden I offered a workshop on this objective [2].

[1] IBDG – Informationsblätter der Geometrie, Heft 1/2016, Jahrgang 35

[2] Narrowing the Gap between Computer Algebra and Dynamic Geometry,
Proceedings of TIME 2006, http://rfdz.ph-noe.ac.at/acdca/acdca-conferences.html

[3] Renée Gossez & Jacqueline Sengier, *Peaucellier's Inversor*, DNL#36, 1999

Possible additional tasks:

(1) What is the locus in case $a = b$?

(2) The "real locus" is not the full straight line, it is only a segment. Why? Find the end points of this segment.

I wrote to Phil Todd from *Geometry Expressions* about my problems performing the proof with his program. This is his answer:

Hi Joseph,

Good to hear from you. This is interesting, as I am sure I remember this model giving a nice result in an earlier version of GX. Sometimes the stuff you do to fix errors in algebra system make it more conservative in what it is prepared to simplify, sometimes you just shake it into a less happy course.

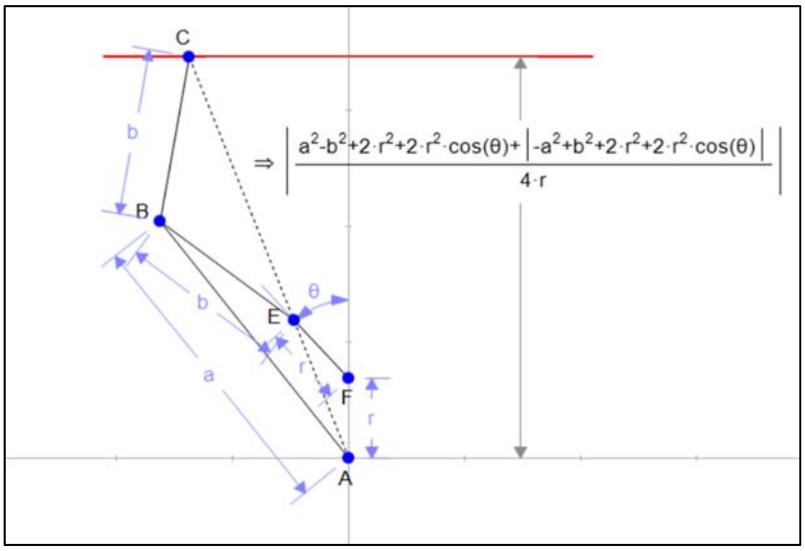
Anyway, I find I cannot directly get a good answer.

However, if I am prepared to cut the mechanism in half, draw in the line OY' and then insist that the symmetry forces U to lie on OY', I can coerce GX to give me a nice answer (I am measuring the distance of point C in my diagram from the x axis, and want it to be independent of theta.. I do have to turn on Use Assumptions, which gets rid of the absolute values in the solution.

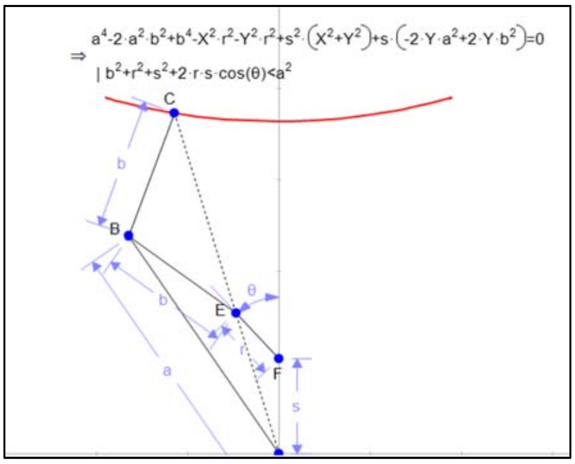
Then I can move your point O off the circle through Y to get a curved locus, which I can show to be a circle. By creating the circle with this equation, I can find its center and radius.

I had a class of teachers make one of these linkages this summer. I will forward a short video.

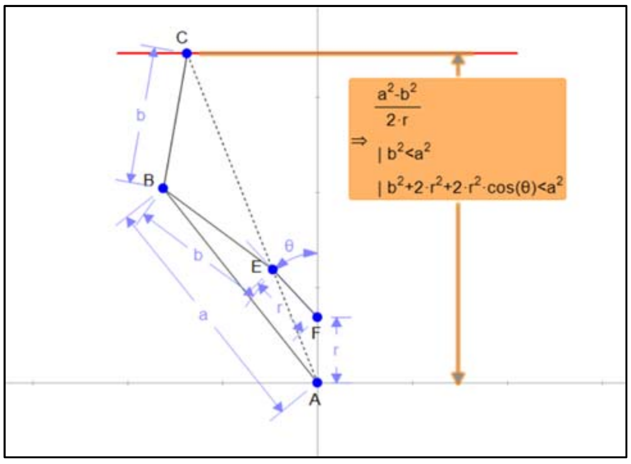
Phil



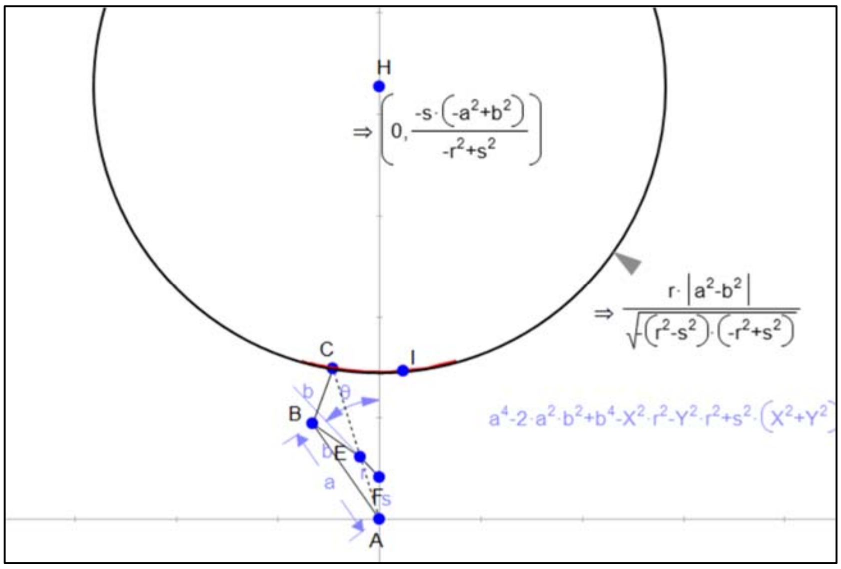
$$\Rightarrow \left|\frac{a^2-b^2+2\cdot r^2+2\cdot r^2\cdot\cos(\theta)+\left|-a^2+b^2+2\cdot r^2+2\cdot r^2\cdot\cos(\theta)\right|}{4\cdot r}\right|$$



$$\Rightarrow a^4-2\cdot a^2\cdot b^2+b^4-X^2\cdot r^2-Y^2\cdot r^2+s^2\cdot\left(X^2+Y^2\right)+s\cdot\left(-2\cdot Y\cdot a^2+2\cdot Y\cdot b^2\right)=0$$
$$\mid b^2+r^2+s^2+2\cdot r\cdot s\cdot\cos(\theta)<a^2$$



$$\Rightarrow \frac{a^2-b^2}{2\cdot r}$$
$$\mid b^2<a^2$$
$$\mid b^2+2\cdot r^2+2\cdot r^2\cdot\cos(\theta)<a^2$$



$$\Rightarrow \left(0, \frac{-s\cdot\left(-a^2+b^2\right)}{-r^2+s^2}\right)$$

$$\Rightarrow \frac{r\cdot\left|a^2-b^2\right|}{\sqrt{\left(r^2-s^2\right)\cdot\left(-r^2+s^2\right)}}$$

$$a^4-2\cdot a^2\cdot b^2+b^4-X^2\cdot r^2-Y^2\cdot r^2+s^2\cdot\left(X^2+Y^2\right)$$