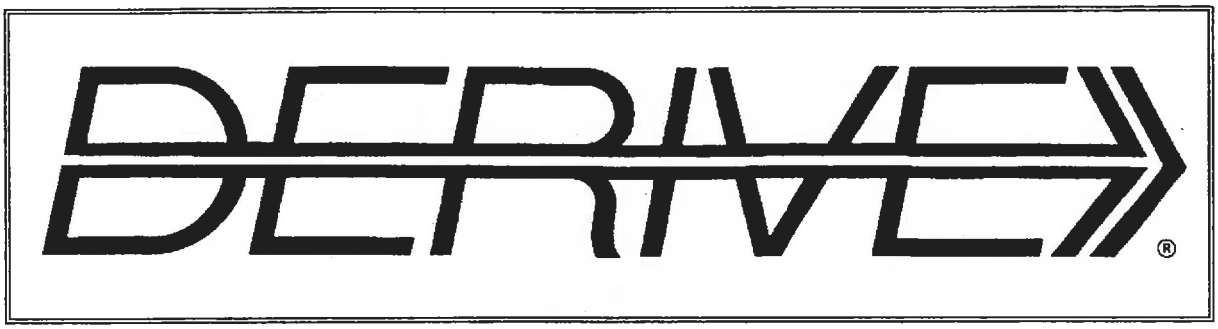
THE DERIVE - NEWSLETTER #42

THE BULLETIN OF THE



USER GROUP

+TI 92

Contents:

June 2001

[1] **Funktionen zweier Variablen untersuchen mit *DERIVE* und DPGraph**, R. Wukonisch, bk-teachware 2001, ISBN 3-901769-35-8

[2] **Prüfungsaufgaben für das Unterrichten mit *DERIVE* und dem TI-89/92/92+ (Band 2),** V. Kokol-Voljc & B. Kutzler, bk-teachware 2001, ISBN 3-901769-39-0

[3] **Math -Trainer I - Practice makes Perfect,** Josef Böhm, bk-teachware 2001, to appear

[4] **Abituraufgaben mit Graphikrechnern und Taschencomputern,** H.Knechtel & W. Weiskirch (Hrsg), Schroedel 2001, ISBN 3-507-73237-8

[5] **Wie die Mathematik in die Umwelt kommt,** H.-G. Weigand (Hrsg), Schroedel 2001, ISBN 3-507-73236-X

[6] **Gleichungen in der Sekundarstufe 1 mit dem TI-92,** W. Herget & E. Lehmann (Hrsg), Schroedel 2001, ISBN 3-507-73232-7

Interesting WEB sites http://.....

www.kutzler.com	Bernhars Kutzler´s personal web site
www.bk-teachware.com	Bernhard Kutzler´s Online Shop
www.acdca.ac.at	Austrian Center of Didactics of Computer Algebra & T^3 Austria
www.mathguide.de	over 1000 records, mainly in the area of pure mathematics
www.equate.org.uk	How to create dynamic, interactive learning resources (HTML 4.0)
www.langara.bc.ca/mathstas/resource/onWeb/catalogues	Internet Resources for the Mathematics Student
www.univie.ac.at/future.media/mo/	University of Vienna, mathe - online
www.vts.rdn.ac.uk	Subject focused Tutorials covering key information skills for the mathematics Internet environment
members.vol.at/edis_corner	Eduard Engler´s home page (German)
www.dessci.com/webmath/	Dynamic Math on the Web
www.mathtv.com/Trig/pages/toc.htm	Trigonometry video lessons online

Our friend Paul Drijvers from the Netherland would like to inform you all about the

CAME 2001 Symposium

Communicating Mathematics through CAS

18 - 19 July in Utrecht

http://ltsn.mathstore.ac.uk/came/events/freudenthal/

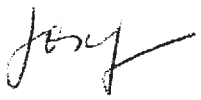
Dear friends,

As you can see below, this page is full of information. So I cannot write an extended letter this time.

You can find the program of our ICTCM5 Special Group - it is the biggest group of all - and we had to ask the Conference organizers for more time as provided. Many thanks to all who contributed and we are sure to have interesting sessions.

Another highlight and fixpoint in your calender should be July 2002. You are friendly invited to **VISIT-ME** (= Vienna International Symposium on Integrating Technology into Mathematics Education) next summer. You find more information on the last two pages of this DNL issue.

For this summer my wife and I wish you a wonderful time and we hope to meet again in fall.



Tentative Program of Special Group 1: DERIVE, TI-92 and other CAS

ICTMT5, Klagenfurt, 6 - 9 August 2001

chaired by: Vlasta Kokol-Voljc, Bernhard Kutzler & Josef Böhm

Carl Leinbach (US): Using a CAS to Teach Algebra - Going Beyond the Manipulation
Yungkook Jun (AUT): Theorema based TI-92 Simulator for Exploratory Learning
Guido Herweyers (BEL): Elimination of Parameters and Substitution with CA
Hans Dirnböck (AUT): The Evolvente Curve of a Circle used for Gear-Wheels
John Cosgrave (IRL): Fermat´s "Little" Theorem
Josef Lechner (AUT): Standardization of Normal Distribution - An Anachronism?
Wilhelm Weiskirch (GER): Ortskurven - Loci
Detlev Berntzen (GER): Movies from the TI-92
Ernö Scheiber (ROM): Teaching Runge-Kutta Type Methods Assisted by CAS
Vlasta Kokol-Voljc (SLO): Technology in Pre-Service Teacher Training
Heiko Knechtel (GER): Mathematics with Graphic and Symbolic Calculators
Otto Wurnig (AUT): Advantages and Dangers in the Teaching of Stochastics by using CAS
Alex Lobregt (NED): Introducing Fourier Series with DERIVE (Workshop)
Josef Böhm (AUT): From Pole to Pole
Karl-Heinz Keunecke (GER): Curvature of Functions as an Aid of Treating Sequences, ...
Rolf Wasen (SWE): Computers in Engineering Education
Wolfgang Pröpper (GER): The TI-92 as a Tool for Analytic Geometry
Halil Ardahan (TUR): Issues on Integrating CAS in Teaching Mathematics
Mykola Kolodnytky (UKR): Teaching Elementary Number Theory with a Software System

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

Many thanks to Walter Wegscheider from the ACDCA and Benjamin Kaineder from bk-teachware for their cooperation in updating the respective pages.

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: September 2001
Deadline 15 August 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
Flatterbandkurven, Rolfs, GER
Basic Conceptions on Recursion Theory, Urrego, COL
A Mistake presenting Graphs on the TI, Himmelbauer, AUT
Comparing statistics tools: a pie chart with DERIVE, a stem & leaf diagram on the TI,
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

Bernhard Kutzler, Austria

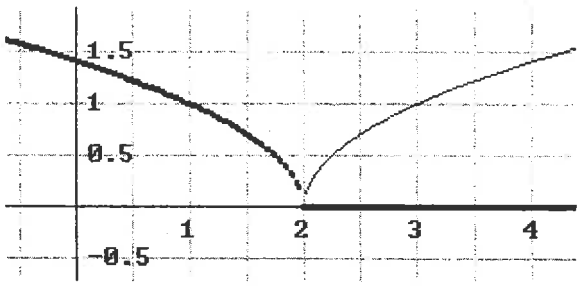
DERIVE gives 0 as the right-sided limit of SQRT(x-2) as x approaches 2. *DERIVE* gives 0 also as the left-sided limit of SQRT(x-2) as x approaches 2. According to the real graph this seems wrong – and teachers are not happy with this.
I guess it is again the complex numbers.... Do you have a good answer for teachers?

Al Rich, Hawaii

Hi Bernhard,
If you plot SQRT(x-2) with the "Plot real and imaginary parts" switch on, you will see that both the real AND imaginary parts approach 0 as x approaches 2 from the left. Therefore, the left-sided limit of SQRT(x-2) is 0 $*\pi$, which in turn simplifies to 0.

In *DERIVE*, SQRT(x-2) is NOT undefined if x is less than 2, it is imaginary. Similarly, LN(x) is NOT undefined if x is less than 0, it is imaginary or complex.

Aloha, Albert.



Al Rich had some comments on David Leigh´s web site (recommended in the last DNL). I think that his proposals are very valueable for all of you which try to use DfW5´s programming capabilities.

I took a brief look at David Leigh's web site, and I want to congratulate him on laying the ground-work for a friendly introduction to Derive 5 programming. However, as you pointed out, there is still much work to be done to make it a comprehensive tutorial. The following are my initial comments and suggestions on the "Basic Programming Functions and Operators" page:

1. Change "the Declare menu option" to "the Declare > Function command". The latter syntax is used consistently throughout the Derive on-line help and the "Introduction to Derive 5" book.
2. I take issue with the statement "It is also possible to define functions through the Declare menu option, but this is not recommend (sic) for writing substantial programs." The Declare > Function command has the advantage that the function name and arguments are on a separate line than the definition. Also, any existing definition for the function can be displayed and edited by clicking on the pull-down menu. Thus I think users should be encouraged to use this command.
3. Change the sentence "Notice that function names can be written and displayed in lower and mixed case, this was not possible before version 5." to "Notice that function names can be written and displayed in lower and mixed case. Before version 5 this could only be done by switching to case-sensitive input mode."
4. Change all references to "the author bar" to "the expression entry toolbar" or "the expression entry line". To be precise, "the expression entry toolbar" refers to the entire dockable toolbar, including the 5 buttons at its left end. "The expression entry line" refers to the field on the toolbar where mathematical expressions are actually entered.
5. Terminating expressions with an = sign was a hack introduced in version 3 of Derive that should NOT be encouraged (in fact it may be eliminated in future versions). Its use results in an equation which is not easily used in subsequent calculations. Therefore I recommend changing the sentence "Alternatively, enter Square(6)= in the author bar and enter." to "Alternatively, type Square(6) on the expression entry line, and press Ctrl+Enter or click on the <author and simplify> button to the left of the entry line." where the actual icon is used for the <author and simplify> button.

6. Instead of "function and program mean the same thing" I would say "in Derive a program is a collection of function definitions".
7. The following statements are false: "Also, the assignments for x and y are only applied locally. That is, the assignments are only valid for the duration of the PROG(). In other words, when the PROG() has finished x and y are not assigned the values of 1 and 2 in the current algebra window." Rather, it is extremely important that programmers realize that such variables are NOT local.
8. Rather than calling PROG and LOOP "functions", I prefer calling them "control constructs", since they do not necessarily simplify all their arguments once, as functions do, and then return a value.

Hope this helps gives you and David some ideas for your web page.

Aloha, Albert

Another discussion with Albert Rich and Nurit Zehavi being involved:

Nurit Zehavi, Israel Nurit.Zehavi@weizmann.ac.il

Subject: more on abs(X)

>1. I am currently upgrading the Derive-based books for jhs to the Windows version. I have problems with the abs-value in relations.

> I am aware that Derive sees x in |x| as complex.

No. When Derive simplifies an expression involving the variable x, it uses the declared domain of x as set by the Declare > Variable Domain command. The default domain of variables is real.

> Therefore the solution of an equation like |x-1|=3 is complex, and when solving the inequality |x-1|<3, x has to be interpreted as real. But, this confuses the students.

In Derive, when you solve an equation or inequality for a variable, the variable is assumed to be a dummy variable, just like the integration variable in definite integrals. Thus while solving equations or inequalities, external declarations of the dummy variable are temporarily ignored. Instead the variable is assumed to be complex when solving equations, and real when solving inequalities.

> Moreover, when you ask Derive to plot the relations |x-1|= 3 |x-1|<3 , x is >interpreted as real.

When you plot an expression in a 2D-plot window, the independent variable (in this case x) is plotted over the range of real values along the horizontal axis range. However, this does NOT in any way imply that this range of values is the entire domain of the independent variable.

> You suggest to choose the "solve for real" option -but then it appears on the screen, and we must say something (apologize) to students and to the teachers.

I am sorry you have to apologize for the correct answers that Derive is returning. We are considering possibly having the option for a "Real" mode of operation for Derive 6 (in addition to the current "Complex" mode).

However, I am not convinced that this is a good idea. For example, should the expression SQRT(-1) simplify to "?" in the "Real" mode of operation?

>2. I trust that you are aware that in 5.03 adding the vectors [0,0]+[0,0] yields "?" It is ok in 5.04

Yes, it was found and fixed for version 5.04 of Derive 5.

>3. 2DPlotting graphs of inequalities: when plotting the union of two relations, the inersection part is covered by the color of ONE of the relations. Do you plan to change the coloring scheme?

Thank you for pointing out this anomaly. It has been resolved for the next release of Derive 5.0x.

Theresa Shelby, Hawaii

Hi Josef,
How are you and Noor? We are enjoying nice sunny weather in Honolulu - not enough rain, but hopefully that will change. Thank you for the recent March 2001 DNL#41. I look forward to reading about all the interesting ways people use Derive.

Thank you also for sharing your findings in the article "Importing Data to DERIVE and the TIs". I am asked this question somewhat frequently and hadn't even considered the issue with commas representing decimal points in the European style. I have a couple of comments that hopefully will make the procedure easier. On page 38 it reads: "Remove all titles - they are no numerical data - and substitute the tab-spaces by commas to obtain: ...".

1) Instead of removing them, consider preceeding a title line with a semi-colon (";") - that will cause it to be read as an annotation to the subsequent data expression.

2) It is not necessary to convert the tab-spaces to commas. Derive 5.0x should be able to parse a data file where the values are separated by tabs-spaces OR commas. Let me know if you find otherwise.

3) Albert and I have recently done some work to improve Derive's File > Load > Data command. Derive 5.05 will allow commas separating numbers in data files to be preceeded and/or followed by spaces. Please let us know if there is anything else we can do to make it easier to import data files.

Lou Klauder, USA

Currently Derive gives the result of symbolic integration of a piecewise continuous function as a single expression valid everywhere and incorporating a multiplicity of unit-step functions. While the expression presented is correct, it does not make it easy to see what formula applies inside each zone. It would be a help for people who are looking for symbolic (as opposed to numerical) results if there were an option to get Derive to show the result of integration of a piecewise continuous function zone by zone without reliance on unit-step functions.

Al Rich

Hello Lou Klauder, Derive returns antiderivatives as a single expression valid over the entire real line if the integration variable is real (over the entire complex plane if the integration variable is complex). After determining this general antiderivative, if you would like to see what it looks like in a particular interval (i.e. "zone"), declare the integration variable to be in the interval using the Declare > Variable Domain command. Then, simplify the antiderivative. Hope this helps.

See an example, Josef:

$$x :\varepsilon \text{ Real}$$

$$f(x) := \left(-\frac{x}{2}+1\right)\cdot \text{CHI}(-\infty, x, 0) + \left(1+\frac{x}{2}\right)\cdot \text{CHI}(0, x, 4) + \left(5-\frac{x}{2}\right)\cdot \text{CHI}(4, x, \infty)$$

$$\int f(x)\, dx = \frac{(4 - x)\cdot|x - 4|}{4} + \frac{x\cdot|x|}{4} - \frac{x\cdot(x - 12)}{4}$$

$$x :\varepsilon \text{ Real } (-\infty, 0)$$

$$\int f(x)\, dx = -\frac{x^2 - 4\cdot x - 16}{4}$$

$$x :\varepsilon \text{ Real } [0, 4]$$

$$\int f(x)\, dx = \frac{x^2 + 4\cdot x + 16}{4}$$

$$x :\varepsilon \text{ Real } [4, \infty)$$

$$\int f(x)\, dx = -\frac{x^2 - 20\cdot x + 16}{4}$$

Re: Spread of Infection

By *Stefan Welke*, Bonn Germany, Spwelke@aol.com

1. The Problem

We consider the following simple infection model [1]:

- The population consists of N individuals which are not infected at the time $t = 0$.
- After the first period Δt one individual is infected.
- An infected individual infects one other individual, infected or not, or itself within one period with probability $p = \frac{1}{N}$.

Now two questions arise:

- **(A)** Which is the average number of infected individuals after n periods?
- **(B)** Which is the average number of periods until all N individuals are infected?

We shall give a complete answer to both questions together with the necessary *DERIVE* functions to calculate numerical results.

2. Transition probabilities

The key to the solution of our problem is the knowledge of the following transition probabilities:

Definition 1: Let $n \le N$ individuals been infected and let $0 \le k \le N$. Then $p_N(n,k)$ denotes the conditional probability that k individuals are infected after the next period.

Clearly $p_N(n,k) = 0$ for $0 \le k < n$.

It proves to be useful to consider infection as a function $f : S_n \longrightarrow S_N$ between sets $S_n := \{a_1, a_2, ..., a_n\} \subseteq S_N := \{a_1, a_2, ..., a_n, ..., a_N\}$, where S_N is the set of all individuals and S_n is the subset of infected individuals. So $f(a_k) = a_m$ simply means: a_k infects a_m. The advantage of this approach is that the calculation of $p_N(n,k)$ reduces to counting particular functions between finite sets because each way of infection respectively function has the same probability.

Now let $0 \le n \le k \le \min(2n, N)$ and let $S_n \subseteq S_k \subseteq S_N$ with subsets S_n, S_k as above. We want to determine the number of different ways in which the n individuals of S_n infect the $k-n$ individuals from S_k or equivalently: How many functions $f : S_n \longrightarrow S_k$ with the additional property $\{a_{n+1}, ..., a_k\} \subseteq f(S_n)$ do exist? We denote the set of all such functions with $F(n,k)$. Each function $f \in F(n,k)$ can be decomposed in a surjective part $f_s : S_K \longrightarrow \{a_{n+1}, ..., a_k\}$, where S_K is a particular subset of S_n with exactly $K \ge k-n$ elements, and a remaining part $f_r : S_n \longrightarrow S_n$. If $s(K, k-n)$ is the number of surjective functions as above, then the following lemma holds:

Lemma 1: $|F(n,k)| = \sum_{K=k-n}^{n} \binom{n}{K} \cdot s(K,k-n) \cdot n^{n-K}$

Proof: For a particular set S_K the number of remaining elements of S_n is $n-K$. Thus there are n^{n-K} functions $f_r : (S_n - S_K) \longrightarrow S_n$. There are $\binom{n}{K}$ choices for the set S_K and the number K of elements varies from $k-n$ to n. □

The desired formula for the transition probability follows easily from lemma 1:

Lemma 2: $p_N(n,k) = \frac{\binom{N-n}{k-n}|F(n,k)|}{N^n} = \frac{(N-n)!}{N^n (k-n)!(N-k)!} \sum_{K=k-n}^{n} \binom{n}{K} s(K,k-n) \cdot n^{n-K}$

Proof: The number of possible choices for the $k-n$ individuals to become infected is $\binom{N-n}{k-n}$, and the number of all possible functions $f : S_n \longrightarrow S_N$ is N^n.□

We need to know how to calculate the numbers $s(K,k-n)$ in order to get an effective formula for $p_N(n,k)$. For that reason we define the Stirling numbers of the second kind:

Definition 2: Let $m \geq k$ be natural numbers. Then $S2(m,k)$ is the number of different ways a set with m distinct elements can be portioned into k disjoint, non-empty subsets. The numbers $S2(n,k)$ are called Stirling numbers of the second kind.

The relevance of the Stirling numbers for our problem is that they are connected with the numbers $s(K,k-n)$, and that an explicit formula for Stirling numbers exists. This is the content of the following lemma:

Lemma 3

- (1) $S2(m,k) = \sum_{\mu=0}^{k} (-1)^{k-\mu} \binom{k}{\mu} \frac{\mu^m}{k!}$
- (2) For $K \geq k-n$ the following identity holds: $s(K,k-n) = S2(K,k-n) \cdot (k-n)!$

Proof: (1) can be found in [3],p.180. (2) Every surjective function f, which maps a finite set of K elements onto a finite set $\{b_j\}_{j=1,\ldots,k-n}$, defines a partition of the domain of definition into $k-n$ disjoint subsets $\bigcup_{j=1}^{k-n} f^{-1}(b_j)$. Two functions that differ only by a permutation of the target set define the same partition. $S2(K,k-n)$ is by definition the number of different partitions of a set with K elements into $k-n$ disjoint subsets. Thus the number of surjective functions $s(K,k-n)$ equals $S2(K,k-n) \cdot (k-n)!$.□

Theorem 1

Let $0 \le n \le N$ and $0 \le k \le N$, then: $p_N(n,k) = \left(\frac{n}{N}\right)^n \frac{(N-n)!}{(N-k)!} \sum_{\mu=k-n}^{n} \binom{n}{\mu} \frac{S2(\mu,k-n)}{n^\mu}$

Proof: A combination of the formulae of the previous lemmata yields the result. □

Now we have done the hardest part of the work. The following *DERIVE* code presents our results:

```
      KRONECKER(i,j):=
        If i=j
#1:        1
           0
```

$$\#2:\quad S2(m,k) := \sum_{\mu=0}^{k} \frac{(-1)^{k-\mu} \cdot \mathrm{COMB}(k,\mu) \cdot \mu^m}{k!}$$

```
#3:   S2(6, 3) = 90
```

Remark: J. Wiesenbauer presented in [4], p.20 a much faster version of $S2(m,k)$. We keep this simple code because the speed of calculating this function does not harm our later results.

$$\#4:\quad p(n,k,m) := \frac{\left(\frac{k}{n}\right)^k \cdot (n-k)!}{(n-m)!} \cdot \sum_{i=m-k}^{k} \frac{S2(i,m-k) \cdot \mathrm{COMB}(k,i)}{k^i}$$

$$\#5:\quad p(5,2,3) = \frac{3}{5}$$

In this notation `p(n, k, m)` equals $p_n(k,m)$. We define the **transition Matrix** $TM(n) := (p_n(k,m))_{\substack{1 \le k \le n \\ 1 \le m \le n}}$ of conditional probabilities:

```
#6:   TM(n):=VECTOR(VECTOR(p(n,k,m),m,n),k,n)
```

$$\#7:\quad TM(5) = \begin{bmatrix} \frac{1}{5} & \frac{4}{5} & 0 & 0 & 0 \\ 0 & \frac{4}{25} & \frac{3}{5} & \frac{6}{25} & 0 \\ 0 & 0 & \frac{27}{125} & \frac{74}{125} & \frac{24}{125} \\ 0 & 0 & 0 & \frac{256}{625} & \frac{369}{625} \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

We can read off the transition probabilities, for example: If three individuals in a population of five are infected, then with probability $\frac{24}{125}$ all five individuals shall be infected within the following period.

3. Question A

Let $q_n(N,k)$ be the probability that exactly k individuals are infected after n periods and let $Q_n(N) := (q_n(N,1),...,q_n(N,N))$ be the corresponding vector of probabilities. Then the following lemma settles **Question A**:

Lemma 4

- (1) $Q_{n+1}(N) = Q_n(N) \circ TM(N) = Q_1(N) \circ TM(N)^n$, and $Q_n(N) = Q_1(N) \circ TM(N)^{n-1}$
- (2) The average number of infected individuals after n periods is $m(N,n) =< Q_1(N) \circ TM(N), (1,2,...,N) >=< Q_1(N), TM(N) \circ (1,2,...,N) >$.

Here $< v,w >:= v \cdot w$ denotes the dot product for vectors of equal dimension.

Proof: (1) Obviously $q_{n+1}(N,k) = \sum_{\mu=1}^{k} q_n(N,\mu) \cdot p_N(\mu,k)$ is true because all possible states after n periods contribute to the $n+1$.vector of probabilities. Since $p_N(\mu,k) = 0$ for $\mu > k$ even $q_{n+1}(N,k) = \sum_{\mu=1}^{N} q_n(N,\mu) \cdot p_N(\mu,k)$ holds. This proves the left equality of (1). The second and third equality follow by iteration.

(2) The average number of infected individuals after n periods is $m(N,n) = \sum_{k=1}^{N} k \cdot q_n(N,k)$. Thus the first equality of (2) is proved. The second equality is an application of standard facts of matrix algebra.□

The corresponding *DERIVE* code reads:

```
#8:  infection_vector(n,m):=VECTOR(KRONECKER(1,k),k,n)·TM(n)^(m-1)
#9:  infection_vector(4,7)
#10: [0.000244140625,0.00439453125,0.0696037989109,0.925757529214]
```

Here $Q_m(n)$ equals `infection_vector(n, m)`. We can see, that the probability for three infections in a population of four is about 6.9% after seven time steps. The *DERIVE* equivalent for $m(N,n)$ is called `mean_infection_rate(N, n)` and defined as follows:

```
#11: mean_infection_rate(n,m):=infection_vector(n,m)·VECTOR(k,k,n)
#12: mean_infection_rate(4,7)=3.92087471671
```

This is the point to compare our result with Adrian Oldknow's result in [1], p.6. We first define a function which computes a least square fit of a logistic function f to a set of points. f is of the form $f(x) = \frac{n}{1 + a \cdot e^{-bx}}$, a,b real, positive, and n is the size of the population.

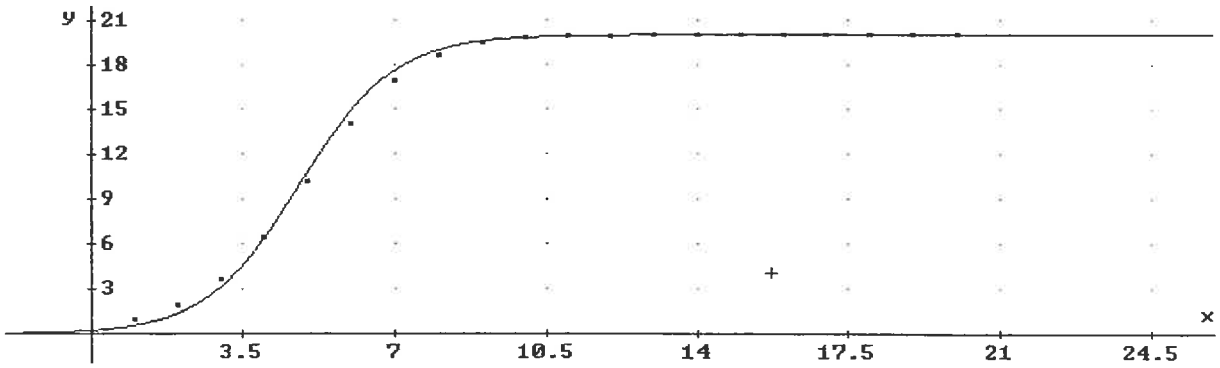
```
     logistic_fit(dat,n):=
       Prog
#13:     d:=VECTOR([dat↓k↓1,LN(n/dat↓k↓2-1)],k,DIM(dat))
         y_:=FIT([x,a·x+b],d)
         n/(1+EXP(y_))

#14: data_list(n,m):=VECTOR([k,mean_infection_rate(n,k)],k,m)
```

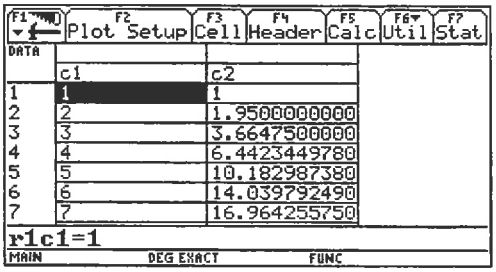
The following example computes the logistic fit for the first 20 average infection rates within a population of 20 individuals:

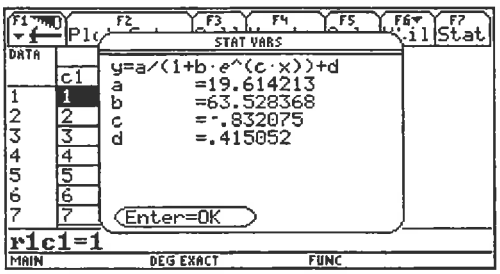
```
#15: logistic_fit(data_list(20,20),20)
```

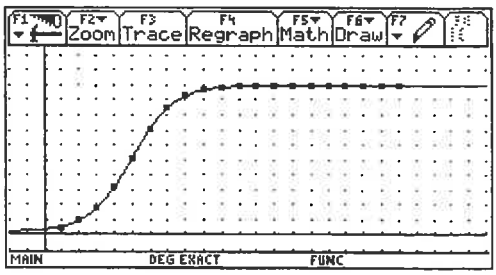
$$\#16: \quad \frac{2.95852 \cdot 10^{6} \cdot \hat{e}^{0.9204225209 \cdot x}}{1.47926 \cdot 10^{5} \cdot \hat{e}^{0.9204225209 \cdot x} + 1.2525363 \cdot 10^{7}}$$

The graph below demonstrates the quality of the `logistic_fit` .









Remarks: (a) Adrian's approach was quite different: He used the first three points to determine the logistic function completely, thus the first three values fit exactly. Our approach in contrast approximates all 20 points with the least square method, so at worst no point fits exactly. (b) Actually our `logistic_fit` is not exactly a least square fit to the logistic function but rather the exponential of a least square fit to the logarithm of the logistic function. This is due to the restrictions of the built-in *DERIVE* function `FIT`. The result differs slightly from the real least square fit. (c) We have at least some numerical evidence that Adrian's approach is not too far apart from reality.

3. Question B

The next lemma settles Question B:

Lemma 5 The expected number of time-steps until all N individuals are infected is:

$$E(N) = \sum_{n=2}^{\infty} n(q_n(N,N) - q_{n-1}(N,N))$$

Proof: The probability that all N individuals are at first infected after n periods is:
$X(n) = q_n(N,N) - q_{n-1}(N,N)$.
Thus the formula for $E(N)$ is proved by the definition of the expectation value of the random variable X.□

According with lemma 4, $q_n(N,N)$ is the last component of the vector
$Q_n(N) = Q_1(N) \circ TM(N)^{n-1}$ i.e.
$q_n(N,N) =< (1,0,...,0) \circ TM(N)^{n-1}, (0,...,0,1) >=< (1,0,...,0), TM(N)^{n-1} \circ (0,...,0,1) >$
The corresponding DERIVE code is:

```
#17: VECTOR(q(10,k),k,10)
#18: [0, 0, 0, 0, 0.02784654282, 0.2311090584, 0.5353264582,
      0.7688540444, 0.8977722780, 0.9576531977]
```

4. Effective computation of $m(N,n)$ and $E(N)$

The core of all computations related to the questions A and B is the computation of the matrix-powers $TM(N)^n$. A thorough analysis of the matrices $TM(N)$ leads to formulae in terms of powers of their eigenvalues.
Our first observation is: $TM(N)$ is an upper triangular matrix. As a consequence all diagonal elements are eigenvectors. If the eigenvalues are all different, then a basis of different eigenvectors exists which is equivalent to the statement that $TM(N)$ is equivalent to a diagonal matrix. But unfortunately this is not always the case as we learn from the next lemma:

Lemma 6

- (1) The eigenvalues $\lambda_i = p_N(i,i) = (\frac{i}{N})^i$ of $TM(N)$ are all different if and only if N is not an integer multiple of an integer power n^n, i.e.: $N \neq m \cdot n^n$, for all $m,n \in IN$.
- (2) If $N = m \cdot n^n$, then $TM(N)$ possesses for every number n with this property an eigenvalue of multiplicity 2.

We omit the proof of (1) and give only a hint how to prove a part of (2). The function $g(x) := (\frac{x}{N})^x$ is strictly convex in the interval $(0,N]$. g attains its minimum at $x = \frac{N}{e}$. Therefore g attains each value at most twice.

Examples:
The eigenvalues for $N = 4$ and $N = 8$ are: $\{\frac{1}{4}, \frac{1}{4}, \frac{27}{64}, 1\}$ and $\{\frac{27}{512}, \frac{1}{16}, \frac{1}{16}, \frac{3125}{32768}, \frac{1}{8}, \frac{729}{4096}, \frac{823543}{2097152}, 1\}$. If $N = 2^2 \cdot 3^3$, then $TM(108)$ possesses two eigenvalues of multiplicity 2.
What we really need to prove the following theorem is that the multiplicity of the eigenvalues is at most two, and that $TM(N)$ is upper triangular.

Theorem 2

Let $\lambda_1,\lambda_2,...,\lambda_K=1$, $K\le N$, be the distinct eigenvalues of $TM(N)$. Then there exist a vector $w=(\alpha_1,\alpha_2,...,\alpha_{K-1})$, a matrix $M=\begin{pmatrix}\beta_{11} & \beta_{12} & .... & \beta_{1,K-1} & 0\\ \beta_{21} & \beta_{22} & .... & \beta_{2,K-1} & 0\\ : & : & :::: & : & :\\ \beta_{N-1,1} & \beta_{N-1,2} & .... & \beta_{N-1,K-1} & 0\\ \beta_{N,1} & \beta_{N,2} & .... & \beta_{N,K-1} & 1\end{pmatrix}$ and a vector $v=(\delta_1,\delta_2...,\delta_{K-1})$ with rational coefficients and with the following properties:

1. $\delta_j=0$ if and only if the eigenvalue λ_j has multiplicity two
2. $m(N,n)=1+<w,((1+n\cdot\delta_1)\cdot\lambda_1^n,...,(1+n\cdot\delta_{K-1})\cdot\lambda_{K-1}^n)>=1+\sum_{i=1}^{K-1}\alpha_i\cdot(1+n\cdot\delta_i)\cdot\lambda_i^n$
3. $$Q_n(N)^T=\begin{pmatrix}q_n(N,1)\\ q_n(N,2)\\ :\\ :\\ q_n(N,N)\end{pmatrix}=M\circ\begin{pmatrix}(1+n\cdot\delta_1)\cdot\lambda_1^n\\ (1+n\cdot\delta_2)\cdot\lambda_2^n\\ :\\ (1+n\cdot\delta_{K-1})\cdot\lambda_{K-1}^n\\ 1\end{pmatrix}$$
4. $E(N)$ exists and is a rational number.

We do not prove this result here, we only mention the idea of the proof, which is an application of a decomposition of the triangular matrix $TM(N)$. Every triangular matrix is equivalent to a diagonal block matrix B , i.e.: $TM(N)=W^{-1}\circ B\circ W$ with a matrix B which has only simple eigenvalues or $n\times n$ triangular matrices D in its diagonal and zeroes elsewhere. Every such triangular matrix D corresponds to an eigenvalue with multiplicity $n\ge 2$ [29], p.238 . Together with the representations from lemma 4 and the properties of the scalar product we obtain the statements (2) and (3) of the previous theorem. $E(N)$ is a linear combination of infinite series in the eigenvalues λ_j with rational coefficients. These series are closely related to geometric series, so the limit is again rational.
We learn from Theorem 2 that the sequences $\{q_n(N,k)\}_{n\in IN}$ and $\{m(N,n)\}_{n\in IN}$ are actually linear combinations of geometric sequences and sequences of the form $\{n\cdot q^n\}_{n\in IN}$.

The following *DERIVE* functions compute $m(N,n)$, $q_n(N,N)$, and $E(N)$. These functions implement the results of theorem 2 in the following way: We first compute the quantities $m(N,N)$ and $q_n(N,N)$ with the help of matrix powers, and then we obtain the coefficients $\alpha_1,...,\alpha_{K-1}$,$\delta_1,...,\delta_{K-1}$, and $\beta_{N,1},...,\beta_{N,K-1}$ as solutions of linear systems of equations. The technical details need some efforts and are shortly explained.

The function `var_list(n)` creates a vector of n variables to be used later in the solutions function.

#19: codes_vector(n):=APPEND([97],REVERSE(VECTOR(48+MOD(FLOOR($\frac{n}{10^k}$),10),k,0,FLOOR(LOG(n,10)))))

#20: codes_vector(5)=[97,53]

#21: var_list(n):=VECTOR(CODES_TO_NAME(codes_vector(j)),j,n)

#22: var_list(5)=[a1,a2,a3,a4,a5]

Now comes the function eigenvalue_powers(n,m) that sorts the eigenvalues of TM(n) and creates a vector of powers λ^m of the first n-1 eigenvalues resp. $m*\lambda^m$ if λ is an eigenvalue of multiplicity 2.

```
#23: eigenvalue_powers(n,m):=
       Prog
         w:=SORT(VECTOR((k/n)^k,k,n))
         VECTOR(IF(w↓k=w↓(k+1),m·w↓k^m,w↓k^m),k,n-1)
```

simple_mean_infection_rate(n,m) computes the mean-infection-rate in a population of n individuals after m steps as a sum of m-th powers of eigenvalues of TM(n).

```
#24: simple_mean_infection_rate(n,m):=
       Prog
         m_:=VECTOR(eigenvalue_powers(n,k),k,n-1)
         w_:=var_list(n-1)
         b_:=VECTOR(n-mean_infection_rate(n,k),k,n-1)
         s_:=(SOLUTIONS(m_·w_-b_,w_))↓1
         n-s_·eigenvalue_powers(n,m)
```

#25: simple_mean_infection_rate(5,3) = $\frac{353}{125}$

The function complete_infection(n,m) computes the probability that all n individuals are infected after m steps.

```
#26: complete_infection(n, m) :=
       Prog
         m_ := VECTOR(eigenvalue_powers(n, k), k, n - 1)
         w_ := var_list(n - 1)
         b_ := VECTOR(1 - (infection_vector(n, k))↓n, k, n - 1)
         s_ := (SOLUTIONS(m_·w_ - b_, w_))↓1
         1 - s_·eigenvalue_powers(n, m)
```

#27: VECTOR(complete_infection(8, m), m, 10)

#28: [0, 0, 0, 0.00240325927734, 0.126797319135, 0.417254004696, 0.686987260878, 0.853429334731, 0.936683085928, 0.973860743461]

The function expected_number_of_steps(n) computes the average number of steps until all n individuals are infected.

```
#29: expected_number_of_steps(n) := Σ(m=2, ∞) m·(complete_infection(n, m) -
       complete_infection(n, m - 1))

#30: expected_number_of_steps(3) = 43/10

#31: VECTOR([k, expected_number_of_steps(k)], k, 2, 5)

#32: [ 2          3
       3        4.3
       4  5.1081081081
       5  5.74903608381 ]
```

There is still another way to compute the expected number of steps. The application of matrix algebra yields a very elegant result. We first define the ***reduced transition matrix*** by $RTM(n) := (p_n(k,m))_{1\le k\le n-1 \atop 1\le m\le n-1}$, i.e. we drop the last row and the last column of the ordinary transition matrix. Now we define the vectors $b_n := (p_n(1,n),...,p_n(n-1,n))$, which is the last column of $TM(n)$ with the exception of the last element, and $w_{n-1} := (1,0,...,0) \in IR^{n-1}$.
Now we can state the following theorem:

Theorem 3

$$E(N) = 1 + < w_{N-1} \circ (Id - RTM(N))^{-2}, b_N >$$

where <..,..> denotes the scalar or dot product again.

Proof: We have by lemma 5:

$$X(m) = q_m(N,N) - q_{m-1}(N,N) = \sum_{i=1}^{N} q_{m-1}(N,i) \cdot p_N(i,N) - q_{m-1}(N,N)$$

Because $p_N(N,N) = 1$, we conclude:

$$X(m) = \sum_{i=1}^{N-1} q_{m-1}(N,i) \cdot p_N(i,N) =< (q_{m-1}(N,1),...,q_{m-1}(N,N-1)), b_n >$$

By the same reasoning as in the proof of lemma 4 follows:
$(q_{m-1}(N,1),...,q_{m-1}(N,N-1)) = w_{N-1} \circ RTM(m)^{m-2}$ for $m \ge 2$. Thus we obtain:

$E(N) = \sum_{m=1}^{\infty} m \cdot X(m) =< w_{N-1} \sum_{m=2}^{\infty} m \cdot RTM(N)^{m-2}, b_{N-1} >$, because $X(1) = 0$. The following equality holds for all complex numbers with $|z| < 1$: $\sum_{m=2}^{\infty} m \cdot z^{m-2} = \frac{1}{(1-z)^2} + \sum_{m=0}^{\infty} z^m$.

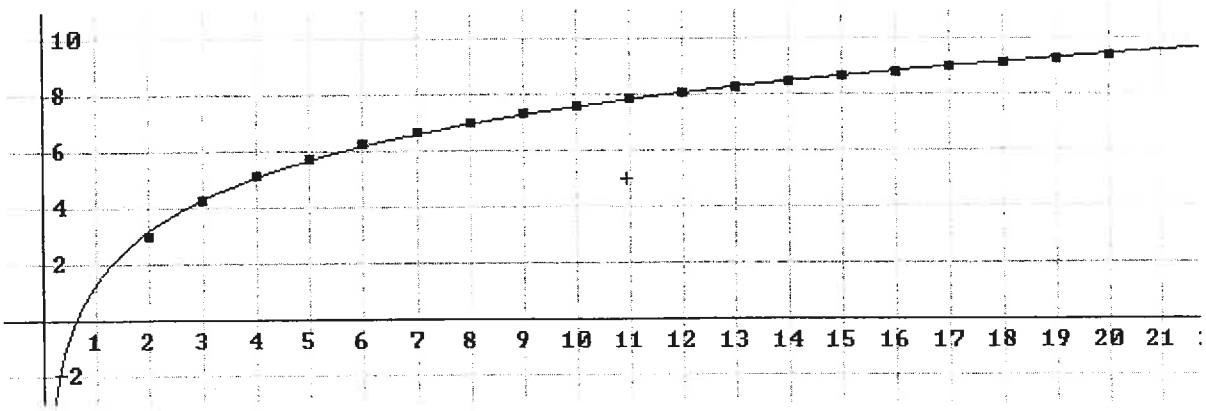
A standard result from functional analysis about power series of linear operators tells us that we can replace z by any linear operator/matrix A with norm $\|A\| < 1$. Since $RTM(N)$ is upper triangular and has only eigenvalues $\left(\frac{k}{N}\right)^k, 1 \le k \le N-1$, this condition is satisfied. Thus we have: $E(N) =< w_{N-1} \circ (Id - RTM(N))^{-2}, b_N > + \sum_{m=0}^{\infty} < w_{N-1} \circ RTM(N)^m, b_n >$

The second sum equals the sum of the probabilities $X(m)$ and is therefore equal to 1. Thus the theorem is proved.□

This last theorem admits the following *DERIVE* function E2(n):

```
#33: RTM(n) := VECTOR(VECTOR(p(n, k, m), m, n - 1), k, n - 1)

#34: b(n) := VECTOR(p(n, k, n), k, n - 1)

#35: w(n) := VECTOR(KRONECKER(1, k), k, n)

                                                              -2
#36: E2(n) := 1 + (w(n - 1)·(IDENTITY_MATRIX(n - 1) - RTM(n))  )·b(n)

#37: VECTOR(E2(k), k, 2, 20)

#38: [3, 4.3, 5.108108108, 5.749036083, 6.242469833, 6.663119581,
      7.020029536, 7.331712735, 7.609306956, 7.858803553, 8.085080268,
      8.292342220, 8.483635401, 8.661169662, 8.826706158, 8.981746767,
      9.127572122, 9.265244280, 9.395631480]

#39: VECTOR([k, E2(k)], k, 2, 20)
```

The last result has been computed in exact mode and approximated afterwards to save space. The following plot represents the data of #39 as points:



This graph coincides with Josef's graphs in [1], p.12. The line is the graph of a least square fit of a logarithmic function of type $g(x) = a + b \cdot \ln(x)$. The parameters a, b are computed as follows:

```
#40: VECTOR([k, E2(k)], k, 2, 20)
#41: log_fit(dat) := FIT([x, a + b_·LOG(x)], dat)
#42: log_fit(VECTOR([k, E2(k)], k, 2, 20))
#43: 2.724946932·LN(x) + 1.296810869
```

This result is reasonably different from Adrian's $s \approx 2 + 2.88 \ln N$. We use formula #43 to extrapolate our result for larger numbers and to compare this result with Josef's table in [1], p.12:

```
#44: VECTOR([x, 2.724946932·LN(x) + 1.296810869], x, [20, 30, 40, 50,
       100, 200, 1000])
```

```
#45:
[   20   9.460022336 ]
[   30   10.56489323 ]
[   40   11.34881162 ]
[   50   11.95686595 ]
[  100   13.84565523 ]
[  200   15.73444452 ]
[ 1000   20.12007742 ]
```

Note that this result is nothing but speculation. A more precise value for $N = 200$ is $E(200) \approx 15.1126$.

5. Conclusions

(1) We calculate the ratios $\frac{E(N)}{N}$ on the basis of #38:

```
#46: VECTOR(#38↓k/(1+k),k,DIM(#38))

#47: [1.5, 1.433333333, 1.277027027, 1.149807216, 1.040411638,
      0.9518742258, 0.8775036920, 0.8146347483, 0.7609306955, 0.7144366866,
      0.6737566890, 0.6378724784, 0.6059739572, 0.5774113108, 0.5516691348,
      0.5283380451, 0.5070873401, 0.4876444357, 0.4697815740]
```

The numerical evidence is that the sequence $\{E(N)/N\}_{N \in IN}$ is strictly decreasing. Thus the relative speed of infection seems to increase with the size of the population.

(2) Our theoretical results exhibit the existence and structure of a solution to the spread-of-infection problem but are of limited practical use. A computation of $E(1000)$ for example requires by our method the inversion of a 999×999 matrix.

(3) Whether this is a STAR or a SNOG approach is left to the reader.

References:

[1] J. Böhm, A. Oldknow: "Modelling the spread of infection", DNL #39, 2000
[2] M. Koecher: „Lineare Algebra und Geometrie", Springer-Verlag, 1985
[3] W. Vogel: „Wahrscheinlichkeitstheorie", Vandenhoeck & Ruprecht, 1970
[4] J. Wiesenbauer: "Titbits 19", DNL #41, p.20, 2001

Erratum:

Unfortunately there happened a typing error twice in the last DNL#41, page 14. In two comment lines it should read p3 instead of p2:

```
/* Points p1 and p3 determine the slope m_ of the regression line */

/* The y-intercept d1 of line(p1,p3) is given as */
```

Many thanks to Paul Drijvers who put my attentention on this mistake.

Drachenkurven – Dragon Curves

Richard Schorn, Kaufbeuren Richard.Schorn@t-online.de

Die Zeichenbefehle stammen wörtlich aus dem Artikel von Josef Lechner a.o., A Turtle for DfW5, DNL #38, Seiten 20-22.

Die beiden Funktionen T0 und DR0 sind das "Kondensat" aus vielen Plot-Programmen, die ich im Lauf der letzten 25 Jahre in verschiedenen Sprachen geschrieben habe (Assembler (WANG 600), BASIC, TurboPascal, CorelDraw-Script).

The Plot commands are from Josef Lecher a.o. "A Turtle for DfW5", DNL#38, p 20-22.

The functions T0, DR0, T1 and DR1 are the "condensate" from many plot programs which I have written during the last 24 years using various programming languages (Assembler (WANG 800), BASIC, Turbo Pascal, CorelDraw-Script).

```
#1:  [InputMode := Word, Angle := Degree, Precision := Approximate]

#2:  init := PROG(xcor_ := 0, ycor_ := 0, heading_ := 0, trace_ := [[0,0]])

#3:  new := PROG(xcor_ := 0, ycor_ := 0, heading_ := 0, trace_ := [[0,0]],
         RETURN trace_)

#4:  home := PROG(x_cor := 0, ycor_ := 0, heading_ := 0, trace_ := APPEND
         (trace_, [[xcor_, ycor_]]), RETURN trace_)

     fd(l) :=
       Prog
         xcor_ := xcor_ + l·COS(heading_)
#5:      ycor_ := ycor_ + l·SIN(heading_)
         trace_ := APPEND(trace_, [[xcor_, ycor_]])
         RETURN trace_

#6:  bk(l) := fd(-l)

     lt(a) :=
       Prog
#7:      heading_ := MOD(heading_ + a, 360)
         RETURN trace_

#8:  rt(a) := lt(-a)

#9:  Block commands := commands
                                  DIM(commands)
```

Before plotting one has to switch on the "Approximate Before Plotting"-Option in the Plot Window. The next functions are the "Dragon creators"!

You have to simplify `new =` *before each new plot to set the initial parameters.*

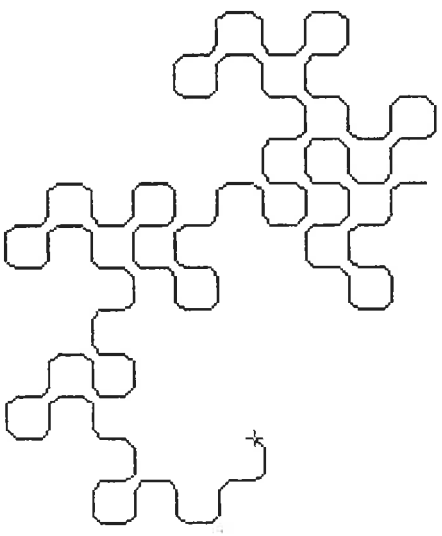
```
     T0(where) :=
        Prog
          what := where
          pot := 1
          dir := false
          Loop
            If MOD(what, 2) = 1 exit
            pot := 2·pot
            what := what/2
#10:        hh := (what - 1)/2
          If MOD(hh, 2) ≠ 0
             b1 := true
             b1 := false
          If pot ≤ where
             b2 := true
             b2 := false
          dir := b1 XOR b2
          RETURN dir
```

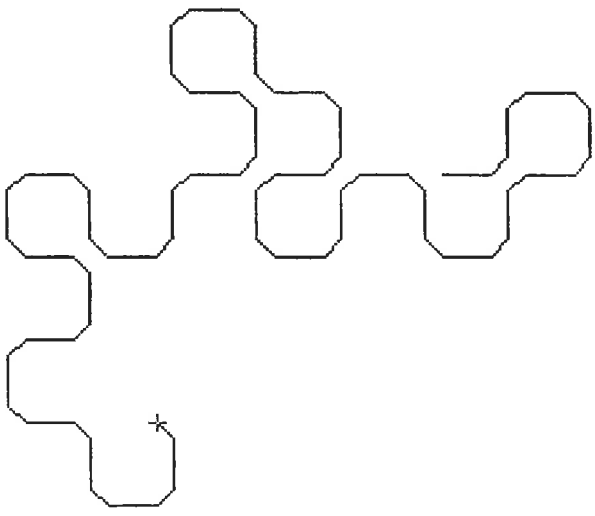
```
      DR0(st, sp) :=
        Prog
          where := st
          Loop
            dir := T0(where)
#11:        If dir
               Block(rt(45), fd(0.5), rt(45), fd(1))
               Block(lt(45), fd(0.5), lt(45), fd(1))
            where := where + 1
            If where = sp + 1
               RETURN trace_
```

```
#12: new = [[0, 0]]
```

```
#13: DR0(1, 100)
```

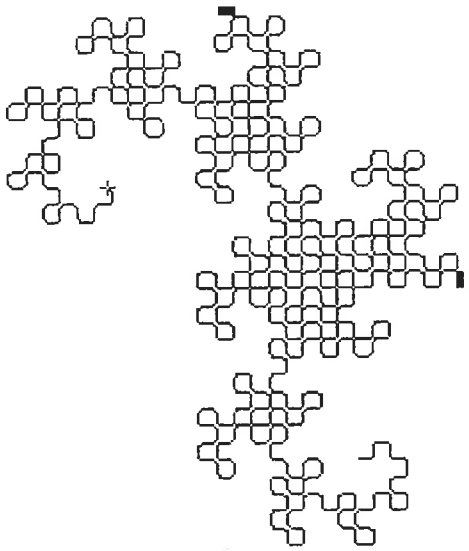
```
#14: new = [[0, 0]]

#15: DR0(123456, 123489)
```





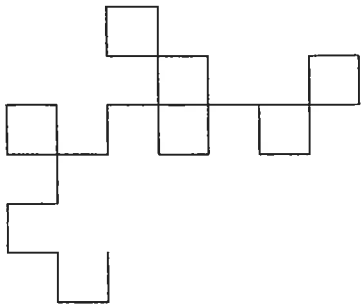
```
#16: new = [[0, 0]]
```

```
#17: DR0(1, 512)
```



```
DR00(st, sp) :=
  Prog
    init
    where := st
    Loop
      dir := T0(where)
#18:  If dir
        Block(rt(90), fd(1))
        Block(lt(90), fd(1))
      where := where + 1
      If where = sp + 1
        RETURN trace_
```

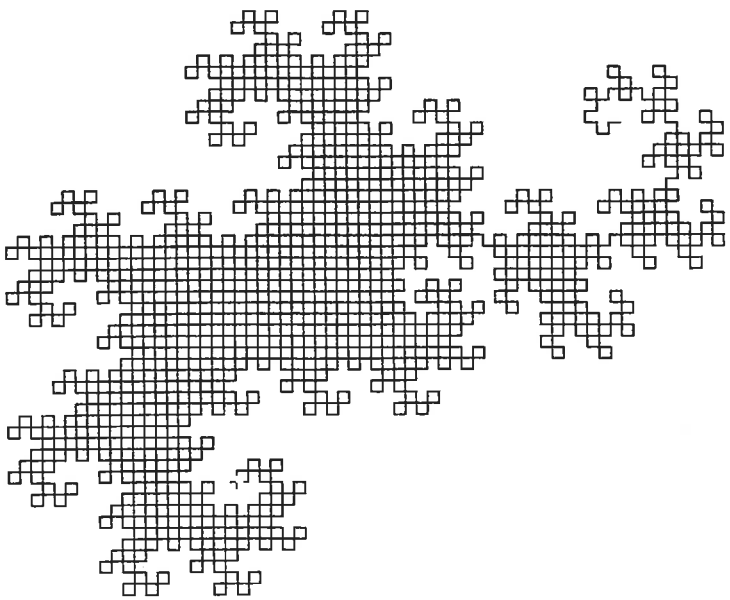
```
#19: DR00(123456, 123489)
```



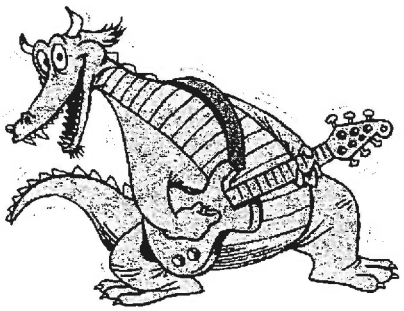
`DR00(st,sp)` *omits the 45° bends and the graph loses a lot of its beauty, but we win calculation time and so we are able to plot "curves" with a larger value for* `sp` *in a reasonable time.* `init` *makes sure that we always start plotting in (0\|0) without the need of simplifying* `new =` *before producing new plot.*

We are ready to wait for the first Dragon to appear:

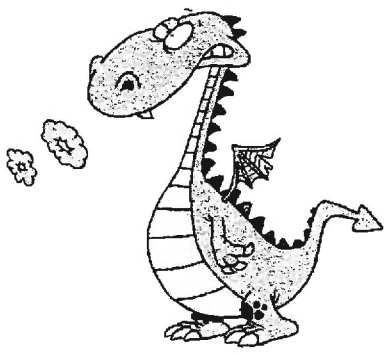
```
#20: DR00(1, 2048)
```



Generalized Dragon Curves

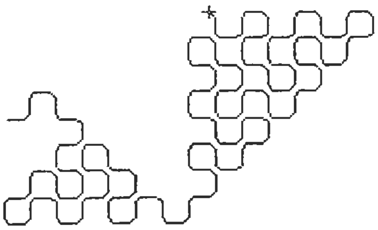


```
      T1(where) :=
        Prog
          what := where
          ww := 0
          pot := 1
          dir := false
          Loop
            If MOD(what, 2) = 1 exit
            pot := 2·pot
            what := what/2
            ww := ww + 1
#21:        hh := (what - 1)/2
          If MOD(hh, 2) ≠ 0
             b1 := true
             b1 := false
          If pot ≤ where
             b2 := true
             b2 := false
          If which↓(ww + 1) = 0
             b3 := true
             b3 := false
          dir := b1 XOR b2 ∧ b3
          RETURN dir

      DR1(st, sp) :=
        Prog
          where := st
          Loop
            dir := T1(where)
#22:        If dir
               Block(rt(45), fd(0.5), rt(45), fd(1))
               Block(lt(45), fd(0.5), lt(45), fd(1))
            where := where + 1
            If where = sp + 1
               RETURN trace_

      DR11(st, sp) :=
        Prog
          init
          where := st
          Loop
            dir := T1(where)
#23:        If dir
               Block(rt(90), fd(1))
               Block(lt(90), fd(1))
            where := where + 1
            If where = sp + 1
               RETURN trace_
```

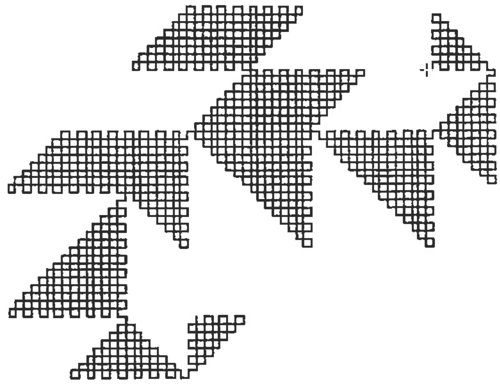


```
#24: which := [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

#25: DR1(1, 100)



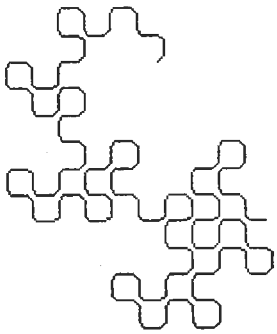
#26: DR11(1, 2048)



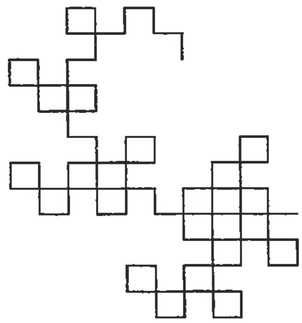
#27: which:=[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]

#28: new = [[0, 0]]

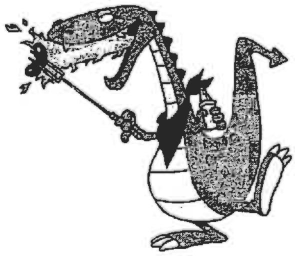
#29: DR1(1, 100)

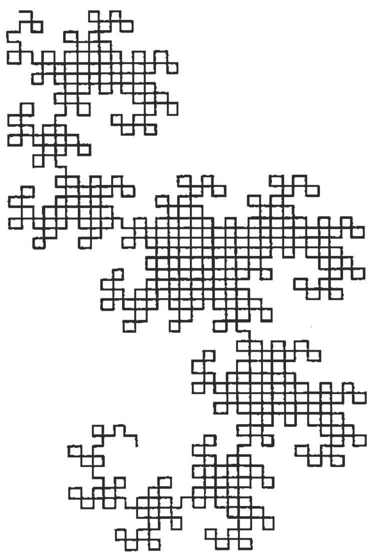


#30: DR11(1, 100)



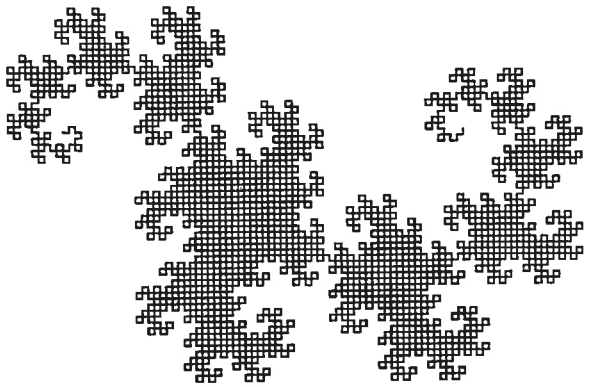
#30: DR11(1, 34^2)



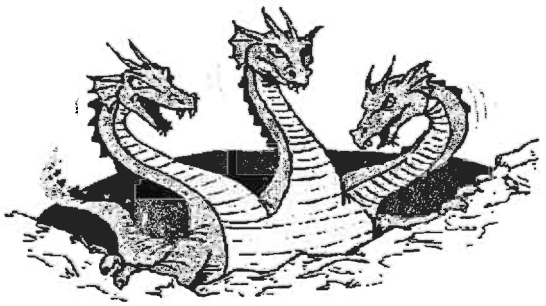


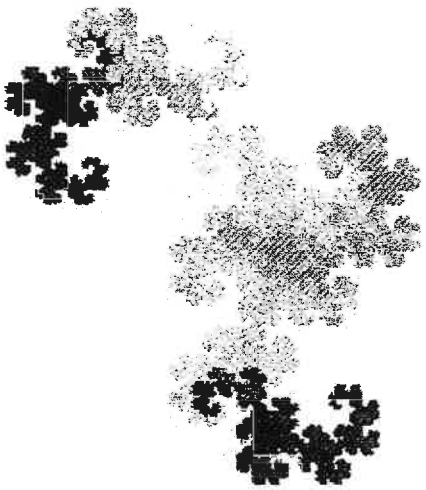
Let´s close with two "Super Dragons" setting up the connection to Christine Smith´s question for producing the "Jurassic Park Fractal".

DR00(1, 4096)



DR00(1, 8192)





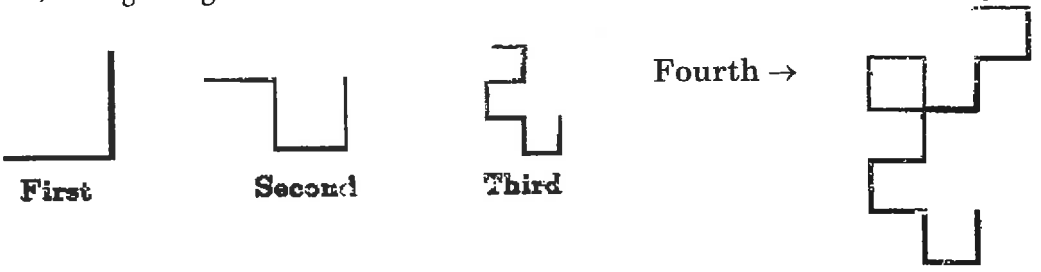
From: Christine White, christine@ds20.freeserve.co.uk
Subject: Help - Jurassic Park - Fractal

Jurassic Park Fractal after 50 foldings

I had hoped to find a newsgroup on which to post this plea, unfortunately my ISP doesn't list anything to do with DERIVE, perhaps you can help.

I recently did some work with Yr 7 & 8 (Age approx 12 -13) on the Jurassic Park Fractal as described on Cynthia Lanius' excellent web site (http://math.rice.edu/~lanius/frac/jurra.html). I remembered using DERIVE to create various Fractals at university and looked up my work on this. Unfortunately the Jurassic Park Fractal continues as an addition at the end of the previous iteration, whereas the examples I had programmed are all based on the original line. Is their any way I can program DERIVE to do this? Perhaps you could ask your members for me.

Take the strip in both hands, fold the paper over end-to-end, right hand onto left, and -- crease. Now fold it again in the same way, right onto left, and crease, and again, right . onto left and crease, and again right onto left and crease. You have folded it four times in all, right?



The title of the next contribution is RSA_Demo.DFW. It is really an original .dfw-file converted into the RTF-format. This is a very useful feature of the latest version of DERIVE 5.04. Josef

RSA_Demo.DFW

by Richard Schorn, Kaufbeuren, Richard.Schorn@t-online.de

Das Programm ist einem von mir in UBASIC geschriebenen Programm nachempfunden (RSADEMO.UB, Juni 1994)
Es folgt der Darstellung in Paulo Ribenboim , The little book of big primes, ab Seite 102 zum Thema: Public Key Kryptographie nach R.L.Rivest, A.Shamir und L.M.Adleman, 1978. Kurz RSA-Verschlüsselung.

Auf die Theorie des Verfahrens wird hier n i c h t eingegangen!

This program has its base in a UBASIC program (RSADEMO.UB), written by me in June 1994.
It follows the presentation of the algorithm given by Paulo Ribenboim, The little book of big primes, pp 102, dealing with Public Key Cryptography from R.L.Rivest, A.Shamir and L.M.Adleman, 1978.

We will n o t explain here the theory of the process!

Ziel der RSA-Verschlüsselung ist sicherer Code für die Übermittlung von vertraulichen Informationen, wobei ein "öffentlicher Schlüssel" verwendet wird. Dieser kann z.B. in einer Datenbank abgelegt sein, in die jedermann Einsicht hat.

Ein solcher Schlüssel besteht aus zwei Zahlen N und S:

N ist das Produkt von zwei (möglichst) großen Primzahlen **P** und **Q**. Die beiden Faktoren P und Q werden **nicht** bekanntgemacht.
Hier werden Primzahlen verwendet, die sich aus Fakultäten durch Addition oder Subtraktion von 1 ergeben.

S ist eine beliebige Zahl, die aber zu $\Phi = (P - 1)(Q - 1)$ teilerfremd sein muß.
Hier wird die Größe S durch ein kleines Programm `FIND_S(z)` ermittelt:

> Man startet mit Q - P und erhöht diese Größe um eins, solange bis das Ergebnis zu Φ teilerfremd ist.

The aim of RSA-Encryption is to find a safe code for transfering confidential information using a "Public Key". This key can be published by person A and can be used by everybody to encrypt a message for person A. The encrypted message can be sent open.

The key consists of two numbers N and S:

***N** is the product of two (as large as possible) prime numbers P and Q, which person A has to keep secret. In our procedure we use prime numbers created from faculties by adding or subtracting 1.*

***S** is an arbitrary number, which must be prime to $\Phi = (P - 1)(Q - 1)$. Here we find S by using a small program `FIND_S(z)`: we start with Q - P and increase it by 1 until we reach a number prime to Φ.*

```
#1:   P := 30! - 1

#2:                    265252859812191058636308479999999

#3:   Q := 37! + 1

#4:              13763753091226345046315979581580902400000001

#5:   N := P·Q

#6:   365087486919667303362768951061391574869431163902533269938382905540607
        79999999

#7:   Φ := (P - 1)·(Q - 1)

#8:   365087486919667303362768951061390198494122014742742657121318883819527
        00000000

      FIND_S(z) :=
        Prog
          s := Q - P
#9:       Loop
            If GCD(s, Φ) = 1
              RETURN s
            s := s + 1

#10:  S := FIND_S(z)

#11:             13763753090961092186503788522944593920000031
```

Für die Entschlüsselung muss der Anwender einmal die Größe **T** ermitteln. Sie muss der Bedingung S*T = 1 mod Φ genügen. Natürlich bleibt auch diese Zahl geheim.

*For deciphering person A needs to find number **T**, which satisfies S * T = 1 mod Φ. A must keep this number secret.*

```
#12:  T := INVERSE_MOD(S, Φ)

#13:  223656850602829266985300397930103063446191137427622418003962986808887
        15483871
```

Die zu übermittelnde Nachricht darf hier nur 25 Zeichen umfassen, daher n :=25. Fehlende Zeichen werden durch blanks (032) ersetzt, überzählige Zeichen werden abgeschnitten.

In our DEMO the message must have not more than 25 characters. Missing characters are replaced by blanks (032), surplus numbers are cut off.

```
#14:  n := 25

#15:  Ms := "Josef und DNL sind OK!"
```

Soll nun einem Benutzer diese geheime Nachricht übermittelt werden, so wird diese zunächst in eine Zahl (<N) umgesetzt. Diese Umsetzung ist auch allgemein bekannt. In diesem Programm wird die Nachricht mit Hilfe des ASCII-Codes erzeugt.

Dazu dient die Funktion `TEXT_IN_ZAHL(t)`.

Durch die Umsetzung ergibt sich so eine 74- oder 75-stellige Zahl. Im Beispiel entsteht die 74-stellige Zahl M =

= 74111115101102032117110100032068078076032115105110100032079075033032032

If anybody wants to send a message to person A, he/she has to convert it into a number < N. This conversion is public and in our program we will use the ASCII-Code.

Function `TEXT_IN_ZAHL(t)` *converts the plain message into a number of 74 or 75 digits. In our example we receive the number M given above.*

```
      TEXT_IN_ZAHL(t) :=
        Prog
          t1 := NAME_TO_CODES(APPEND(t, "                              "))
          s := 0
          i := 1
#16:      Loop
            If i = n
               RETURN s
            s := 1000·s + t1↓i
            i := i + 1

#17:  M := TEXT_IN_ZAHL(Ms)

#18:  74111115101102032117110100032068078076032115105110100032079075033032~
        032
```

Diese Zahl wird mit S potenziert (!), es wird aber modulo N gerechnet. Man erhält die Zahl M_, die dann verschickt wird. Jeder kann die Zahl lesen, aber nur der "Besitzer" der Zahl T kann den Sinn entschlüsseln.

We - the computer - calculates M ^ S modulo N giving the encrypted message M_, which will be sent. Everybody can read the message, but only person A - the "Owner" of number T is able to decrypt it.

```
#19:  M_ := POWER_MOD(M, S, N)

#20:  68111729600920181493916434553533768397128666088635438648461903581728~
        6673124
```

Der rechtmäßige Empfänger potenziert die entstandene Zahl M_ mit T und erhält die Zahl Mo. Natürlich wird wieder modulo N gerechnet.

Wegen Mo = (M^S)^T = M^(S*T) = M ergibt sich die ursprüngliche Zahl M!

The receiver calculates M_ ^ T modulo N giving the result Mo, which is identical to M.

```
#21:  Mo := POWER_MOD(M_, T, N)

#22:  74111115101102032117110100032068078076032115105110100032079075033032~
        032
```

Das Ergebnis wird dann mittels der Funktion `ZAHL_IN_TEXT(z)` in die ursprüngliche Nachricht umgesetzt.

Function `ZAHL_IN_TEXT(z)` *converts number Mo = M into the original message.*

```
      ZAHL_IN_TEXT(z) :=
        Prog
          m := []
          i := 1
          Loop
#23:        If i = n
               RETURN CODES_TO_NAME(m)
            h := MOD(z, 1000)
            z := (z - h)/1000
            m := APPEND([h], m)
            i := i + 1

#24:  ZAHL_IN_TEXT(Mo)

#25:                ZAHL_IN_TEXT(Mo) = "Josef und DNL sind OK!"
```

Assume, that you are person A and I send a message using Richard´s public key (N,S):

119225606910150077078077912792935370400758686521607055826063344121678516804 1

What is my message?

Additional references (also for the next contribution):

[1] Johann Wiesenbauer, Public Key Kryptosysteme in Theorie und Programmierung in Didaktikhefte der ÖMG, Heft 30, Oktober 1999; j.wiesenbauer@tuwien.ac.at

[2] Johann Wiesenbauer, Using *DERIVE* to Explore the Mathematics Behind the RSA Cryptosystem, Workshop at the 4th International DERIVE & TI-92 Conference, Liverpool

[3] Dirk Warthmann, Cryptology with *DERIVE* in the Classroom, lecture at the 4th International DERIVE & TI-92 Conference, Liverpool; d.warthmann@gmx.de

The Merkle - Hellmann - Algorithm,
another Public Key Cryptosystem

Josef Böhm, Würmla, Austria

1. Function `convert(text)` converts a string into a sequence of 14-digit binary numbers = binary word. `convert` applies the built in `NAME_TO_CODES(text)` function together with OutputBase := Binary to directly convert the pairs of characters via their ASCII-Code to binary numbers. ASCII-Codes between 32 and 64 must be accomplished by a leading 0.
   Characters with an ASCII-Code > 127 are not allowed (special characters, which are usually not used in messages). In case of a message with an odd number of characters we add a blank (ASCII-Code 32 = `"0100000"`).

```
User:
      convert(text, code_) :=
         Prog
           OutputBase := Binary
           code_:=NAME_TO_CODES(text)
#1:        code_:=VECTOR(IF(DIM(c_)<7,APPEND("0",STRING(c_)),STRING(c_)),c_,code_)
           code_:=IF(EVEN?(DIM(code_)), code_, APPEND(code_, ["0100000"]))
           OutputBase := Decimal
           code_:=VECTOR(APPEND(code_↓i, code_↓(i + 1)), i, 1, DIM(code_), 2)

User=Simp(User):
#2:   convert(Josef) = [10010101101111, 11100111100101, 11001100100000]
```

```
User=Simp(User):
#2:   convert(Josef) = [10010101101111, 11100111100101, 11001100100000]
```

2. `cv(m)` generates a "structured" coding vector of m components. Each component is greater than the sum of the preceding components.

```
User:
      cv(m, cv_ := [], n_ := 0) :=
        Prog
          x_ := RANDOM(100000) + 1
          Loop
            If n_ = m
#3:             RETURN cv_
            c_ := APPEND(cv_, [x_])
            x_ := Σ(c_) + RANDOM(10000) + 1
            cv_ := c_
            n_ :+ 1

User:
#4:   cv(14)

Simp(#4):
#5:   [41404,45054,95677,182224,371926,740745,1481862,2962824,5929138,
       11854718,23710896,47417068,94842866,189677298]

User:
#6:   code1 := [41404,45054,95677,182224,371926,740745,1481862,2962824,
       5929138,11854718,23710896,47417068,94842866,189677298]
```

3. `encode(text,code)` encodes the plain message by multiplying each binary vector of #2 by `code1`, giving three large numbers:

```
User:
      yes(ch) :=
        If (NAME_TO_CODES(ch))↓1 = 49
#7:        1
           0

User:
#8:   encode(text, code) := VECTOR( Σ_{j=1}^{14} code_j ·yes(v_j), v_, convert(text) )

User:
#9:   encode(Josef, code1)

Simp(#9):
#10:  [365504463, 248391070, 7128267]

User:
#11:  [41404,45054,95677,182224,371926,740745,1481862,2962824,5929138,
       11854718,23710896,47417068,94842866,189677298]
```

Decoding of this message is very easy because of the special structure of code1 (repeated above):

189677298 ≤ **365504463** --> c[14] = **1**; 365504463 - 189677298 = 175827165
94842866 ≤ 175827165 --> c[13] = **1**; 175827165 - 94842866 = 80984299
47417068 ≤ 80984299 --> c[12] = **1**; 80984299 - 47417068 = 33567231
23710896 ≤ 33567231 --> c[11] = **1**; 33567231 - 23710896 = 9856335
11854718 > 9856335 --> c[10] = **0**
5929138 ≤ 9856335 --> c[9] = **1**; 9856335 - 5929138 = 3927197
2962824 ≤ 3927197 --> c[8] = **1**; 3927197- 2962824 = 964373
1481862 > 964373 --> c[7] = **0**
740745 ≤ 964373 --> c[6] = **1**; 964373 - 740745 = 223628
371926 > 223628 --> c[5] = **0**
182224 ≤ 223628 --> c[4] = **1**; 223628 - 182224 = 41404
95677 > 41404 --> c[3] = **0**
45054 > 41404 --> c[2] = **0**
41404 ≥ 41404 --> c[1] = **1**; 41404 - 41404 = 0.

Compare the bold printed bits with the first binary word of our message!

So we have to "hide" the structured vector `code1` using modulo arithmetic:

4. We need two large numbers m and w with GCD(m,w) = 1 for encoding the structured coding vector (e.g. code 1).
   `wm` = returns a pair of numbers `[w,m]` which must be kept secret.

```
User:
      find_wm(w, m) :=
        Prog
          m := (RANDOM(10) + 30)! + RANDOM(10^35)
          w := (RANDOM(10) + 25)! + RANDOM(10^35)
#12:      Loop
            If GCD(m, w) = 1
               RETURN [w, m]
            w :+ 1

User:
#13:  wm := find_wm(w, m)

User=Simp(User):
#14:   wm = [6392998172132230262902958871196 1114, 2932131215201925332249774
              5214189529]

User:
#15:  pair1 := [6392998172132230262902958871196 1114,
       2932131215201925332249774 5214189529]
```

5. `pubcode(str_c,c_pair)` returns the public coding vector which has been generated using the structured vector `str_c` together with the coding pair `c_pair`. Each component of `str_c` is multiplied by w modulo m. Doing this `str_c` loses its structure and cannot be used to decode the message as demonstrated above.

```
User:
      pub_code(str_c, c_pair) := MOD(c_pair  ·str_c, c_pair )
#16:                                        1              2

User:
#17:  pub_code1 := pub_code(code1, pair1)
```

6. Publish the following code to encode messages for you. The encoding process is the same as before using `code1`. The encoded message can be sent open, because only you are able to decipher it.

```
User=Simp(User):
#18:  pub_code1 = [48299782425436171796395642924231l0,
       10261155299730272700581948430217428,272183668255900446993210430826l7604,
       14422003921794470676115634402839133, 13252681217842577909759939490618413,
       18620672065712894024313831837925074, 10289077023688668179465550603487008,
       12009047119079988972877370681827314, 44688247864783690511782661256267 98,
       20802360473571259952587266775975966, 20320142048401779455553291035718036,
       15600486216535907779607467705092707, 87645858902867732429442387330261l9,
       21672497630322520661374024517265543]
```

Now we encode two test messages using pub_code1:

```
User:
#19:  encmess1 := encode("Josef", pub_code1)

User:
#20:  topsecret := encode("This is my second attempt with Merkle-Hellmann!"
          , pub_code1)

User=Simp(User):
#21:  encmess1 = [120708237921156321043603689331743834,
       12497010520968228324835906727648258 6, 514333116123077308654735501768108 23]
```

7. For decoding the first two characters of `encmess1` one has to find out in which way the number

120708237921156321043603689331743834

can be composed as the sum of `pub_code1`'s components. (2^14 possibilities). Take a 1000 digit binary word and then one has to find the one right composition among 2^1000. This number exceeds the number of atoms in our universe.

We have to reverse the hiding process by finding the inverse to w mod m. `invw(c_pair)` returns a large - to keep secret - number which after a multiplication modulo m unhides the message.

```
User:
      invw(c_pair) := INVERSE_MOD(c_pair , c_pair )
#22:                                     1        2

User:
#23:  invw1 := invw(pair1)

Simp(#23):
#24:                          212226318883168769894659474632080 60

User=Simp(User):
                                MOD(invw1·pair1 , pair1 ) = 1
#25:                                        1       2

User:
      MOD(encmess1·invw1, pair1 )
#26:                        2

Simp(#26):
#27:                          [365504463, 248391070, 7128267]
```

Compare with expression #10!

8. decode(mess, code, c_pair) returns the original 14-digit binary words (process of page 30).

```
User:
      decode(mess, code, c_pair, n_ := 1, m1_, pl_, plain_ := []) :=
        Prog
          mess := MOD(mess·INVERSE_MOD(c_pair↓1, c_pair↓2), c_pair↓2)
          Loop
            If n_ > DIM(mess)
               RETURN plain_
            pl_ := ""
            j := 14
            m1_ := mess↓n_
            Loop
#28:          If j = 0 exit
              If m1_ ≥ code↓j
                 Prog
                   pl_ := ADJOIN("1", pl_)
                   m1_ := m1_ - code↓j
                 Prog
                   pl_ := ADJOIN("0", pl_)
              j := 1
            plain_ := APPEND(plain_, [pl_])
            n_ :+ 1

User:
#29:  decode(encmess1, code1, pair1)

Simp(User):
#30:  [10010101101111, 11100111100101, 11001100100000]→Split it up, improve decode

User:
      decode(mess, code, c_pair, n_ := 1, m1_, pl_, plain_ := []) :=
        Prog
          mess := MOD(mess·INVERSE_MOD(c_pair↓1, c_pair↓2), c_pair↓2)
          Loop
            If n_ > DIM(mess)
               RETURN plain_
            pl_ := ""
            j := 14
            m1_ := mess↓n_
            Loop
#31:          If j = 0 exit
              If m1_ ≥ code↓j
                 Prog
                   pl_ := ADJOIN("1", pl_)
                   m1_ := m1_ - code↓j
                 Prog
                   pl_ := ADJOIN("0", pl_)
              j := 1
            plain_:=APPEND(plain_,APPEND([pl_↓[1,...,7]],[pl_↓[8,...,14]]))
            n_ :+ 1

User:
#32:  decode(encmess1, code1, pair1)

Simp(#29):
#33:  [1001010, 1101111, 1110011, 1100101, 1100110, 0100000]
```

9. At last we need an auxiliary function to convert the binary words into the respective ASCII-Codes and further into the original characters. Notice that I change from APPEND to ADJOIN!!

```
User:
#34:  bintodec(binword) := CODES_TO_NAME( Σ_{i=1}^{7} IF(binword_i = "1", 2^(7-i), 0,0) )

User=Simp(User):
#35:  bintodec("1001010") = "J"

User:
      decode(mess, code, c_pair, n_ := 1, ml_, pl_, plain_ := []) :=
        Prog
          mess := MOD(mess·INVERSE_MOD(c_pair↓1, c_pair↓2), c_pair↓2)
          Loop
            If n_ > DIM(mess)
               RETURN plain_
            pl_ := ""
            j := 14
            ml_ := mess↓n_
            Loop
#36:          If j = 0 exit
              If ml_ ≥ code↓j
                 Prog
                   pl_ := ADJOIN("1", pl_)
                   ml_ := ml_ - code↓j
                 Prog
                   pl_ := ADJOIN("0", pl_)
              j :- 1
            plain_ := ADJOIN(plain_, ADJOIN(bintodec(pl_↓[1, ..., 7]),
                             bintodec(pl_↓[8, ..., 14])))
            n_ :+ 1

User=Simp(User):
#37:  decode(encmess1, code1, pair1) = Josef
```

We can unveil our "topsecret" message:

```
User:
#38:  decode(topsecret, code1, pair1)

Simp(#38):
#39:  "This is my second attempt with Merkle-Hellmann!"
```

Last comment: Problems occur when coding numbers - even under quotes - which are following a blank:

```
#40:  test1 := encode("1", pub_code1)

#41:  test01 := encode(" 1", pub_code1)

#42:  test101 := encode("1 1", pub_code1)

#43:  test1001 := encode("1  1", pub_code1)

#44:                          decode(test1, code1, pair1) = "1"

#45:                      decode(test01, code1, pair1) = ADJOIN("", ?)

#46:                         decode(test101, code1, pair1) = "1 1 "

#47:                    decode(test1001, code1, pair1) = ADJOIN("1 ", ?)
```

But doing the same with letters, we meet no problems.

```
#48:  testa := encode(" a", pub_code1)

#49:                          decode(testa, code1, pair1) = " a"

#50:  testa00a := encode("a  a", pub_code1)

#51:                         decode(testa00a, code1, pair1) = "a  a"
```

Some closing comments:

Although this encryption is said to be not secure it could be useful to demonstrate an application of number theory. I think that in this way of presentation the process and its programming as well can be followed by the students. I don´t want to compete with Johann Wiesenbauer´s art of programming producing a very tight code using many wonderful mathematical short cuts.

I produced again a RTF - file - including the annotations. There is one thing to be repaired by the "DERIVE Makers". The quotes - marking a string - do not appear. So I had to add the quotes.

between

Volker Loose, Germany, volker@VWLOOSE.DE

Hello DERIVE-users,

in DNL#37 I found the function shadow created by Josef Lechner, which marks the area between horizontal axis and plot of a function.

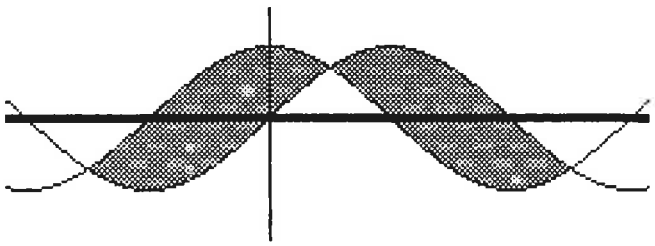
With a slight modification it is possible to mark the area between the plots of two functions. I attach my solution.

```
#1:  between(f, g, int) := int_1 ≤ x ≤ int_2 ∧ SUBST(IF(f_ ≥ g_, g_ ≤ y ≤
       f_, f_ ≤ y ≤ g_), [f_, g_], [f, g])
```

See some examples:

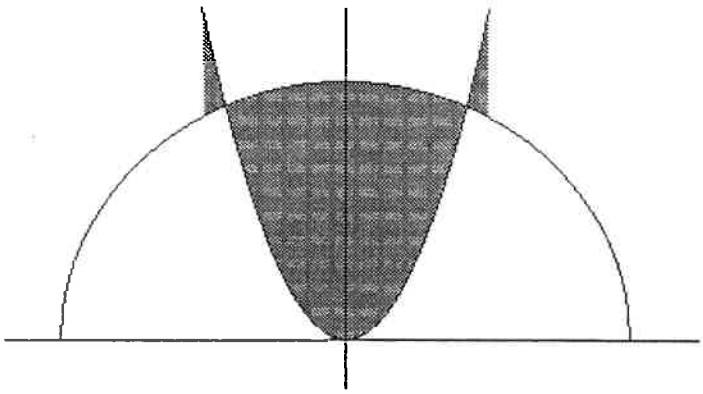
```
#2:  [y = SIN(x), y = COS(x)]
```

$$\#4:\quad \text{between}\left(\text{SIN}(x),\ \text{COS}(x),\ \left[-\frac{3\cdot\pi}{4},\ \frac{5\cdot\pi}{4}\right]\right)$$



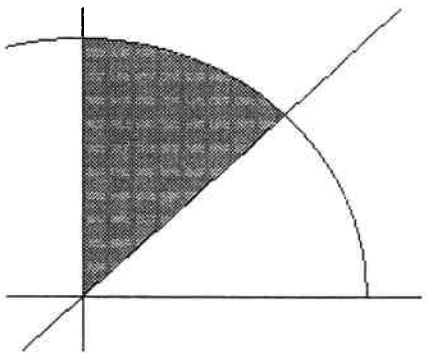
#5: $[y = \sqrt{(25 - x^2)},\ y = x^2]$

#7: $\text{between}(\sqrt{(25 - x^2)},\ x^2,\ [-2.5,\ 2.5])$



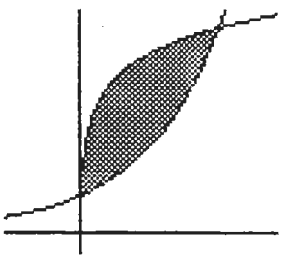
#9: $y = x$

#10: $\text{between}\left(\sqrt{(25 - x^2)},\ x,\ \left[0,\ \frac{5 \cdot \sqrt{2}}{2}\right]\right)$



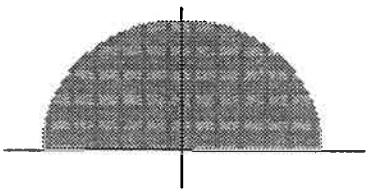
#12: $[y = \text{EXP}(x),\ y = \text{LN}(x) + 5]$

#14: $\text{between}(\text{EXP}(x),\ \text{LN}(x) + 5,\ [0,\ 1.74])$

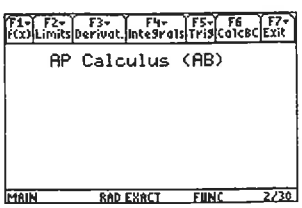


For g = 0 between does the same as Josef Lechner´s shadow.

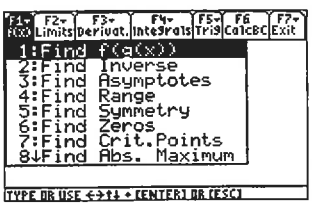
#17: $\text{between}(\sqrt{(25 - x^2)},\ 0,\ [-5,\ 5])$



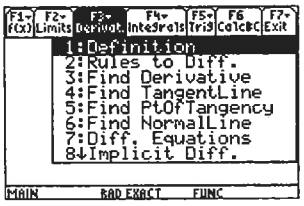
Recently I received a mail from Nils Hahnfeld, a German teacher working on the US Virgin Islands. He wrote about a Calculus program package, which he has developed for his students. He sent his package for a try and I was very much impressed by his results. I´d like to invite you to visit his home page.



AP Calculus AB (Start)

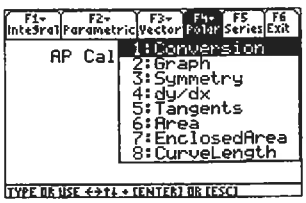


PreCalculus

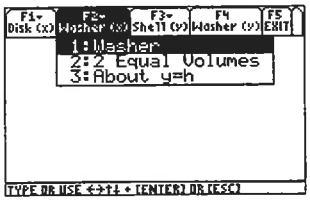


Derivatives

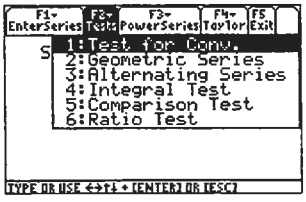
Integrals



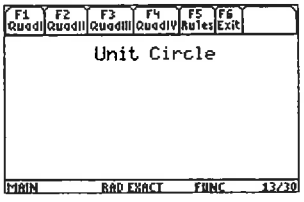
AP Calculus BC



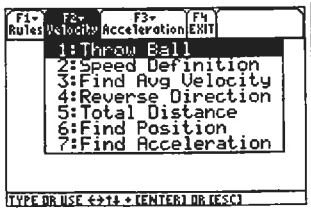
Volume



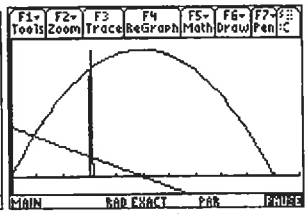
Sequencies & Series



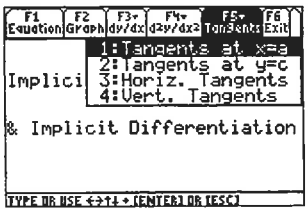
Unit Circle



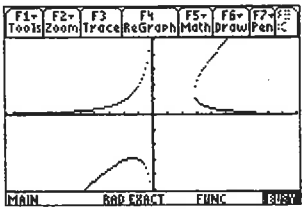
1 dim. Motion



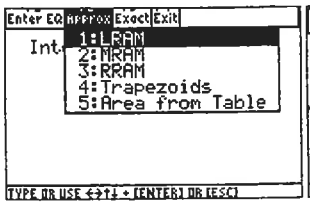
Motion Animation



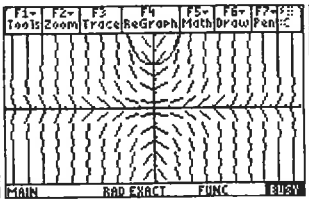
Implicit Differentiation



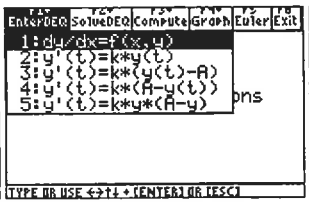
Implicit Plots

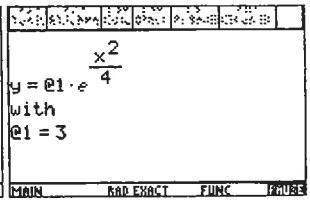


Area Approx

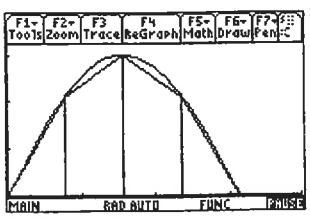


Slope Fields & Particular Solution

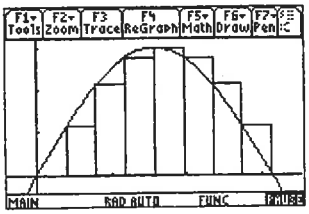




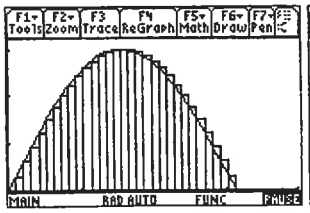
Differential Equations



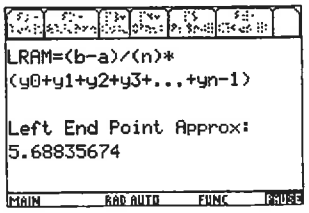
Trapezoid Areas



LRAM



LRAM



LRAM

The TI-89 program can easily be adapted for the TI-92 PLUS. One has only to change some lines of text to be displayed on the screen, because they are too long for the TI-92 screen. This causes a dimension error message. Maybe that Nils will offer a TI-92 version on demand. Nils is German, so it will be no problem - from my point of view - to receive a German program version.

Nils´ home page is: www.nhahnfeld.com or www.islands.vi/~shawn/ti-89.htm.
His email-address: nhahnfeld@antilles.k12.vi.us

End Exam 2001 in St. Pölten

Josef Böhm, Würmla

As many of the non Austrain readers might know, we don't have national exams in Austria. Each school - and sometimes each teacher - sets up his/her tasks for the end exams (= "*Matura*") on the various forms of Secondary Schools. In my school - the "Bundeshandelsakademie St. Pölten - we always set up a common exam. This year we were three - Tania Koller, Günter Schraik and I - to teach in the last forms and we decided to give three tasks: one from statistics with a bit probability theory, one from financial mathematics (investment calculation) and one from analysis. The working time is 4 hours.

It was on me to set the analysis problem. Inspired by Carl Leinbach's lecture in Gösing and our following discussion and a by a problem set by Sibylle Stachniss-Carp I posed the problem as follows:

1cm ≙ 600m

In the region north of River Danube a golf course is planned. Its northern border is formed by a newly to build street which shall near *Winkl* (point W) continue the straight - also to be constructed - street S_1. Then it shall pass point O to finally near *Absdorf* (point A) run into the - also to be constructed - road S_2.

a) In first approximation find the line of construction of the northern border, which shall be described by a polynomial function of lowest degree. The connections between this road and S_1 and S_2 must be smooth.
The data must be read off from the map as accurate as possible. What is the function term?
Give reasons for your choice of the polynomial degree.

b) The connection between roads should not be only smooth - i.e. differentiable, but one has to avoid sudden changes of curvature, too. In order to reach this goal the curves must have the second derivatives in common.
Which polynomial function will describe the run of the road now. Use this function for the next tasks.

c) Draw by hands the new road on the map using at least 6 additional points.
Define the line S_1 - new road - S_2 as a piecewise defined function. Write down the correct syntax.

d) What is the difference in the lengths of both models?

e) Which area is available for the course if the southern border is formed by an existing road S (connecting road between W and A)? In 1st approximation you can use the airline between W and A.

f) What do you think? Is the real area greater or less that the value calculated? Try to improve your result in a not too difficult way. Describe shortly your choice of improvement.

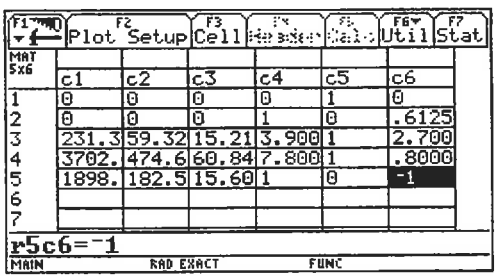
Proposal for a solution

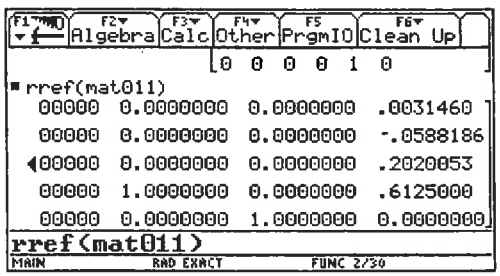
a) Most of the students chose the origin in W with x - axis horizontal and y-axis vertical, but there were also some of them who chose line WA as x-axis with W as origin. Let´s follow the majority:

We find by measuring W(0 | 0), O(3.9 | 2.7) and A(7.8 | 0.8), the slope of S_1 is either found by rise/run $\approx$ 0.6125 or tan(31°). To model the smooth connection in A one has to draw the tangent on S_2 using a ruler and read off the slope as $\approx$-1, So we obtain material to set up 5 equations - to determine 5 variables, which leads to the conclusion to set up a polynomial of degree 4: $y = a\,x^4 + b\,x^3 + c\,x^2 + d\,x + e$.

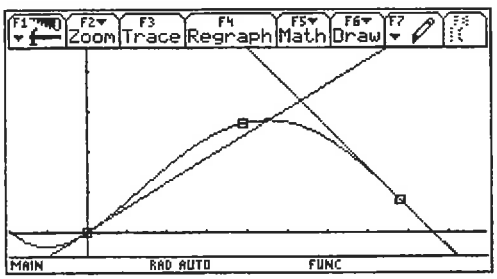
$y(0) = 0$	→	$e = 0$
$y'(0) = 0.6125$	→	$d = 0.6125$
$y(3.9) = 2.7$	→	$3.9^4\,a + \ldots\ldots = 2.7$
$y(7.8) = 0.8$	→	$7.8^4\,a + \ldots\ldots = 0.8$
$y'(7.8) = -1$	→	$4 \cdot 7.8^3 + \ldots\ldots = -1$

The solution of the system of linear equations was obtained by using the `rref()` command for producing the row reduced echelon form:





$$y = 0.0032460\,x^4 - 0.0588186\,x^3 + 0.2020053\,x^2 + 0.6125\,x$$



b) Two additional conditions give to more equations → two more parameters → polynomial of degree 6.

$y''(0) = 0$

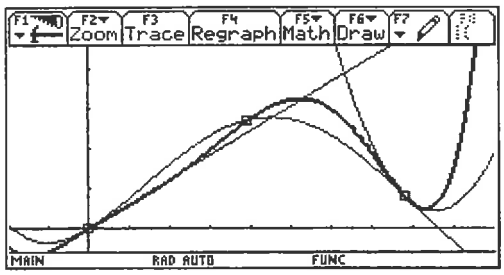
To find an approximation for the 2nd derivative we need a quadratic approximation for S_2 in point A. The students read off three points on S_2 close by and including A: A(7.8 | 0.8), B(8.3 | 0.45), C(8.9 | 0.5).

Using the regression tool we find the parabola passing A, B and C: $y = 0.7121\,x^2 - 12.1652\,x + 52.3627$

Hence $y''(7.8) = 1.4242$ and we improve $y'(7.8) = -1.056$.

The matric 7 × 8 matrix must be filled and using `rref()` again we find the solutions for the parameters a through g in order to define the polynomial of degree 6:

$$y = 0.000309\,x^6 - 0.005205\,x^5 + 0.0245\,x^4 - 0.02947\,x^3 + 0.6125\,x.$$ (stored as `pol6(x)`)

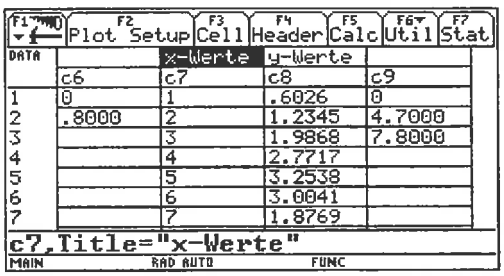


You can see both polynomials together with the graphic representation of S_1 and the approximating parabola of S_2.

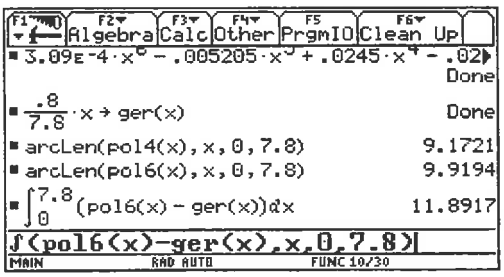
`when(x<0,0.6x,when(x<7.8,y2(x),y3(x)))`

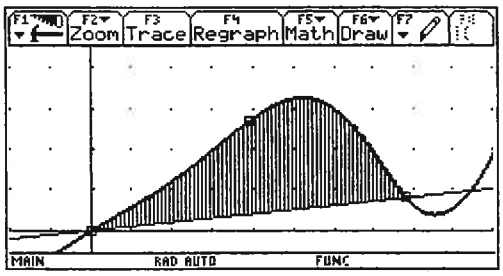
Line WA: $y = \frac{0.8}{7.8}\,x$ (stored as `ger(x)`)

c) From the table we easily can obtain the coordinates of points of the curve to tranfer them onto the map:



d) The arc lengths can be found in the Graph Screen (F5) or in the Home Screen:





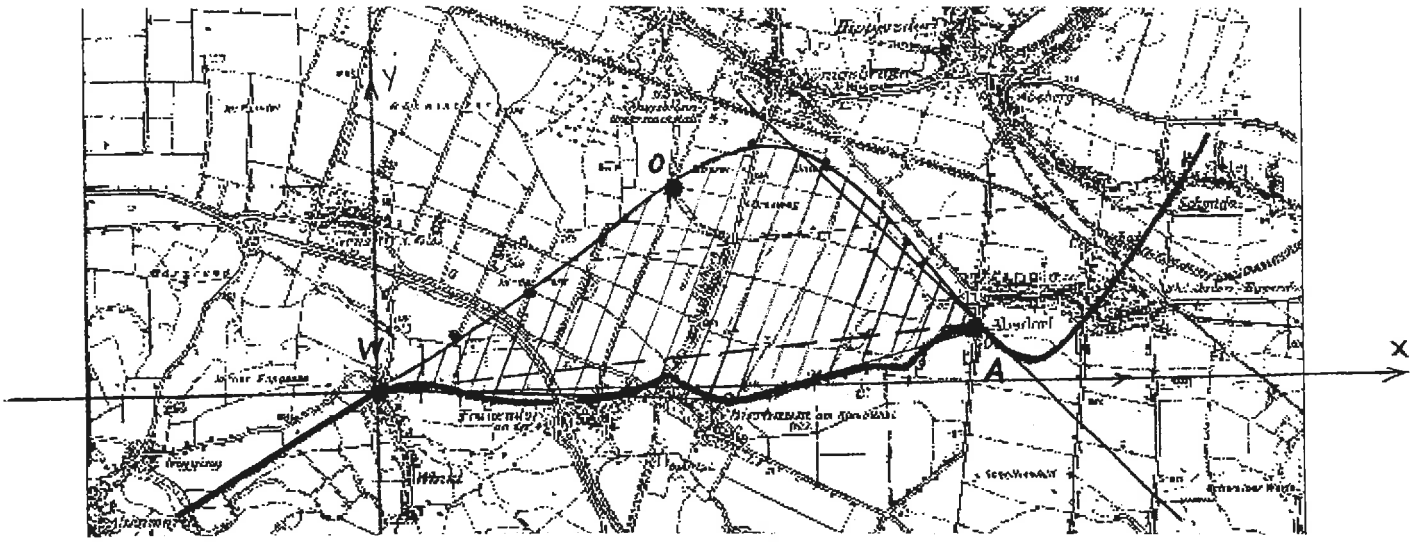
The lengths are 5.50km and 5.95km, so there is a difference of 450m.

e) The area is the integral shown above. But - not surprising for all of you who are teachers - the most difficult problem for the students was the conversion of the 11.89 cm^2 into km^2!!!

Area ≈ 11.89 × (600 m)^2 = 11.89 × (0.6 km)^2 = 4.3 km^2.

f) The real area is larger.

We can approximate the additional area by a triangle Δ: 8 × 0.6 / 2 = 2.4 cm^2 → 0.93 km^2 (total 5.23 km^2).
or we find another parabola passing W, A and point D(4.7 | -0.1) and then we use the TI to calculate the area between pol6(x) and this parabola giving a total sum of 5.17 km^2.



The students enjoyed this problem and most of them solved it in major parts. One of them - a girl - asked Tania some days after the examination for a copy of the problem, because she liked it and wanted to keep it as a memory.

Normal Distribution and Error Function with *DERIVE* and on the TI-92

Josef Lechner & Robert Kremslehner, Austria

Josef Lechner sent a DfW-file containing some questions concerning solving inverse problems with the Normal Distribution. Tasks like $P(400 - c \le 400 + c) = 0.95$ solving for c or $P(X \ge 700) = 0.95$ with a given $\sigma = 2$ solving for the mean μ take too much time on the TI and might cause also problems working with *DERIVE*.

Robert Kremslehner, one of Josef's students (final year on the gymnasium) found a way to overcome this deficiency by writing a tool for answering all the problems.

It must be said that we can solve all the problems on the TI-89 / TI-92+ applying the FLASH Statistics package with or without using the Numeric Solver. (You can find the respective screen shots at the end of the article).

See first Josef's *DERIVE*-file accomplished with some own findings. Then I add a copy of Robert Kremslehner's derivation for his Error Function approximation closing with the TI-code and some examples.

Density and Distributionfunction for the Normal Distribution

$$\#1: \quad NV(m, s, x) := \frac{1}{s \cdot \sqrt{(2 \cdot \pi)}} \cdot \hat{e}^{-1/2 \cdot ((x - m)/s)^2}$$

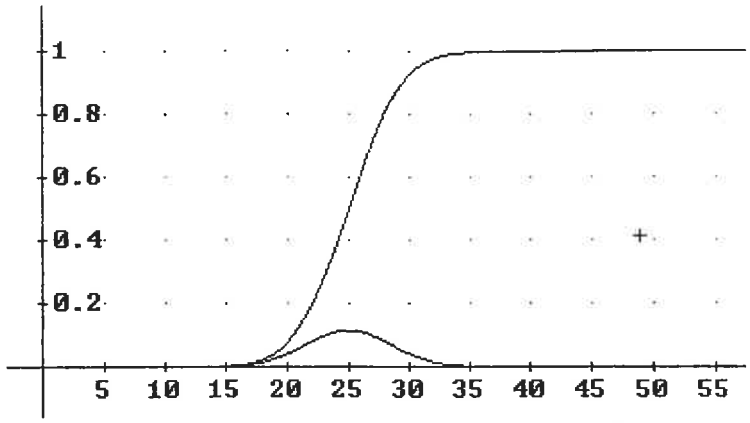
$$\#2: \quad F(m, s, x) := \int_{-\infty}^{x} NV(m, s, z)\, dz$$

#3: NV(25, 3.5)

#4: F(25, 3.5, x)

Problem: For a random variable X with mean = 500 find the standard deviation σ that $p(495 \le X \le 505) = 0.95$. Don't apply standardization!

Attempt #1: even in DERIVE 5 it doesn't work without the Error function ERF()!



#5: $\text{Norm}(m, s, a, e) := \int_{-\infty}^{e} \text{NV}(m, s, z)\, dz - \int_{-\infty}^{a} \text{NV}(m, s, z)\, dz$

#6: NSOLVE(Norm(500, σ, 495, 505) = 0.95, σ)

Cancelled because of receiving no result within a reasonable calculation time.

Attempt #2: Using the DERIVE built-in NORMAL-function - which can be compared with expression #2, but is "internally" built up in another way we obtain the solution immediately:

#7: Norm1(m, s, a, e) := NORMAL(e, m, s) - NORMAL(a, m, s)

#8: NSOLVE(Norm1(500, σ, 495, 505) = 0.95, σ)

Approx(#8):

#9: $\sigma = 2.55106$

But what is "behind" the NORMAL-function? The Error function!

User=Simp(User):

#10: $$\text{NORMAL}(x, m, s) = \frac{\text{ERF}\left(\frac{\sqrt{2}\cdot(x - m)}{2\cdot s}\right) + 1}{2}$$

My conclusion: without the built-in Error function you will not become happy working with the Normal Distribution.
(Weeping in his misery one Josef greets the other one!)

The "other" Josef loaded Josef Lechner's dfw-file and made some "funny" experiencies:

#11: NSOLVE(Norm(500, σ, 495, 505) = 0.95, σ)

Simp(#11'):

#12: $$\text{NSOLVE}\left(\text{ERF}\left(\frac{5\cdot\sqrt{2}}{2\cdot\sigma}\right) = \frac{19}{20},\ \sigma\right)$$

Approx(#12):

#13: $\sigma = 2.55106$

If I highlight subexpression Norm(...) in expression #11 and then simplify I receive #12, and approximating or simplifying this it takes 0.039 sec to have the result. Don't ask me why???

DERIVE seems to know our most secret thoughts and recognizes the Error function in our User defined NV-function - Wizardry and Sorcery!!??

I define another function for solving problems of this kind:

$$\#14:\quad FV(m,\ s,\ a,\ b)\ :=\ \int_a^b NV(m,\ s,\ z)\ dz$$

What is happening now?

```
#15:  NSOLVE(FV(500, σ, 495, 505) = 0.95, σ)
Simp(#15):
```

$$\#16:\qquad \sigma = -10\cdot ERF\left(\frac{0.707106\cdot(m-505)}{s}\right)\ \vee\ \sigma = 10\cdot ERF\left(\frac{0.707106\cdot(m-495)}{s}\right)$$

Don't ask me where the variables m and s are coming from??
But then using SOLVE followed by NSOLVE:

```
User:
#17:  SOLVE(FV(500, σ, 495, 505) = 0.95, σ)
Simp(#17):
```

$$\#18:\qquad \sigma\cdot\sqrt{\left(\frac{1}{\sigma^2}\right)}\cdot ERF\left(\frac{5\cdot\sqrt{2}\cdot\sqrt{\left(\frac{1}{\sigma^2}\right)}}{2}\right) = \frac{19}{20}$$

```
Solve(#18,σ):
```

$$\#19:\qquad NSOLVE\left(\sigma\cdot\sqrt{\left(\frac{1}{\sigma^2}\right)}\cdot ERF\left(\frac{5\cdot\sqrt{2}\cdot\sqrt{\left(\frac{1}{\sigma^2}\right)}}{2}\right) = \frac{19}{20},\ \sigma\right)$$

```
Simp(Solve(#18,σ)):
```

$$\#20:\qquad \sigma = 2.55106$$

Some other standard problems (and their solutions):

```
m = 400, s = 0.5, p(|X - 400| < ε = 0.95; find ε
m = μ, s = 2, p(X > 700) = 0.95; find the mean
m = 100, s = 6, p(X > a) = 0.1093; find a
```

```
#21:  NSOLVE(Norm(400, 0.5, 400 - ε, 400 + ε) = 0.95, ε)
```

$$\#22:\qquad NSOLVE\left(ERF(\sqrt{2}\cdot\epsilon) = \frac{19}{20},\ \epsilon\right)$$

$$\#23:\qquad \epsilon = 0.979981$$

```
#24:  NSOLVE(Norm(μ, 2, 700, ∞) = 0.95, μ)
```

$$\#25:\qquad NSOLVE\left(\frac{ERF\left(\frac{\sqrt{2}\cdot(\mu-700)}{4}\right)+1}{2} = \frac{19}{20},\ \mu\right)$$

$$\#26:\qquad \mu = 703.289$$

#27: NSOLVE(Norm(100, 6, a, ∞) = 0.1093, a)

Simp(#27'):

#28:

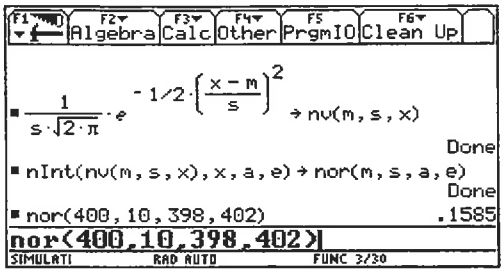
$$\text{NSOLVE}\left(\frac{1 - \text{ERF}\left(\frac{\sqrt{2}\cdot(a - 100)}{12}\right)}{2} = \frac{1093}{10000},\ a\right)$$

Approx(#28):

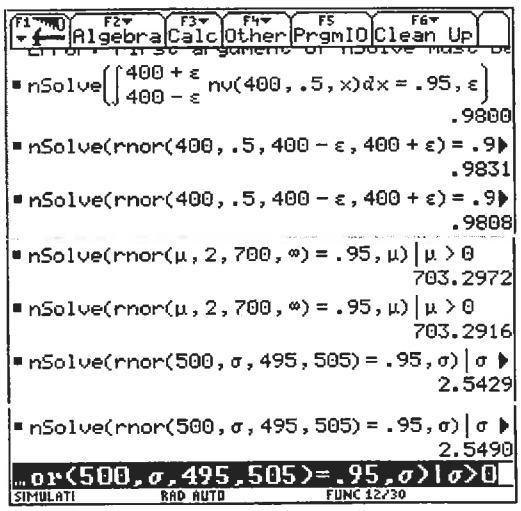
#29: a = 107.381

On the TI-92, TI-89 & TI-92+:

The two functions nv( ) and nor( ) are useful for calculating probabilities only. They cannot be used for solving inverse problems (finding the boundaries, the mean and the standard deviation).




```
rerf(x)
Func
If x≤-3.6 Then
  Return  -1
ElseIf x≥3.6 Then
  Return  1
Else
  Return  round(x/(√(π)*n)*(e^(-x^2)+1+2*Σ(e^(-i^2/n^2*x^2),i,1,n-1))|n=10)
EndIf
EndFunc

rnormal(m,s,x) = 1/2+rerf((√(2)*x-√(2)*m)/(2*s))/2

rnor(m,s,a,e) = rnormal(m,s,e)-rnormal(m,s,a)
```



more or less exact result.

n = 10 (default value)

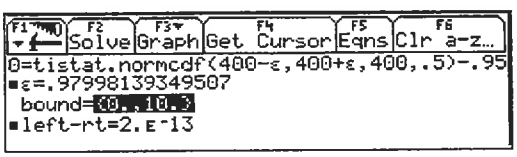
n = 20

n = 10

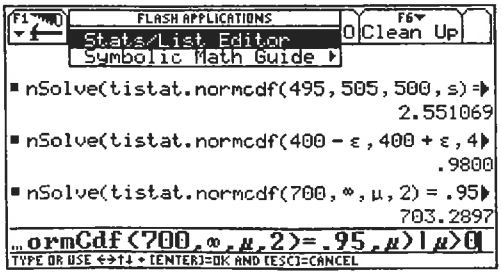
n = 20 (compare with the exact value see below!)

n = 10

n = 20



The last two screen shots are from a TI-92+ with very useful Flash Application "Stats/List Editor" installed.



$$F(x+\Delta x) = F(x) + \frac{f(x)+f(x+\Delta x)}{2}\cdot\Delta x$$

$$F(0) = F(0)$$

$$F(\Delta x) = F(0) + \frac{f(0)+f(\Delta x)}{2}\cdot\Delta x$$

$$F(2\Delta x) = F(0) + \frac{f(0)+f(\Delta x)}{2}\cdot\Delta x + \frac{f(\Delta x)+f(2\Delta x)}{2}\cdot\Delta x$$

$$\vdots$$

$$\Rightarrow F(n\cdot\Delta x) = F(0) + \frac{\Delta x}{2}\cdot\sum_{i=0}^{n-1} f(i\cdot\Delta x) + f((i+1)\cdot\Delta x)$$

$$n\cdot\Delta x = x \Rightarrow \Delta x = \frac{x}{n}$$

$$f(x) = \frac{2\cdot e^{-x^2}}{\sqrt{\pi}} \Rightarrow F(x) = ERF(x)$$

$$\Rightarrow ERF(x) = ERF(0) + \frac{x}{2n}\cdot\sum_{i=0}^{n-1} f\left(i\cdot\frac{x}{n}\right) + f\left((i+1)\cdot\frac{x}{n}\right)$$

DERIVE

$$\Rightarrow \frac{x\cdot e^{-x^2/n^2}\cdot\sum_{i=0}^{n-1} e^{-i^2\cdot x^2/n^2 - 2i\cdot x^2/n^2}}{\sqrt{\pi}\cdot n} + \frac{x\cdot\sum_{i=0}^{n-1} e^{-i^2\cdot x^2/n^2}}{\sqrt{\pi}\cdot n}$$

händisch vereinfacht

$$\boxed{\frac{x}{\sqrt{\pi}\cdot n}\cdot\left(e^{-x^2} + 1 + 2\cdot\sum_{i=1}^{n-1} e^{-i^2/n^2\cdot x^2}\right)}$$

(n=5 schon gute Werte)

$$\Rightarrow \lim_{n\to\infty}\left(\underbrace{\frac{x\cdot e^{-x^2}}{\sqrt{\pi}\cdot n}}_{\downarrow\ 0} + \frac{2x\cdot\sum_{i=1}^{n-1} e^{-i^2\cdot x^2/n^2}}{\sqrt{\pi}\cdot n} + \underbrace{\frac{x}{\sqrt{\pi}\cdot n}}_{\downarrow\ 0}\right)$$

$$\Rightarrow 0 + \frac{2x}{\sqrt{\pi}\cdot n}\cdot\sum_{i=1}^{n-1} e^{-i^2(x^{-2}/n^{-2})} + 0$$

$$\Rightarrow \underline{ERF(x) = \lim_{n\to\infty}\frac{2x}{\sqrt{\pi}\cdot n}\cdot\sum_{i=1}^{n-1} e^{-i^2\cdot x^2/n^2}}$$

Johann Wiesenbauer presented in an earlier DNL his "Macro for Solving Equations". I received some questions, why this nice tool doesn't work any longer in version 5. I posed Johann the problem and he told me that it is only necessary to define some variables as global (expr. #1). Here is the latest version of his Macro:

A Macro for Solving Equations (version for DfW5)

((c) J.Wiesenbauer, Vienna)

```
#1:   [e :=, eq :=, o :=]

      DO(u) :=
        Prog
#2:       o := ADJOIN(u, o)
          eq := LIM(u, @, eq)

      expa(u) :=
        Prog
#3:       o := ADJOIN(u, o)
          eq1 := EXPAND(eq)
          eq := eq1

      UNDO(u) :=
        Prog
#4:       o := DELETE(o)
          eq := LIM(ITERATE(u, @, @, -1), @, eq)

      RECORD(u) :=
        Prog
#5:       o := []
          eq := u
          e := u

      PLAY := PROG(o := REVERSE(o), eq := [[e, [FIRST(o)]]], LOOP(IF(o = [], RETURN
#6:

        REVERSE(eq)), e := lim FIRST(o), o := REST(o), eq := ADJOIN([e, IF(DIM(o) > 0,
                           @→e

        [FIRST(o)], )], eq)))
```

A simple example: Always start with simplifying RECORD(equation):

#7: RECORD($3^{2\cdot x+1} - 7 = 20$)

#8: $3^{2\cdot x+1} - 7 = 20$

#9: $DO(@ + 7) = (3^{2\cdot x+1} = 27)$

#10: $DO(LOG(@, 3)) = (2\cdot x + 1 = 3)$

#11: $DO\left(\frac{@}{2}\right) = \left(\frac{2\cdot x+1}{2} = \frac{3}{2}\right)$

#12: Oops!!

#13: $UNDO\left(\frac{@}{2}\right) = (2\cdot x + 1 = 3)$

#14: $DO(@ - 1) = (2\cdot x = 2)$

#15: $DO\left(\frac{@}{2}\right) = (x = 1)$

Call PLAY to replay your solving process:

#16: $$\text{PLAY} = \begin{bmatrix} 3^{2\cdot x+1} - 7 = 20 & [@ + 7] \\ 3^{2\cdot x+1} = 27 & \left[\frac{\text{LN}(@)}{\text{LN}(3)}\right] \\ 2\cdot x + 1 = 3 & [@ - 1] \\ 2\cdot x = 2 & \left[\frac{@}{2}\right] \\ x = 1 & \end{bmatrix}$$

A 2nd Example:

#17: $$\text{RECORD}\left(\frac{2\cdot x-3}{15} - \frac{4\cdot x-9}{20} = \frac{8\cdot x-27}{30} - \frac{16\cdot x-81}{24} - \frac{9}{40}\right)$$

#18: $$\frac{15-4\cdot x}{60} = \frac{45-8\cdot x}{20}$$ But:

#19: $$\text{RECORD}\left('\left(\frac{2\cdot x-3}{15} - \frac{4\cdot x-9}{20} = \frac{8\cdot x-27}{30} - \frac{16\cdot x-81}{24} - \frac{9}{40}\right)\right)$$

#20: $$\frac{2\cdot x-3}{15} - \frac{4\cdot x-9}{20} = \frac{8\cdot x-27}{30} - \frac{16\cdot x-81}{24} - \frac{9}{40}$$

#21: $$\text{DO}(@\cdot 120) = (2\cdot(15 - 4\cdot x) = 6\cdot(45 - 8\cdot x))$$

#22: $$\text{DO}\left(\frac{@}{2}\right) = (15 - 4\cdot x = 3\cdot(45 - 8\cdot x))$$

#23: $$\text{DO}(@ - 15 + 24\cdot x) = (20\cdot x = 120)$$

#24: $$\text{DO}\left(\frac{@}{20}\right) = (x = 6)$$

#25: $$\text{PLAY} = \begin{bmatrix} \frac{15-4\cdot x}{60} = \frac{45-8\cdot x}{20} & [120\cdot @] \\ 2\cdot(15 - 4\cdot x) = 6\cdot(45 - 8\cdot x) & \left[\frac{@}{2}\right] \\ 15 - 4\cdot x = 3\cdot(45 - 8\cdot x) & [24\cdot x + @ - 15] \\ 20\cdot x = 120 & \left[\frac{@}{20}\right] \\ x = 6 & \end{bmatrix}$$

Take notice of the **Quote Operator** ' in expression #19 to supress evaluation of the fractions.
Unfortunately it is not possible to apply EXPAND, FACTOR, ... on the "memorized" @.

Purpose of the Conference

In 1992 ACDCA (the Austrian Center for Didactics of Computer Algebra) started a conference series which has become a driving force in bringing technology, in particular computer algebra systems (CAS), into the classroom.

The conference series comprised two strands: the ACDCA Summer Academies, which were more oriented towards didactical questions connected with the use of technology for teaching and learning and the Int'l Derive (& TI-89/92) Conferences, which were more oriented towards concrete applications of these and similar tools for teaching in schools and at universities. In summer 2002 for the first time the two events will be held at the same time at the same place.

The conference program will comprise keynote lectures, contributed presentations, workshops, and whatever else participants would like to do.

Conference Co-Chairs

Helmut Heugl, ACDCA, Austria
hheugl@netway.at
Hans-Christian Reichel, Univ of Vienna, Austria
reichel@radon.mat.univie.ac.at

Central Planning Committee

Josef Böhm, Helmut Heugl, Bernhard Kutzler

Social Program Chair

Nancy Köstlbauer, Tulln, Austria

Website

To be announced / look at www.acdca.ac.at

VISIT-ME / 7th ACDCA Summer Acacemy

Call for Submissions

Deadline: 31 March 2002

Topics include:

- CAS-enabling curricula
- CAS-supported teaching methods
- Assessing with technology
- Dispensable and indispensable skills, abilities, and competencies
- CAS as pedagogical tools for visualization, trivialization, experimentation, concentration ...
- New classroom examples using CAS
- The next generation: PeCAS = Pedagogical CAS

Please indicate what type of presentation you want to make (lecture, workshop, ...). Send a title and an abstract (10-20 lines) to:

b.kutzler@eunet.at

VISIT-ME / 5th Derive & TI-89/92 Conf.

Call for Submissions

Deadline: 31 March 2002

Topics include:

- Applications in the natural sciences, research&development, economy, social sciences, industry, ...
- Interfaces to other programs/tools
- Producing and using utility files
- Programming in Derive and the TI-89/92
- Problems and limitations
- Web resources - going online
- Teaching and Learning with Derive and the TI-89/92

Please indicate what type of presentation you want to make (lecture, workshop, ...). Send a title and an abstract (10-20 lines) to:

nojo.boehm@pgv.at

VISIT-ME

Vienna Int'l Symposium on Integrating Technology into Mathematics Education

combining the
7th ACDCA Summer Academy
and the
5th Int'l Derive & TI-89/92 Conference

organized by

ACDCA
Austrian Center for Didactics of Computer Algebra

in cooperation with the
Institute of Mathematics
at the University of Vienna

10-13 July 2002
Vienna - AUSTRIA

1st Announcement
&
Call for Submissions

History of the Conference

The past conferences were:

- 1992: 1st ACDCA Summer Academy, Krems, Austria
- 1993: 2nd ACDCA Summer Academy, Krems, Austria
- 1994: 1st Int'l Derive Conference, Plymouth, UK
- 1995: 3rd ACDCA Summer Academy, Honolulu, USA
- 1996: 2nd Int'l Derive Conference, Bonn, Germany
- 1997: 4th ACDCA Summer Academy, Sarö, Sweden
- 1998: 3rd Int'l Derive & TI-89/92 Conference, Gettysburg, USA
- 1999: 5th ACDCA Summer Academy, Gösing, Austria
- 2000: 6th ACDCA Summer Academy, Portoroz, Slovenia & 4th Int'l Derive & TI-89/82 Conference, Liverpool, UK

Keynote Speakers

.....
to be announced
.....
.....

Technical Support Team

Walter Wegscheider (website)
.....
rest to be announced
.....

VISTIME / 7th ACDCA Summer Acacemy
Program Committee

Vlasta Kokol-Voljc, Slovenia (Co-Chair)
Bernhard Kutzler, Austria (Co-Chair)
.....
members to be announced
.....
.....
.....
.....
.....
.....

VISTIME / 5th Derive & TI-89/92 Conf.
Program Committee

Josef Böhm, Austria (Co-Chair)
Carl Leinbach, USA (Co-Chair)
.....
members to be announced
.....
.....
.....
.....
.....
.....

Tentative timetable

- 31 March 2002: Deadline for submissions
- 14 April 2002: Notification of acceptance
- 15 April 2002: Deadline for early bird registration
- 30 May 2002: Deadline for standard registration

Local Travel Agent

Scheuringer, Kuoni, Stockerau

Conference Headquarter Hotel

to be announced

Relevant Tourist Internet Addresses

www.wien.at
www.vienna.at

Neighboring Conference

1-6 July 2002
2nd International Conference on the Teaching of Mathematics (ICTM)
Crete, Greece
www.math.uoc.gr/~ictm2

Sponsors

Soft Warehouse Europe