Como ensinar Matemática Discreta para calouros com português muito ruim

Eduardo Ochs (UFF, Rio das Ostras, RJ, Brazil)
http://angg.twu.net/math-b.html#ebl2019-m

Mesa redonda: "Algumas abordagens para disciplinas de Lógica em cursos de graduação"
Resumo (da minha apresentação na mesa):

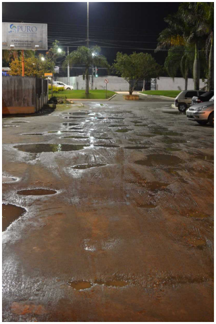
Considere a seguinte situação: um curso de Matemática Discreta para calouros, com conteúdo muito grande, num campus em que entram muitos alunos muito fracos com muito pouca base matemática e português muito ruim; além disso nos pedem pra que ensinemos o máximo possível nesse curso e não deixemos passar alunos que não tiverem aprendido o suficiente, porque este curso de Matemática Discreta é pré-requisito para cursos que exigem uma certa maturidade matemática, e ele deve servir como "filtro"... como estruturar um curso assim pra que ele funcione bem?

(Resumo, continuação)
O problema mais básico aqui é: que linguagem utilizar? Os alunos sempre começam tentando discutir suas idéias em português, tanto entre si quanto com o professor, mas o português deles é impreciso demais, e não há tempo hábil para debugá-lo no curso! Então precisamos ir introduzindo desde o início notações matemáticas precisas que sejam fáceis o suficiente de usar, e aí usar sempre exemplos em notação matemática... a linguagem matemática adequada vira a base que torna as discussões em português possíveis.

(Resumo, continuação)
Nesta palestra vou mostrar como esta "linguagem matemática adequada para calouros" pode ser dividida em três camadas: uma camada mais básica, em que tudo pode ser calculado até o resultado final num número finito de passos sem precisar de criatividade praticamente nenhuma; uma camada média, em que alguns objetos infinitos, como $\mathbb{N}$ ou $\mathbb{Z}$, passam a ser permitidos; e uma camada acima destas que inclui uma linguagem para provas formais. Além disto vou mostrar modos de visualizar vários dos objetos matemáticos do curso, e discutir alguns pontos em que esta abordagem do curso ainda tem "buracos".

Discrete Mathematics at PURO/UFF

I teach in a campus that UFF has in Rio das Ostras, in the countryside of the state of Rio de Janeiro... The entrance of the campus looks like this:



⇐ "PURO": Pólo Universitário de Rio das Ostras

(Let's start with) Discrete Mathematics

I teach in a city called Rio das Ostras, in the countryside of the state of Rio de Janeiro, in a big federal university ("UFF"), but in one of its smallest campi, away from the capital.
I sometimes teach Discrete Mathematics to Computer Science students there.

Many of the students there — even in CompSci — enter the university with very little knowledge of:
1) how to deal with variables,
2) how to write their calculations,
3) how to test their ideas.

Discrete Mathematics is a first-semester course there.
Let me explain my approach to fix (1), (2), and (3).

Discrete Mathematics at PURO/UFF
I structured the Discrete Mathematics ("DM") course in three layers.

Layer 1 consists of calculating things and learning how to use mathematical notation and definitions.
Layer 2 introduces some infinite objects, like $\mathbb{N}$ and $\mathbb{Z}$.
Layer 3 introduces a formal language for doing proofs.

Everything in Layer 1 can be calculated in a finite number of steps with very little creativity.

One of the basic things that we learn in Layer 1 is **set comprehensions** — that (ta-daaa! Magic!!!!!!) are the base for λ-calculus and Type Theory.

Basic Mathematical Objects
Here's a definition ("for adults") of the mathematical objects that we deal with in Level 1. Notation: $\mathcal{O}$ is the set of BMOs.

a) Numbers belong to $\mathcal{O}$; $\mathbb{Z} \subset \mathcal{O}$.
b) The truth-values belong to $\mathcal{O}$: $\{\mathbf{T}, \mathbf{F}\} \subset \mathcal{O}$.
c) $\mathcal{O}$ is closed by "finite lists" and "finite sets".
d) Finite strings belong to $\mathcal{O}$.

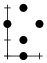
Item (d) is only introduced at the end of the course, when we show that "valid expressions" can be defined formally... "`1+2*(3+4)`" is "valid", but "`)+)`" is not.

Some graphical representations are allowed.

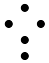
Some graphical representations

We define graphical representations for:
a) (finite) subsets of $\mathbb{Z}^2$,
b) functions whose domains are (a),
c) directed graphs...

$$K = \left\{ \begin{array}{ccc} & (1,3), & \\ (0,2), & & (2,2), \\ & (1,1), & \\ & (1,0) & \end{array} \right\} =$$



$$=$$



$$f = \left\{ \begin{array}{ccc} & ((1,3),1), & \\ ((0,2),0), & & ((2,2),2), \\ & ((1,1),1), & \\ & ((1,0),1) & \end{array} \right\} = \begin{array}{ccc} & 1 & \\ 0 & & 2 \\ & 1 & \\ & 1 & \end{array}$$

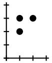
$$(V, A) = \left(\{1,2,3,4,5\}, \left\{ \begin{array}{c} (1,2),(1,3), \\ (2,4),(3,4), \\ (4,5) \end{array} \right\} \right) = \begin{tikzcd} & 1 \arrow[dl] \arrow[dr] & \\ 2 \arrow[dr] & & 3 \arrow[dl] \\ & 4 \arrow[d] & \\ & 5 & \end{tikzcd}$$

Layer 1: Set Comprehensions

One of the first things that I present to students is a syntax for set comprehensions using “generators”, “filters” and a “result expression”...

$$\{\underbrace{\underbrace{a \in \{1,2\}}_{\text{gen}}, \underbrace{b \in \{2,3\}}_{\text{gen}}, \underbrace{a < b}_{\text{filt}}}_{\text{context}}; \underbrace{(a,b)}_{\text{expr}}\} \quad = \quad \{(1,2),(1,3),(2,3)\}$$

$$=$$



Note the ‘;’ instead of a ‘|’!
These things can be calculated from left to right using trees in a finite number of steps.

Layer 1: Set Comprehensions (2)
I make the students work in groups, and they solve the $5 + 19 + 16 + 9 + 16 + 7$ exercises quickly.
I suggest this table-ish way to draw the trees...
To calculate
$\{\, a \in \{1, 2\}, b \in \{2, 3\}, a < b \;;\; 10a + b \,\} = \{12, 13, 23\}$
we draw:

a	b	$a < b$	$10a + b$
1	2	**T**	12
1	3	**T**	13
2	2	**F**	$\vert$
2	3	**T**	23

the vertical bar means "abort".
Valid values for the context: $(1, 2, \mathbf{T})$, $(1, 3, \mathbf{T})$, $(2, 3, \mathbf{T})$.

Layer 1: Set Comprehensions — SPOILER

Spoiler: the idea of context will be reused in many other contexts later, but with slightly different notations...
Compare:

1) $\{\, a \in \{1,2\}, b \in \{2,3\}, a < b \,;\; 10a + b \,\} = \{12, 13, 23\}$
2) $a{:}\{1,2\}, b{:}\{2,3\}, a < b \vdash (10a + b){:}\mathbb{N}$
3) $a{:}\{1,2\}, b{:}\{2,3\} \vdash a < b$ $\Leftarrow$ this is false!
4) $A{:}\mathsf{Sets}, B{:}\mathsf{Sets} \vdash A{:}\mathsf{Sets}$
5) $A{:}\mathsf{Sets}, B{:}\mathsf{Sets} \vdash A \times B{:}\mathsf{Sets}$
6) $A{:}\mathsf{Sets}, B{:}\mathsf{Sets}, a{:}A, f{:}A \to B \vdash f(a){:}B$
7) $A{:}\mathsf{Sets}, B{:}\mathsf{Sets}, a{:}A, b{:}B \vdash (a,b){:}A \times B$
8) $A{:}\mathsf{Sets}, B{:}\mathsf{Sets}, a{:}A \vdash (\lambda b{:}B.(a,b)){:}B \to A \times B$

An abuse of language

Remember that I make the students
do $5+19+16+9+16+7$ exercises about set comprehensions...
some of the exercises are about cartesian products, and
some introduce new notations, like:

$\{x \in \{2,3,4\}, y \in \{2,3,4\}, y=3; (x,y)\}$
$\{x, y \in \{2,3,4\}, y=3; (x,y)\}$
$\{(x, y) \in \{2,3,4\}^2, y=3; (x,y)\}$

This abuse of language will be incredibly important later
when we discuss type systems. We will allow things like:

$A{:}\mathsf{Sets}, B{:}\mathsf{Sets}, (a,b){:}A \times B, f{:}A \to B \vdash fa{:}B$

How do we make (a,b) behave as a variable?
(In the singular!)

An abuse of language (2)

Short answer:
(a, b) becomes a "long name" for a variable p,
a becomes an abbreviation for $\pi(a, b)$, i.e., πp,
b becomes an abbreviation for $\pi'(a, b)$, i.e., $\pi' p$,

$A{:}\mathsf{Sets}, B{:}\mathsf{Sets}, (a, b){:}A \times B, f{:}A \to B \vdash f(a){:}B$
becomes:
$A{:}\mathsf{Sets}, B{:}\mathsf{Sets}, p{:}A \times B, f{:}A \to B \vdash f(\pi p){:}B$

(More on "long names" later!)

λ-notation in Discrete Mathematics

In DM we see functions as sets of input-output pairs...

$$\begin{array}{llcl} \text{If} \quad f: & \{1,2,3\} & \to & \{10,20,30\} \\ & x & \mapsto & 10x \end{array}$$

then $f = \left\{ \begin{smallmatrix} (1,10), \\ (2,20), \\ (3,30) \end{smallmatrix} \right\}$.

The students learn that, e.g.,

$\left\{ \begin{smallmatrix} (1,10), \\ (2,20), \\ (3,30) \end{smallmatrix} \right\}(2) = 20$, and $\left\{ \begin{smallmatrix} (1,10), \\ (2,20), \\ (3,30) \end{smallmatrix} \right\}(4) =$ **ERROR**.

We see (in passing) that

$(\lambda x \in \{1,2,3\}.10x) = \{x \in \{1,2,3\}; (x, 10x)\} = \left\{ \begin{smallmatrix} (1,10), \\ (2,20), \\ (3,30) \end{smallmatrix} \right\}$.

(Simultaneous) substitution
I also teach, right in the beginning of the course,
a notation for (simultaneous) substitution...
(Because I can't rely on the students' Portuguese!...)
Examples:

$$
\begin{array}{rl}
& ((x+y)\cdot z)\begin{bmatrix} \scriptstyle x:=a+y \\ \scriptstyle y:=b+z \\ \scriptstyle z:=c+x \end{bmatrix} \\
= & ((a+y)+(b+z))\cdot(c+x)
\end{array}
$$

$$
\begin{array}{rl}
& \text{(Vanessão 20 reais)}\left[\text{a} := \left(\begin{smallmatrix} \int \odot \\ \square \end{smallmatrix}\right)\right] \\
= & \text{(V}\left(\begin{smallmatrix} \int \odot \\ \square \end{smallmatrix}\right)\text{ness}\widetilde{\left(\begin{smallmatrix} \int \odot \\ \square \end{smallmatrix}\right)}\text{o 20 re}\left(\begin{smallmatrix} \int \odot \\ \square \end{smallmatrix}\right)\text{is)}
\end{array}
$$

(Simultaneous) substitution (2)
This is useful to test equations,

$$\begin{array}{l}(x^2 - 4 = 0)\left[x := 3\right] = (9 - 4 = 0) = \mathbf{F}\\(x^2 - 4 = 0)\left[x := 2\right] = (4 - 4 = 0) = \mathbf{T}\end{array}$$

and to define a way to calculate expressions with quantifiers:

$$\begin{array}{lcl}(\forall a \in \{2,3,5\}. a^2 < 10) & = & (a^2 < 10)[a := 2] \wedge \\ & & (a^2 < 10)[a := 3] \wedge \\ & & (a^2 < 10)[a := 5] \\ & = & (2^2 < 10) \wedge (3^2 < 10) \wedge (4^2 < 10) \\ & = & (4 < 10) \wedge (9 < 10) \wedge (16 < 10) \\ & = & \mathbf{T} \wedge \mathbf{T} \wedge \mathbf{F} \\ & = & \mathbf{F}\end{array}$$

Substitution and λ-calculus
...and I mention, very briefly, that we can use substitution to calculate with λ-terms:

$$\begin{aligned}(\lambda x \in \{1,2,3\}.10x)(2) &= (10x)[x := 2] \\ &= 10 \cdot 2 \\ &= 20\end{aligned}$$

Some students later take an optional seminar course that is a very basic introduction to λ-calculus and Category Theory... In it they learn how to handle untyped λ-terms...

Discrete Mathematics at PURO/UFF (2)

Discrete Mathematics is taught to first-semester
Computer Science students there.
Most of these students enter the university with
very little knowledge of how to use variables,
and when they attempt to spell out their ideas in Portuguese,
either speaking or writing, they are too imprecise...
They ask “can I do xxx yyy?” and if I say “yes” they
write weird atrocities in the test, and when I mark them
as wrong they blame me (“you said I could do that!”);
if I say “no, you can’t do that” then they do other
atrocites and blame me, too.
I don’t have time to debug their Portuguese!

Discrete Mathematics at PURO/UFF (3)

...so I decided that I would stress mathetical notation —
Almost all ideas on the blackboard (whiteboard, actually)
would have examples in mathematical notation.

I structured the course of Discrete Mathematics
in three layers (presented more or less in parallel):

Layer 1: Calculating things
in a system with numbers, truth-values, sets and lists
where everything can be calculated in a finite number
of steps with almost no creativity required.

Layer 2: some infinite objects, like $\mathbb{N}$ and $\mathbb{Z}$, are allowed;
we learn how to "calculate", e.g., $\exists k \in \mathbb{Z}.10k = 23$

Layer 3: we add formal proofs.

Layer 1: Calculating things

...in a system with numbers, truth-values, sets and lists where everything can be calculated in a finite number of steps with almost no creativity required.

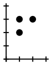
Example:

$$\begin{aligned}(\forall a \in \{2,3,5\}.a^2 < 10) &= (a^2 < 10)[a := 2] \wedge \\ &\quad\ (a^2 < 10)[a := 3] \wedge \\ &\quad\ (a^2 < 10)[a := 5] \\ &= (2^2 < 10) \wedge (3^2 < 10) \wedge (4^2 < 10) \\ &= (4 < 10) \wedge (9 < 10) \wedge (16 < 10) \\ &= \mathbf{T} \wedge \mathbf{T} \wedge \mathbf{F} \\ &= \mathbf{F}\end{aligned}$$

Note the substituion operator:
$(a^2 < 10)[a := 3] = (3^2 < 10)$.

Layer 1: Set Comprehensions
I wrote a lengthy explanation of set comprehensions, using "generators", "filters" and a "result expression". The students started by learning how to calculate things like these (note the ';' instead of a '|'; these '$\{\ldots;\ldots\}$'s are calculated from left to right!):

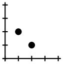
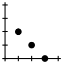
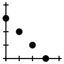
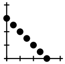
$$\begin{aligned}&\{\underbrace{a\in\{1,2\}}_{\text{gen}},\underbrace{b\in\{2,3\}}_{\text{gen}},\underbrace{a\neq b}_{\text{filt}};\underbrace{(a,b)}_{\text{expr}}\}\\ =\;&\{(1,2),(1,3),(2,3)\}\\ =\;&\end{aligned}$$

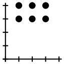
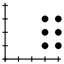
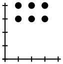
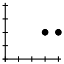


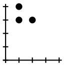
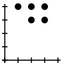
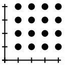
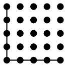
...then $\{\,a\in\{2,3,4\}\mid a^2<10\,\}$
and $\{\,10a+b\mid a\in\{1,2\},b\in\{3,4\}\,\}$.

Layer 1: Set Comprehensions (2)

The “lengthy explanation of set comprehensions”
using “generators”, “filters” and a “result expression”, has:
2 pages of explanations,
2 pages of exercises (5+19+16+9+16+7 exercises),
1 page of answers, all using a graphical notation —
for example:

2) $A =$  $B =$  $C =$  $D =$ 

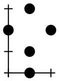
$E =$  $F =$  $G =$  $H =$ 

$I =$  $J =$  $K =$  $L =$ 

Positional notations

...then we adapt the graphical notation for subsets of $\mathbb{Z}^2$...
we define a convention for omitting the axes,

$$K = \left\{ \begin{array}{ccc} & (1,3), & \\ (0,2), & & (2,2), \\ & (1,1), & \\ & (1,0) & \end{array} \right\} =$$



$$= \begin{array}{ccc} & \bullet & \\ \bullet & & \bullet \\ & \bullet & \\ & \bullet & \end{array}$$

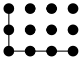
we define a positional notation for functions,

$$f = \left\{ \begin{array}{ccc} & ((1,3),1), & \\ ((0,2),0), & & ((2,2),2), \\ & ((1,1),1), & \\ & ((1,0),1) & \end{array} \right\} = \begin{array}{ccc} & 1 & \\ 0 & & 2 \\ & 1 & \\ & 1 & \end{array}$$

and for subsets, partial functions, and characteristic functons:

$$\begin{array}{ccc} & \bullet & \\ \cdot & & \bullet \\ & \bullet & \\ & \cdot & \end{array} \subset \begin{array}{ccc} & \bullet & \\ \bullet & & \bullet \\ & \bullet & \\ & \bullet & \end{array} \qquad \begin{array}{ccc} & 1 & \\ \cdot & & 2 \\ & 1 & \\ & \cdot & \end{array} \qquad \begin{array}{ccc} & 1 & \\ 0 & & 1 \\ & 1 & \\ & 0 & \end{array}$$

Propositions

Let $W = \{0,1,2,3\} \times \{0,1,2\} =$  .

(Our "set of worlds").

A *proposition on A* is a function $P : A \to \{\mathbf{F}, \mathbf{T}\}$.

Let P, Q, R be these propositions on W:

$P = \{\, ((x,y),z) \mid (x,y) \in W, z = (x \leq 1 \land y \geq 1) \,\}$

$Q = \{\, ((x,y),z) \mid (x,y) \in W, z = (1 \leq x \leq 2 \land y \geq 1) \,\}$

$R = \{\, ((x,y),z) \mid (x,y) \in W, z = (0 \leq x \leq 2 \land y \leq 1) \,\}$

or:

$$P = P(x,y) = (x \leq 1 \land y \geq 1) \quad = \quad \begin{matrix} \mathbf{T} & \mathbf{T} & \mathbf{F} & \mathbf{F} \\ \mathbf{T} & \mathbf{T} & \mathbf{F} & \mathbf{F} \\ \mathbf{F} & \mathbf{F} & \mathbf{F} & \mathbf{F} \end{matrix}$$

$$Q = Q(x,y) = (1 \leq x \leq 2 \land y \geq 1) \quad = \quad \begin{matrix} \mathbf{F} & \mathbf{T} & \mathbf{T} & \mathbf{F} \\ \mathbf{F} & \mathbf{T} & \mathbf{T} & \mathbf{F} \\ \mathbf{F} & \mathbf{F} & \mathbf{F} & \mathbf{F} \end{matrix}$$

$$R = R(x,y) = (0 \leq x \leq 2 \land y \leq 1) \quad = \quad \begin{matrix} \mathbf{F} & \mathbf{F} & \mathbf{F} & \mathbf{F} \\ \mathbf{T} & \mathbf{T} & \mathbf{T} & \mathbf{F} \\ \mathbf{T} & \mathbf{T} & \mathbf{T} & \mathbf{F} \end{matrix}$$

Propositions (2)

In a (long) series of exercises the students learned to visualize these and lots of other propositions on W — actually this set of propositions,

$\mathcal{S} = \{\top,\ \bot,\ P,\ Q,\ R,\ P \land Q,\ P \lor Q,\ P \to Q\}$

and I asked them to draw the Hasse diagram of the partial order on $\mathcal{S}$. They got this:

$$\begin{tikzcd}
 & \top & & & \\
 & P\lor Q \arrow[u] & & P\to Q \arrow[ull] & R \arrow[ulll, bend right] \\
P \arrow[ur] & & Q \arrow[ul] \arrow[ur] & & \\
 & P\land Q \arrow[ul] \arrow[ur] & & & \\
 & \bot \arrow[u] \arrow[uuurrr, bend right] & & &
\end{tikzcd}$$

Propositions (3)

Using '0's and '1's instead of '**F**'s and '**T**'s, what they got was:

$$\begin{tikzcd}
 & \begin{smallmatrix}1111\\1111\\1111\end{smallmatrix} & & & \\
 & \begin{smallmatrix}1110\\1110\\0000\end{smallmatrix} \arrow[u] & & \begin{smallmatrix}0111\\0111\\1111\end{smallmatrix} \arrow[ulll] & \begin{smallmatrix}0000\\1110\\1110\end{smallmatrix} \arrow[ulll, bend right] \\
\begin{smallmatrix}1100\\1100\\0000\end{smallmatrix} \arrow[ur] & & \begin{smallmatrix}0110\\0110\\0000\end{smallmatrix} \arrow[ul] \arrow[ur] & & \\
 & \begin{smallmatrix}0100\\0100\\0000\end{smallmatrix} \arrow[ul] \arrow[ur] & & & \\
 & \begin{smallmatrix}0000\\0000\\0000\end{smallmatrix} \arrow[u] \arrow[uuurrr, bend right] & & &
\end{tikzcd}$$

$$\begin{tikzcd}
 & \top & & & \\
 & P\vee Q \arrow[u] & & P\to Q \arrow[ulll] & R \arrow[ulll, bend right] \\
P \arrow[ur] & & Q \arrow[ul] \arrow[ur] & & \\
 & P\wedge Q \arrow[ul] \arrow[ur] & & & \\
 & \bot \arrow[u] \arrow[uuurrr, bend right] & & &
\end{tikzcd}$$

Comprehensions → contexts → sequents
The part at the left of the ‘;’ in a ‘{. . . ; . . .}’
is called the “context”. For example:

$$\{\underbrace{\underbrace{a \in \{1,2\}}_{\text{gen}}, \underbrace{b \in \{2,3\}}_{\text{gen}}, \underbrace{a \neq b}_{\text{filt}}, \underbrace{c \in \{1,\ldots,a+b\}}_{\text{gen}}}_{\text{context}}; \underbrace{10a + c}_{\text{expr}}\}$$

The *set of possible values* for this context is:

$$\{\underbrace{\underbrace{a \in \{1,2\}}_{\text{gen}}, \underbrace{b \in \{2,3\}}_{\text{gen}}, \underbrace{a \neq b}_{\text{filt}}, \underbrace{c \in \{1,\ldots,a+b\}}_{\text{gen}}}_{\text{context}}; \underbrace{(a,b,c)}_{\text{expr}}\}$$

Comprehensions → contexts → sequents (2)

This is surprisingly similar to contexts in sequents!

$$\underbrace{\underbrace{a \in \mathbb{Z}}_{\text{gen}}, \underbrace{b \in \{1,2,3\}}_{\text{gen}}, \underbrace{\mathsf{prime}(4a+b)}_{\text{filt}}}_{\text{context}} \vdash b = 3$$

The sequent above can be seen as a (false!) proposition.
We used this in the course to debug proofs.
Our formal proofs were written as series of numbered lines,
each line starting by either "Suppose" of "Then".
Each "Then" line had an associated sequent —
its context was made of all the open "Suppose"s.
For each "Then" line in a valid proof its associated sequent
had to be a true proposition.

Dicas (do início do curso de Geometria Analítica)

Você **VAI TER QUE** aprender a definir seus objetos — pontos, retas, conjuntos, círculos, etc... isso provavelmente vai ser algo novo pra você e é algo que precisa de MUITO treino. Dá pra passar em Cálculo 1 e em Prog 1 só aprendendo a "ler" as definições que o professor e os livros mostram, mas em Geometria Analítica NÃO DÁ, em GA você vai ter que aprender a ler **E A ESCREVER** definições.

2) Cada "seja" ou "sejam" que aparece nestas folhas é uma definição, e você pode usá-los como exemplos de definições bem-escritas (éééé!!!!) pra aprender jeitos de escrever as suas definições.
3) Em "matematiquês" a gente quase não usa termos como "ele", "ela", "isso", "aquilo" e "lá" — ao invés disso a gente dá nomes curtos pros objetos ou usa expressões matemáticas

pra eles cujo resultado é o objeto que a gente quer (como nas pags __ e __)... mas *quando a gente está discutindo problemas no papel ou no quadro* a gente pode ser referir a determinados objetos *apontando pra eles com o dedo* e dizendo “esse aqui”.

Dan Ingalls interview on SmallTalk

It started to hit home in the Spring of '74 after I taught Smalltalk to 20 PARC nonprogrammer adults. They were able to get through the initial material faster than the children, but just as it looked like an overwhelming success was at hand, they started to crash on problems that didn't look to me to be much harder than the ones they had just been doing well on. One of them was a project thought up by one of the adults, which was to make a little database system that could act like a card file or rolodex. They couldn't even come close to programming it. I was very surprised because I "knew" that such a project was well below the mythical "two pages" for end-users we were working within. That night I worote it out, and the next day I showed all of them how to do it. Still, none of them were able to do it by themselves. Later, I sat in the room pondering the

board from my talk. Finally, I counted the number of nonobvious ideas in this little program. They came to 17. And some of them were like the concept of the arch in building design: very hard to discover, if you don't already know them.
The connection to literacy was painfully clear. It isn't enough to just learn to read and write. There is also a literature that renders ideas. Language is used to read and write about them, but at some point the organization of ideas starts to dominate mere language abilities. And it help greatly to have some powerful ideas under one's belt to better acquire more powerful ideas [Papert 70s]. So, we decided we should teach design. And Adele came up with another brillian stroke to deal with this. She decided that what was needed was in intermediary between the vague ideas about the problem and the very detailed writing and debugging that had to be done to get it

to run in Smalltalk. She called the intermediary forms design templates.