

MBL - Exercises**EXERCISE @ex:myLabel**

```

% All of the following "attributes" are optional.
% Attributes MUST be listed before code and text

% The current exercise can only be solved, if dependent
% exercises have been passed before
REQUIREMENT=ex:myEx1,ex:myEx2

% forbid shuffling of single/multiple choice answers
ORDER=static % default is shuffle

% display (short!) single/multiple choice answers horizontally
CHOICE_ALIGNMENT=horizontal % default is vertical

% forbid student to reanswer exercise
DISABLE_RETRY=true % default is false

% Limited time to solve the question [seconds].
TIME=4 % default is -1 := not timed

% total score weighting for this exercise. Default score per
% answer is 1. The sum of scores of all answers is weighted
% to the attribute given here.
SCORES=4 % default is 1

% The number of randomized variable instances for the
% exercise; default value is 5.
INSTANCES=3

% generates random variables and calculates the solution.
CODE
    let a/b = randZ(-3,3) % a,b in {-3,...,3}\{0}, a≠b
    let c = a + b % (assume: a=1, b=3) c=4
    let d = a > b % d=false
    let u = true
    let f(x) = x^2 + c % f(x) = x^2 + 4
    let g(x) = x^2 + term(c) % g(x) = x^2 + 1 + 3
    let h(x) = 2x + 5x % h(x) = 2x + 5x
    let y(x) = opt(h) % y(x) = 7x (opt := optimize)
    let w(x) = sin(2x) + 3x
    let wd(x) = diff(w,x) % wd(x) = 2cos(2x)+3
    let s = sqrt(-1)^2 % s = -1
    let v = [1,2,3,4] % vector with 4 elements
    let I = eye<3>() % A = [[1,0,0],[0,1,0],[0,0,1]]

Multiple Choice:
[x] correct answer
[ ] incorrect answer
% Dynamic answers. Only the name of a variable can be given

```

```
% here. Calculations must be done in part CODE.
[:d] correct iff $a > b$ is true
[:u] correct answer, given per variable
```

Single Choice:

```
(x) correct answer
( ) incorrect answer
```

```
% Asks "1 + 3 = [ ]" and provides a keyboard with
% integer numbers.
$a+b=$ #c
```

```
% Asks " $\int (x^2+4) dx = [ ] + C$  ( $C \in \mathbb{R}$ )" and provides a
% keyboard for terms.
% The student answer is first differentiated w.r.t.
% variable x, and then it is evaluated, i.e. compared to
% variable f.
$\int (f) \, , \, dx=$ #f,DIFF=x $+C (C\in\mathbb{R})$
```

```
% Asks "Garfield is a _ _ _.".
% The keyboard only shows letters {a,c,t} in shuffled order.
Garfield is a #"cat".
```

```
% Per default, gap answers give a hint on the number of
% characters needed. Also the keyboard is restricted to the
% letters that occur in the solution.
% Use the following attributes to change that behavior.
Garfield is a #"cat",HIDE_LENGTH,SHOW_ALL_LETTERS.
```

```
% Asks "Order the numbers ascendingly: [3] [2] [4] [1]"
% A correct answer is rewarded with 3 scores.
% → multiple attributes can be combined with "," as separator
Order the numbers ascendingly: #v,ARRANGE,SCORE=3
```

```
% Asks "The 3x3 identity matrix is [ ]".
% Per default, the student only needs to give the elements
% of the solution matrix. If attributes ROWS and/or
% COLS are set to dynamic, then the student also
% needs to find the matrix dimensions.
The 3x3 identity matrix is #I,ROWS=dynamic,COLS=dynamic
```

```
% Asks " $(\sqrt{-1})^2 = [ \ ]$ ". Since variable s is integral (-1), the
% default keyboard would only contain numeral keys.
% To "confuse" students, a keyboard with key "i" would be
% nice. Attribute KEYBOARD sets an existing1 or custom
% keyboard.
```

```
$term(s)=$ #s,KEYBOARD=myKeyboardName
```

```
% Asks "1 + 3 = [A][B][C]..." and provides one correct
% and (n-1) incorrect answers (e.g. {2,5,6,4,1} for CHOICES=5)
```

```
$a+b=$ #c,CHOICES=5
```

```
% Asks "1 + 3 = [A][B][C]..." and provides one correct
% and (n-1) incorrect answers (e.g. {10,5,6,4,20} for
% CHOICES=2 and manually added answers 10 and 20)
```

```
$a+b=$ #c,CHOICES=2+"10"+"20"
```

```
% Asks "3+4 = [7][1+6][2+3][4+3]". All answers are provided
% manually. Two answers are correct. Generally, at least one
% correct answer must be provided. If wished, also all
% provided answers may be correct and the student can click on
% any answer. If answers include variables, they are replaced,
% but the resulting term is not optimized (if wished, this has
% to be done in the part CODE)
% assume a=3, cc = 7
```

```
$3+4=$ #cc,CHOICES=0+"7"+"1+6"+"2+3"+"4+a"
```

```
% Asks "f(x)=sin(2x)+3x, f'(x) = [ ]" and shows (e.g.)
% tokens      [+]  [*]  [sin(2x)]  [3]  [2]  [cos(2x)]  [4]  [1]
% that must be selected to build the solution term,
% here: "f'(x)=2*sin(2*x)+3".
% Tokens are automatically derived from the solution term.
% The constant 1.5 is the factor of automatically generated
% tokens (based on the number of essential tokens). If this
% number is equal to 1, then each generated token is useful
% for the solution.
% Additional tokens (for student confusion) can be provided by
% adding +"MY_TOKEN_1"+"MY_TOKEN_2"... Tokens are in SMPL
% math format.
```

```
$"f"(x)=w, f'(x)=$ #wd,TOKENS=1.5+"pi"
```

¹ https://github.com/mathebuddy/mathebuddy/blob/main/app/lib/keyboard_layouts.dart

Keyboard definition

% keyboards must currently be inserted manually into the app code.

KEYBOARD myKeyboardName

% Keys are separated by one or more spaces.

%

% Special key "!B" is the backspace key.

% Special key "!E" is the enter key.

%

% A "submatrix" where each of the elements is the same

% is rendered as large key (e.g. "sqrt(" in the example spans

% two columns, keys "!B" and "!E" each span two rows).

7 8 9 + - !B

4 5 6 * / !B

1 2 3 ^ () !E

0 i pi sqrt(sqrt(!E

GAME Codename "Event"

(refer to slides in Sciebo)

```
event          % level name
#####
```

```
EXERCISE      % level contains n≥1 exercises
TIME=5
...
EXERCISE
...
```

GAME Codename : "Tetris" (... it's not Tetris)

Answers (or term-tokens) are falling down from top to bottom
 → the student must move them left or right (or keep middle)
 in limited time
 → speed accelerates...

Example

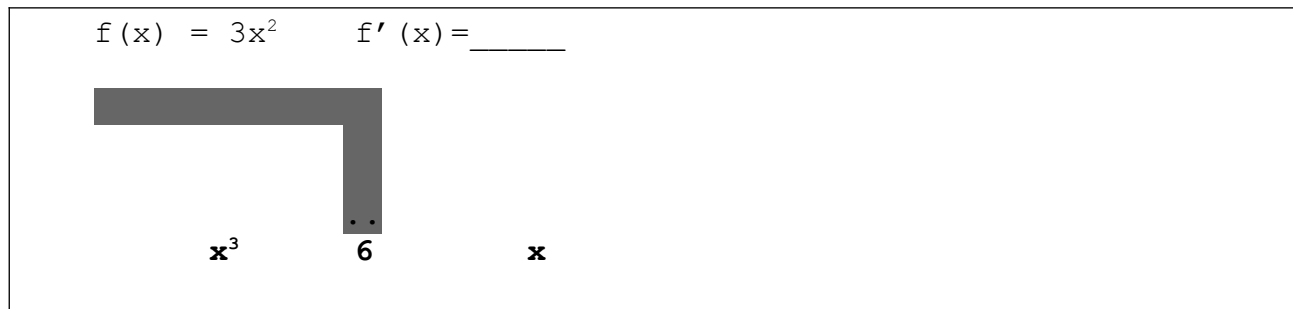
... $\cos(x)$ <i>next falling token</i> x <i>currently falling token</i>		
$f(x) = 3x^2$ $f'(x) = 6$	$f(x) = \sin(x)$ $f'(x) =$	$f(x) = \cos(x)$ $f'(x) =$

```
tetris        % level name
#####
```

```
EXERCISE      % level contains exactly 1 exercise
% shown columns (maybe two columns are the limit on
% smartphones with small displays...??)
COLUMNS=3
$ff(x)=ff$
$ff'(x)=$ #ffd,TOKENS
% code instances; each instance must provide all referenced
% variables
CODE
  let a/b = rand(2,7)
  let ff(x) = a x^b           % "*" can be omitted :-)
  let ffd(x) = diff(ff)
CODE
  let ff(x) = sin(x)
  let ffd(x) = diff(ff)
CODE
  let ff(x) = cos(x)
  let ffd(x) = diff(ff)
...
```

GAME Codename: "Snake"

The answer (or tokens) are randomly placed in a 2D-grid. A snake, that keeps moving into a direction until changed, must be controlled towards the next correct solution. If the solution is "eaten", the snake gets larger. Speed accelerates...

**SMPL - Loops & Conditions**

```

let x = 5
% execute block {...}, while x > 0 is true
while x > 0 {
    x = x - 1
}

% execute block once and repeat loop as long as x < 5 is true
do {
    x = x + 1
} while x < 5

let f = 1
% set (new) variable k to 1, 2, 3, 4, 5 and run the block each
% iteration
for k from 1 to 5 {    % bounds can also be terms
    f = f * k
}

x = 4
let s = 0                % s := sign(x)
if x > 0 {
    s = 1
} elif s < 0 {           % elif := else if
    s = -1
} else {
    s = 0                % redundant here, since s is already 0
}

```

M A T H - R U N T I M E :

OPERAND-TYPE	EXAMPLE (s)	
boolean	true, false	
int	-3, 4	
rational	4/7	
real	3.14, -1.337	
irrational	pi, e	
complex	4+5i	
vector	[1,2,3+4i]	
matrix	[[1,2],[3,4],[5,6]]	(3x2 matrix)
set	{ 3, 4, 4/3 }	
identifier	blub	
string	"hello, world!"	

List of nullary functions (ups; dimensions are some kind of non-null-arity):

eye<3>()	identity matrix with 3 rows and 3 columns
zeros<3>()	zero vector with 3 elements
zeros<3,4>()	zero matrix with 3 rows and 4 columns
ones<3>()	one's vector with 3 elements
ones<3,4>()	one's matrix with 3 rows and 4 columns

List of unary functions: **TODO: acos, asin, ...**

abs(x)	absolute value of x (int, vector, cmplx, ...)
arg(x)	argument of complex x
ceil(x)	smallest integer that is $\geq x$
cols(x)	number of columns of matrix x
conj(x)	conjugate complex number of x
cos(x)	cosine of x
det(x)	determinant of matrix x
exp(x)	exponential function of x
fac(x)	factorial of x
floor(x)	smallest integer x that is $\leq x$
imag(x)	imaginary part of complex x
is_invertible(x)	true, if mat x is invertible, otherwise false
is_symmetric(x)	true, if mat x is symmetric, otherwise false
is_zero(x)	true, if (value,vec,mat,...) x is approx zero
len(x)	number of elements of set or vector
ln(x)	natural logarithm
max(x)	maximum element of vector, matrix, set, ...
min(x)	minimum element of vector, matrix, setm, ...
norm(x)	euclidean norm of vector x
opt(x)	optimize term x (e.g. $2x+3x \rightarrow 5x$)
real(x)	real part of complex x
round(x)	round to nearest integer
rows(x)	number of rows of a matrix
shuffle(x)	shuffles the elements of a vector
sin(x)	sine
sqrt(x)	square root
tan(x)	tan
term(x)	term of variable x (refer to exercise example)
transpose(x)	transpose matrix x
triu(x)	upper triangular part of matrix x

List of binary functions:

binomial(n,k)	binomial coefficient
complex(x,y)	returns a complex number $x+yi$
col(a,k)	gets column k (first is 0) from matrix a
cross(x,y)	cross product of vectors x and y
diff(f,x)	automatic differentiation of function f w.r.t variable x
dot(x,y)	dot (scalar) product of vectors x and y
rand(x,y)	draws a random number in range [x,y]

rand<n>(x,y)	draws a random vector with n elements ...
rand<m,n>(x,y)	draws a random matrix with m rows and n cols ...
randZ(x,y)	same as rand(x,y) without zero
randZ<n>(x,y)	same as rand<n>(x,y) without zeros
randZ<m,n>(x,y)	same as rand<m,n>(x,y) without zeros
row(a,k)	gets row k (first is 0) from matrix a

List of quaternary functions:

int(f,x,a,b)	Numerically approximates integral of function f w.r.t. variable x in bounds from a to b
---------------------	--