First Summary of IDL

- **First (today)**
  1 Demo
  1.1 IDL Overview
  2 Getting startet with the Basics, IDL System
  3 Analyzing and Working with Data

- **Next**

4 Displaying Data as Line Plots

5 Displaying Surface and Contour Plots

6 Displaying Image Data

- **Last**

7 Gridding Random or Irregular Data

8 Volumen Visualization

9 Applications (DLR)

2. Getting Started with the Basics

- IDL's command syntax:
  COMMAND, (opt) parameters, KEYWORD1=val,...
  parameters must occur in specific order
  IDL is not case sensitive!

- keyword shortcuts:
  PLOT, line, CHARS=3, LINES=2
  ;instead of CHARSIZE=3, LINESTYLE=2
  ;/KEYWORD equal to KEYWORD=1

- special characters:

' ; '	first character of a comment
' @ '	for batch execution
' $ '	indicates a continuation line: PLOT, x, y, $ CHARSIZE=2
'&'	multiple statements in a line: x=FINDGEN(20) & PLOT,x

- Getting help

HELP, data ; help about variable data
HELP, data, OUTPUT=var ;var contains HELP's output
HELP,/MEMORY ; dyn. memory currently in use
HELP,/ROUTINES ; compiled routines

? ; online hypertext help

- Save your data or the routines in the IDL session

SAVE, /ALL ;save all local variables incl. contents
;default file: "idlsave.dat"
SAVE, /ROUTINES ;save all currently compiled IDL
;routines in a platform independent
;binary file
SAVE, x_vec, y_vec, FIlENAME='creaso.sav'
;save "x_vec"+"y_vec" in file creaso.sav

- Restore saved variables or routines

RESTORE, 'CREASO.SAV' ;on different platforms...

- IDL System variables (identified by an " ! " -mark) predefined by IDL
  HELP, /SYSTEM_VARIABIES

- Examples
  PRINT, !PI & PRINT,!DPI & PRINT,!VERSION

- IDL environment:

  PRINT, !DIR ; location of the IDL main directory
  PRINT,!PATH ; list of directories to search für
  ; (user-written) procedures

- ***IDLDE***

  File->Preferences, menu's to set interactiv:
  - *IDL path*
  - *Startup file,* executed in batch fashion
    ; when IDL is started
  - Preferences für editor, graphics

3. Analyzing and Working with Data:

- IDL Variables:
- Names can incl. characters '_', $, 0...9
  (1. character a letter !)
- AII IDL Names are case insensitive
- Size: limited only by memory, not by IDL !

- Variable Types and initializing examples:

	scalars:	arrays:
byte (8 bit)	v=0b	a=BYTARR(10,20)
signed int (16 bit)	v=0; v=0s	a=INTARR(12,...,20)
unsigned int (16 bit)	v=0u; v=0us	a=UINTARR(10,20)
signed long int(32bit)	v=0l	a=LONARR(SO,100)
unsigned long int(32bit)	v=0ul	a=ULONARR(10.20)
signed 64-bit long int	v=0ll	a=ION64ARR(10,S)
unsigned 64-bit long	v=0ull	a=ULON64ARR(S.S)
float	v=0.0	a=FITARR(4000)
double	v=0.0d	a=DBLARR(300, ...)
complex	v=COMPLEX(0,0)	a=COMPLEXARR(..)
double complex	v=DCOMPLEX(1,0)	a=DCOMPLEXARR(..)
string	v= 'Hallo'	a=STRARR(10)

- Variable types and value range:

Type conversion	Range
v1 =BYTE(v)	0 to 255
v1=FIX(v)	-32,768 to +32,767
v1 =UINT(v)	0 to 65,535
v1 =LONG(v)	-2,147,483,648 to +2,147,483,647
v1 =LONG64(v)	-2^63-1 to 2^63-1
v2=FLOAT(v)	±10^38 (6-7 decimals significance)
v2=DOUBLE(v)	±10^308 (-14 dec. sign.)

- Heap variables with **IDL 5**

  objects:
  obj = obj_new('class') or *aobj-objarr(10,* ..)

  pointers:
  ptr = ptr_new(init) or *aptr-ptrarr(20. 2*, ..)

- Variable type and structure (array, scalar...) change dynamically:

```
a=FLTARR(20,10)
a=a+1
HELP,a ; get the type, dimensions of 'a'
a=1
HELP,a
```

- Working with Arrays (up to 8 dimensions)
    - Array creation and initialization examples:

```
arr1=[2,8,26,3]                ; by hand
arr2=BYTARR(50, 100)           ;or INTARR( ...), FITARR(...,..
                               ; all elements=0
arr3=BINDGEN(256)              ;or INDGEN( ...),FINDGEN(...)
                               ;elements of arr3=BYTE(index)
```

- Subsetting N-dimensional arrays with 1 D subscripts:

```
arr4=BINDGEN(256,500)     ;create a 2D array
TV, arr4                  ;show array (color bar)
PRINT, arr4[24, 2]        ;
PRINT, arr4[536]          ;
```

- Selecting subarrays with ' * ' and ' : '

' * ' all elements of a column or row:

```
PLOT .arr4[*,17]     ;all (column-) elements of row 17
```

' : ' selecting an arbitrary subscript range, e.g.:

```
arr4[80:220.200:252]=0        ;set all elements of the

TV,arr4                       ; range to 0
```

- Using arrays as indices to other arrays

```
one=[6,5,1,8,4,3]
two=[0,2,4,1]
PRINT, one[two]          ; print one[0]. one[2], ...
```

- Using expressions tor subscripting: WHERE-function:

```
i_vec = WHERE( arr4 LE 100 AND arr4 GE 20)
arr4[i_vec]=255
TV, arr4
```

IDL Operators

- All IDL operators can be used with arrays:

  arr4=arr4 - 100

  arr4=arr4+ (2 * arr4)

- Boolean Operators

  AND, OR, NOT, XOR

- Relational Operators (result 0 (false) or 1 (true))

  GT **G**reater **T**han

  LE **L**ess **E**qual

  EQ **E**qual

  ....

- Minimum '<' and Maximum '>' Operator:

```
x=[5,10,20] >[6,9,30]
PRINT,x
```

- Array concatenation with [...]:

```
v1=[2,3,1]
v2=[2,1,4]
v3=[v1,v2]          ;concatenate v1 with v2
PRINT,v3
```

- Structure variables:
    - 1. Definition of a ***named structure:***

var={*s_name, s_tag* 1: *definition, s_tag2:definition,* ...}

    - 2. Definition of a ***anonymous structure*:**

var={*s_tag1: definition, s_tag2:definition,* ... }

- Example, use of a named structure:

```
var1 = {my_data, $
        date:' ',$          ; date of the measurement
        para:0.0, $         ; measurement parameter
        image: bytarr(256.256)}
var1.date='15.10.2002' ; access a field
```

- duplicate a named structure

```
var2={my_data}          ;copy only the tags
var2=varl               ;copy all contents
```

- Example: use of a vector of structures
  (e.g. to group all datasets together)

```
all_data = REPLICATE({my_data},5) ;arr. of 5 struct.
all_data[0] = var1                ;set element 1
all_data = [all_data, var2]       ;add an additional
                                  ; structure as element 6
all_data[J].date = '20.04.2002'   ;set value of tag
                                  ; 'date' of vector element 4
dates = all_data.date             ;copy the dates of all
                                  ;structures to variable dates
HELP, dates                       ; .....
```