-- Lluís Parcerisa --

C++ as a programming language for CAS

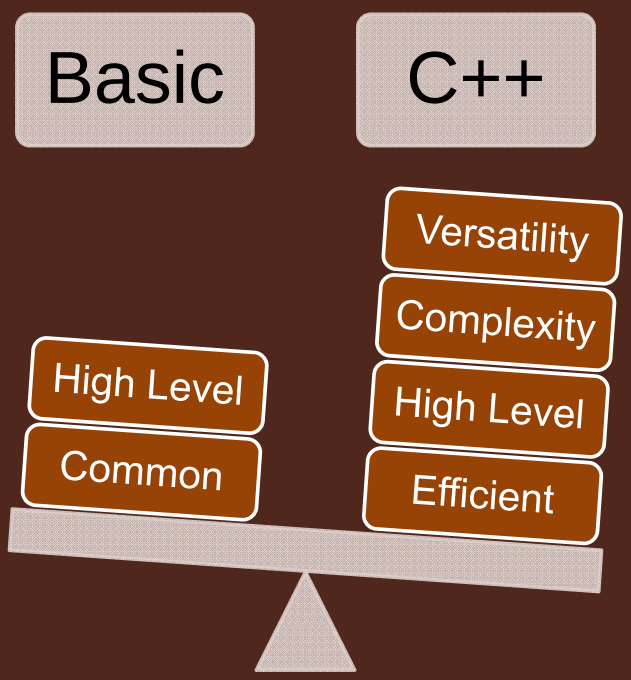
Contents

1.- Introduction	• C++ vs Basic • C++ in a calculator
2.- Application structure	• Development environment • Application model • Example
3.- Functions	• Embedded functions • CAS calling
4.- Real application examples	• Laboratory • Result checking • Circuit Analysis
5.- Conclusions	• C++ as future programming language bridging CAS • Advantages and Drawbacks • Final statement

Contents

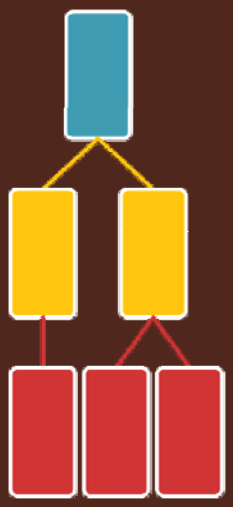
1.- Introduction	• C++ vs Basic • C++ in a calculator
2.- Application structure	• Development environment • Application model • Example
3.- Functions	• Embedded functions • CAS calling
4.- Real application examples	• Laboratory • Result checking • Circuit Analysis
5.- Conclusions	• C++ as future programming language bridging CAS • Advantages and Drawbacks • Final statement

C++ vs Basic - Features



C++ vs Basic – App. Structure

- C++ => Object oriented
  - Code reuse and extension
  - Adequate for complex systems
  - Empower visual applications
  - Prototype construction
  - Speed application programming time
  - Enforce cooperative work
  - Software maintenance



C++ to program calculator apps

Programmer

- Fast learning
- Code division
- Embedded functions
- CAS calling

User

- Easy
- Intuitive
- Useful
- Customizable

Contents

1.- Introduction
- C++ vs Basic
- C++ in a calculator

2.- Application structure
- Development environment
- Application model
- Example

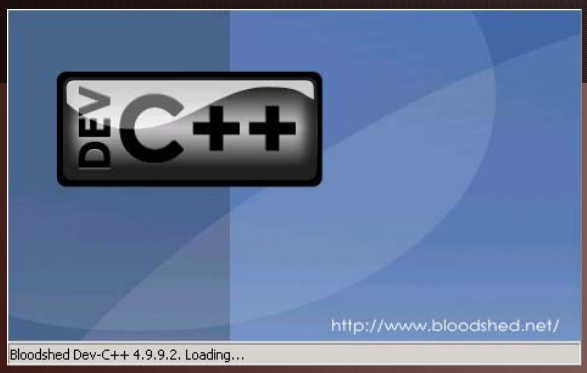
3.- Functions
- Embedded functions
- CAS calling

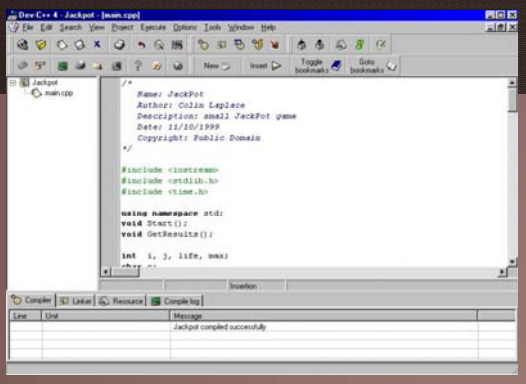
4.- Real application examples
- Laboratory
- Result checking
- Circuit Analysis

5.- Conclusions
- C++ as future programming language bridging CAS
- Advantages and Drawbacks
- Final statement

Development Environment

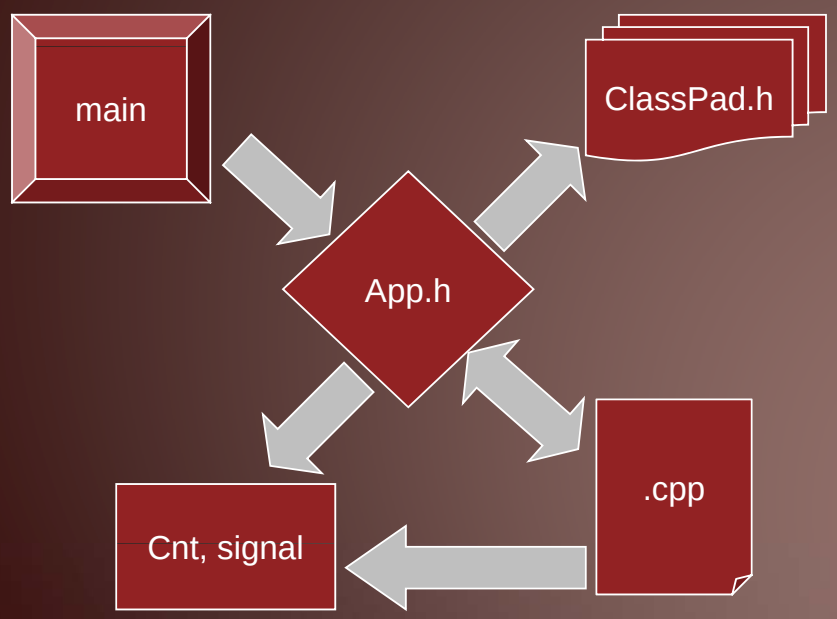
- C Sintax
- Compilation: add-in cpa
- Software: Dev-CPP
- www.cpsdk.com

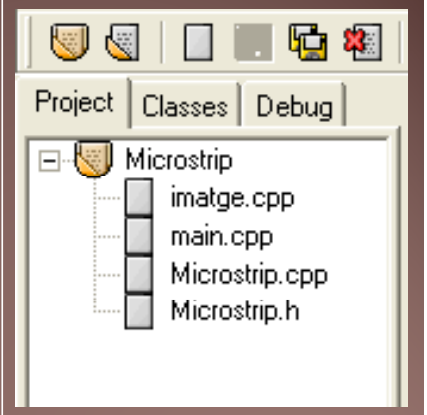




Application Structure

- Model: A. Pizdrowsky / B. White





Implementation example

main.cpp

```
#include "Microstrip.h"
// The main method
void PegAppInitialize(PegPresentationManager *pPresentation)
{
     Microstrip *swin; //create the program as an object
     PegRect Rect; // create a rectangle object for the MainFrame
     Rect.Set(MAINFRAME_LEFT, MAINFRAME_TOP, MAINFRAME_RIGHT,
     MAINFRAME_BOTTOM); // set the rectangle to the usable size of the display
     CPMainFrame *mw = new CPMainFrame(Rect); // create the MainFrame
     PegRect AppRect; // create a rectangle for the Application Window
     AppRect = mw->FullAppRectangle(); // size it to the MainFrame
     swin = new Microstrip(AppRect,mw); // create the program as an Application Window
     mw -> SetTopWindow(swin); // load the application window into the MainFrame
     mw -> SetMainWindow(swin); // set a main window for this module
     pPresentation->Add(mw); // add the MainFrame to the Presentation Manager
}
```

Implementation example

Microstrip.h

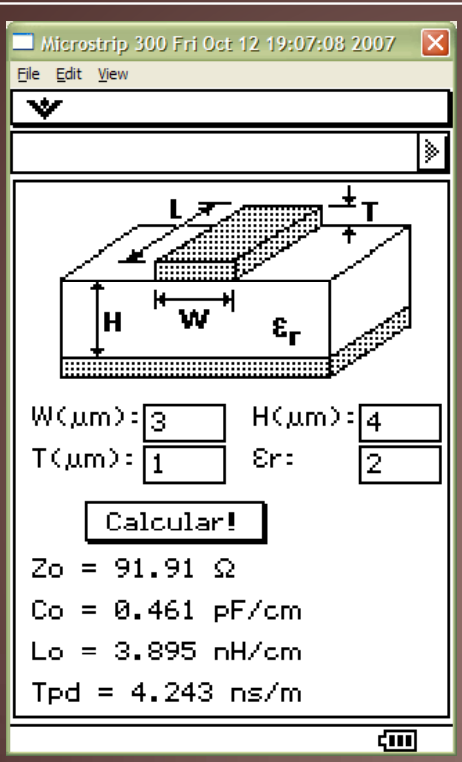
```
#include "ClassPad.h"
#define BOTO 10

class Microstrip: public CPModuleWindow
{
protected:
CPPegString* T;
CPPegString* W;
CPPegString* H;
CPPegString* er;
PegPrompt* res1;
PegPrompt* res2;
PegPrompt* res3;
PegPrompt* res4;
OBCD Zo;
OBCD Co;
OBCD Lo;
OBCD Tpd;

public: Microstrip(PegRect, CPMainFrame*);
~Microstrip(){}
int init();
void Draw();
SIGNED Message(const PegMessage &Mesg);
WORD Dialog(char*, char*);
void Calcula();

};
```


Implementation example

Microstrip.cpp

```
#include "Microstrip.h"

extern PegBitmap gbimatgeBitmap;

Microstrip::Microstrip(PegRect rect, CPMainFrame* frame):CPModuleWindow(rect,0,0,frame)
{
init();
}

int Microstrip::init()
{
    const WORD wStyle = FF_NONE|TJ_LEFT|AF_TRANSPARENT| TT_COPY;
    const SIGNED margin = 5;
    const SIGNED width=mClient.Width()-2*margin;

    PEGCHAR* chrepsilon = "\xEC\xA0";
    PEGCHAR* chrmu = "\xED\x72";

    PegRect rectangle;

    PegPrompt* label5 = new PegPrompt(margin, 80, width/2, "W("+CPString(chrmu)+"m):",0,wStyle);
    W = new CPPegString(margin+41,80,30,"",0, FF_THIN|AF_ENABLED|EF_EDIT);
    AddR(W);
    AddR(label5);


    PegPrompt* label = new PegPrompt(margin, 95, width/2, "T("+CPString(chrmu)+"m):",0,wStyle);
    T = new CPPegString(margin+41,95,30,"",0, FF_THIN|AF_ENABLED|EF_EDIT);
    AddR(T);
    AddR(label);
```

Implementation example



Contents

1.- Introduction
- C++ vs Basic
- C++ in a calculator

2.- Application structure
- Development environment
- Application model
- Example

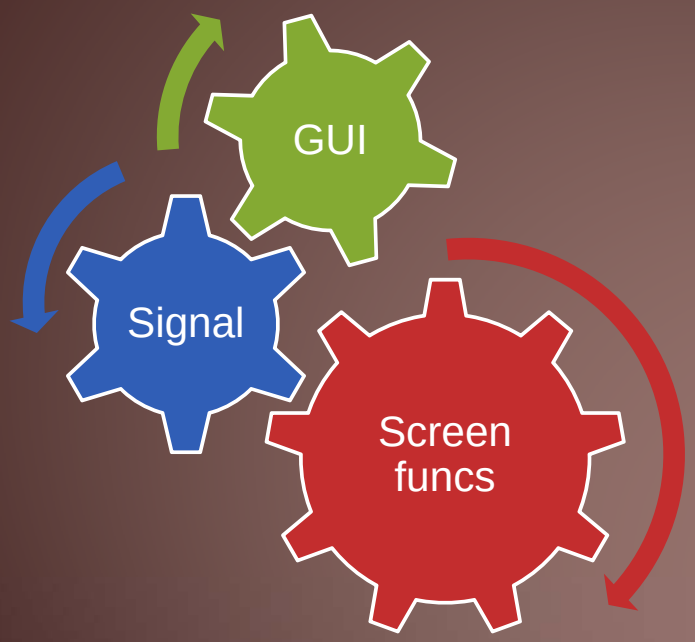
3.- Functions
- Embedded functions
- CAS calling

4.- Real application examples
- Laboratory
- Result checking
- Circuit Analysis

5.- Conclusions
- C++ as future programming language bridging CAS
- Advantages and Drawbacks
- Final statement

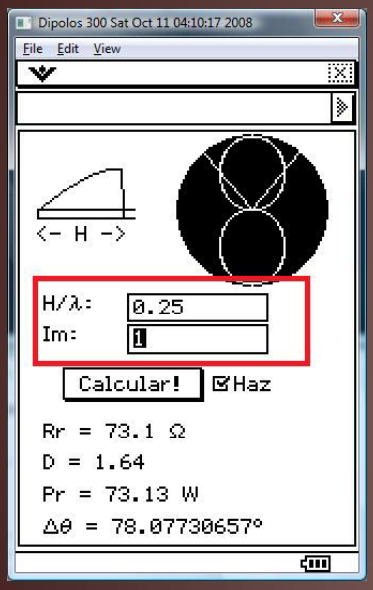
FLAMAGAS División didáctica www.aulaCASIO.com 20

Embedded Functions

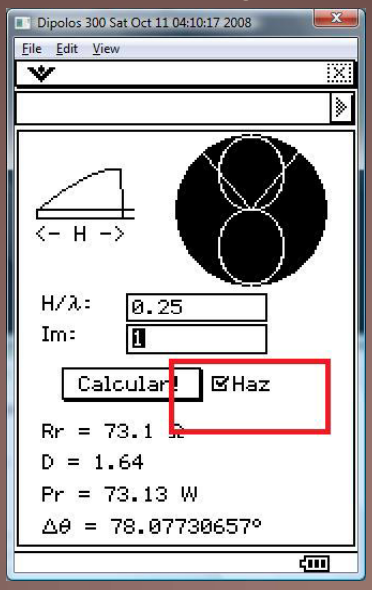


GUI Elements

PegPrompt, CPPegString

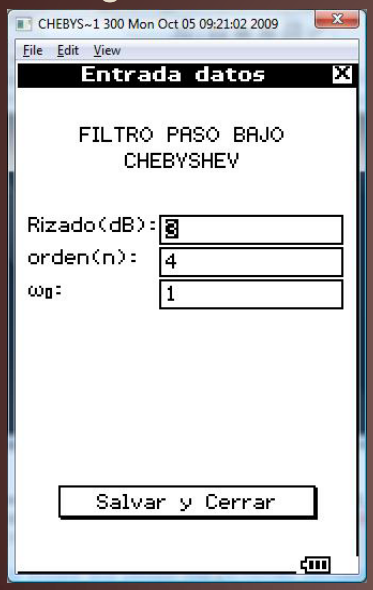


PegCheckBox, PegTextButton

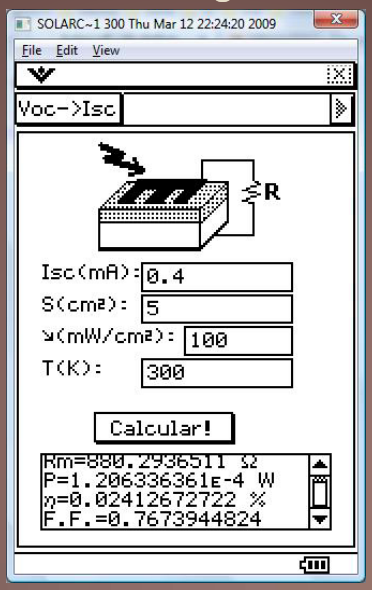


GUI Elements

CPDialog,
PegMessageWindow



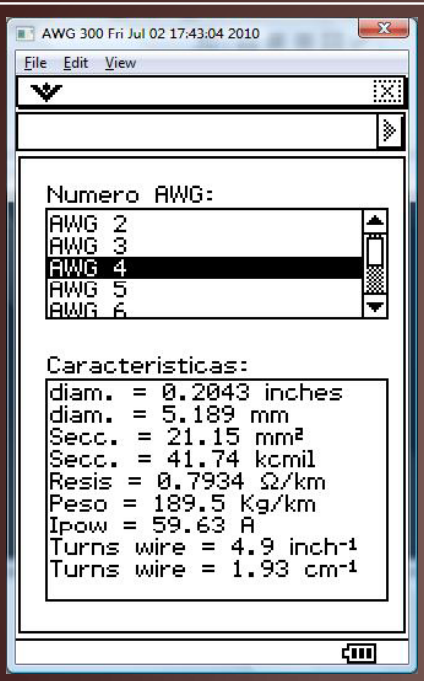
PegVertList, PegComboBox

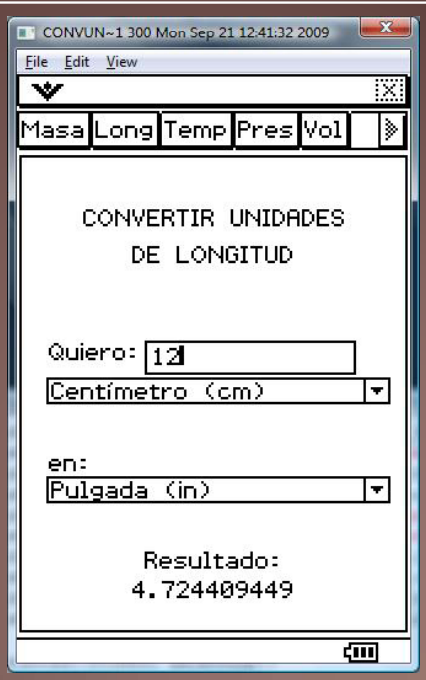


Signals

```
pList->SetSignals(SIGMASK(PSF_LIST_SELECT));

(...)

SIGNED AWG::Message(const PegMessage &Mesg) {
    switch(Mesg.wType) {
        case SIGNAL(ID_VERTLIST,PSF_LIST_SELECT):  {
            cList->Clear();
            Calcula(pList->GetSelectedIndex()-3);
            Draw();
     } break;
        default: return CPModuleWindow::Message(Mesg);
    }
    return 0;
}
```

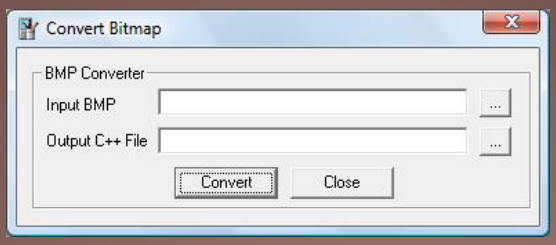
Ex: AWG, Conversor



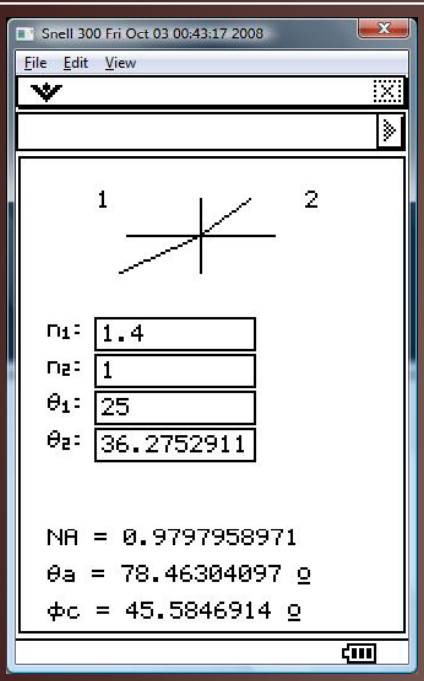


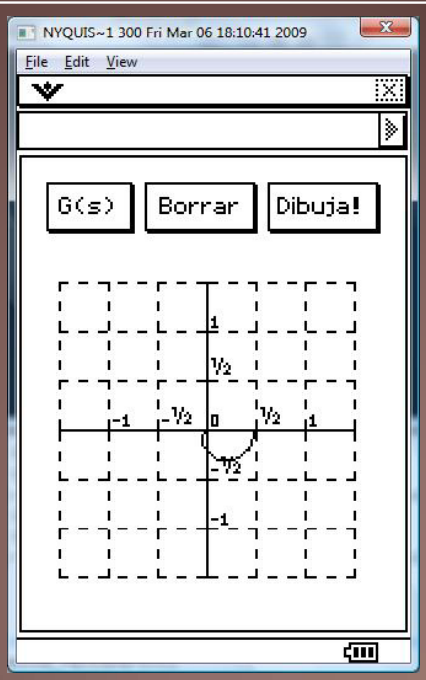
Graphical Functions

- PegScreen
  - Line
  - Rectangle
  - Circle
  - PatternLine
  - Arc
  - Polygon
- Convert Bitmap

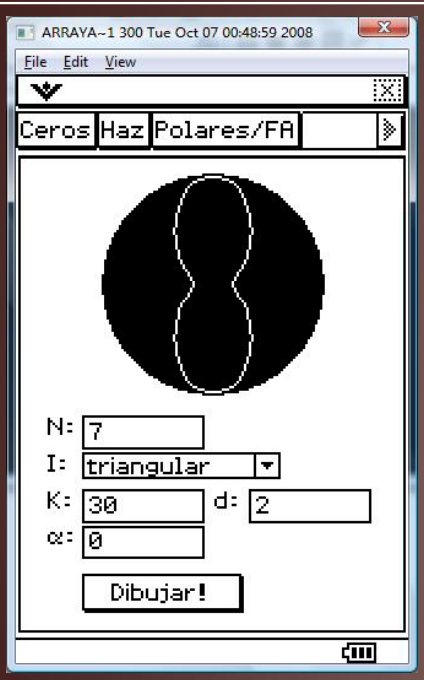


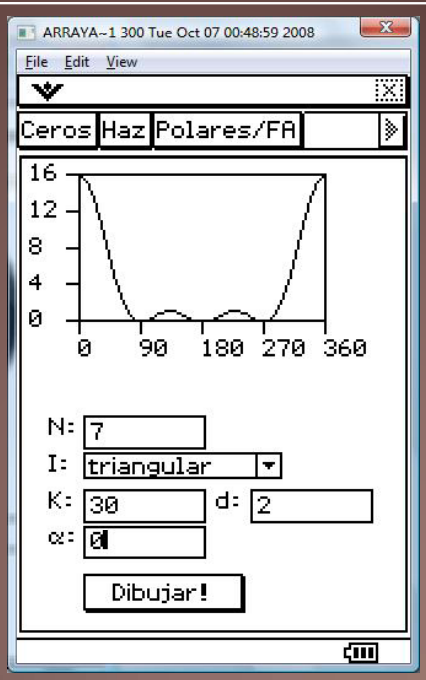
Ex1: Snell, Nyquist Diagram





Ex2: Antenna Arrays





Real numbers

OBCD Data Structure

The structure of an OBCD is defined as:

```
typedef struct obcd_ {
      unsigned char mantissa[IM_CAL_INDIGIT];
      unsigned short exponential;
} OBCD_;

typedef union obcd {
      OBCD_ obcd1;
      unsigned long      dummy[3];
} OBCD;
```

The mantissa of a number is stored in obcd1.mantissa. The mantissa is 10 bytes long, with the least significant 2 bytes reserved for system use. The most significant nibble is reserved for a flag. There is also a 2 byte exponent that is stored in a short. The entire structure looks like this:

eF	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	eS	e1	e2	e3
Flag	Mantissa															reserved				Exponent			

Real numbers

Functions

word **Cal_decfrac_OBC** (OBCD *x, OBCD *y, OBCD *z)
Converts a decimal to a fraction.

word **Cal_add_OBC** (OBCD *x, OBCD *y)
Adds x to y (x+y).

word **Cal_sub_OBC** (OBCD *x, OBCD *y)
Subtracts x from y (x-y).

word **Cal_mul_OBC** (OBCD *x, OBCD *y)
*Multiplies x and y (x*y).*

word **Cal_div_OBC** (OBCD *x, OBCD *y)
Divides y into x (x/y).

word **Cal_pow_OBC** (OBCD *x, OBCD *y)
Raises x to y (x^y).

word **Cal_relat** (OBCD *x, OBCD *y)
Compares x to y to determine an inequality relationship.

word **Cal_relat0** (OBCD *x)
Compares x to 0.

word **Cal_relateq** (OBCD *x, OBCD *y)
Compares x to y and returns equal or not equal.

Contents

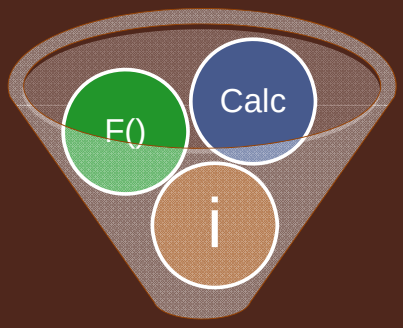
1.- Introduction	• C++ vs Basic • C++ in a calculator
2.- Application structure	• Development environment • Application model • Example
3.- Functions	• Embedded functions • CAS calling
4.- Real application examples	• Laboratory • Result checking • Circuit Analysis
5.- Conclusions	• C++ as future programming language bridging CAS • Advantages and Drawbacks • Final statement

Calling CAS Functions

- **CPExpression Class Reference**
  CPP expression class which manages expression memory. The CPExpression class contains a reference to a CPExpressionObject. A CPExpressionObject should never be used directly. This is a reference counted object that keeps track of the memory used for the CAS expression. When there are no longer any CPExpression objects referring to this memory it is automatically freed.

FLAMAGAS División didáctica www.aulaCASIO.com 20

Calling CAS Functions



CPExpression(x)

CAS → C++

Calling CAS Functions

- Transform
- Calculate
- Complexes
- Matrices
- Equations

Fraction	Approx / toFrac / propFrac	3/2 => 1.5 1.5 => 3/2 1.5 => 1 1/2
Polinomy	Expand / Collect Factor / factorOut rFactor Combine Simplify	(a+b)^2 => … Factorize (x) Root mcd
Trigonometry	tExpand / tCollect expToTrig / trigToExp	sin(a+b) e^jx
Other	Dms / toDMS	3.44 => 3,26,24

Calling CAS Functions

- Transform
- Calculate
- Complexes
- Matrices
- Equations

Taylor

```
taylor(sin(x),x,5,0)
```

$$\frac{x^5}{120}-\frac{x^3}{6}+x$$

Laplace / InvLaplace

$$L[f(t)](s)=\int_0^\infty f(t)e^{-st}dt$$

```
laplace(x'+2x=e^-t,t,x,s)
```

$$-x(0)+Lp\cdot s+2\cdot Lp=\frac{1}{s+1}$$

```
ans|x(0)=3
```

$$Lp\cdot s+2\cdot Lp-3=\frac{1}{s+1}$$

```
solve(ans,Lp)
```

$$\left\{Lp=\frac{3\cdot s}{s^2+3\cdot s+2}+\frac{4}{s^2+3\cdot s+2}\right\}$$

```
invlaplace(getright(ans[1]),s,t)
```

$$e^{-t}+2\cdot e^{-2\cdot t}$$

+ Fourier/InvFourier/FFT…

Calling CAS Functions

- Transform
- Calculate
- Complexes
- Matrices
- Equations

diff / ∫

lim

Σ / Π

fMin / fMax

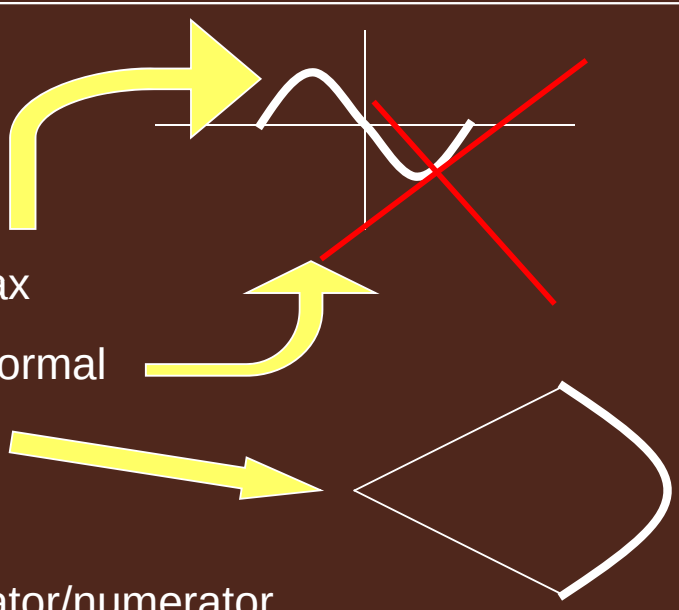
tanLine/normal

arcLine

gcd / lcm

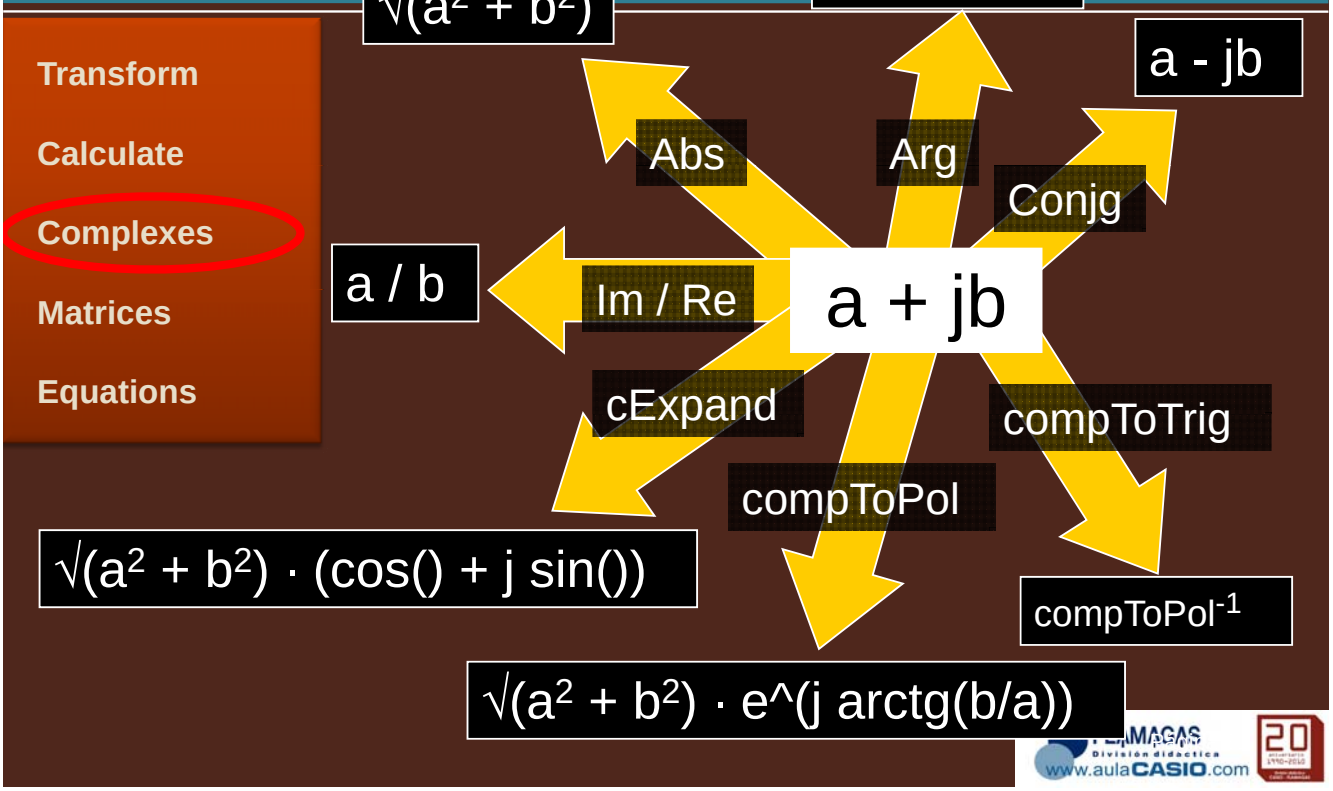
denominator/numerator

mod



Calling CAS Functions

- Transform
- Calculate
- Complexes
- Matrices
- Equations



Calling CAS Functions

- Transform
- Calculate
- Complexes
- Matrices
- Equations

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

- Trn => [a c][b d]
- Dim => {2,2}
- Det => ad - bc
- Norm => √(|a|^2+|b|^2 +|c|^2 +|d|^2)
- eigVl / eigVc => propis
- Rref / Ref => escalonada

escomposició

][a b]

Creation:
- Menú
- Vectors
- 2D

Calling CAS Functions

- Transform
- Calculate
- Complexes
- Matrices
- Equations

Reals → `Solve(3x-8=0)`

Parameter → `Solve(mx^2+x-5=0)`

Complexes → `Solve(y^2-1=0,y)`

Inequalities → `Solve(x^2≠5)`

`Solve(x-5>2)` ← Inequalities

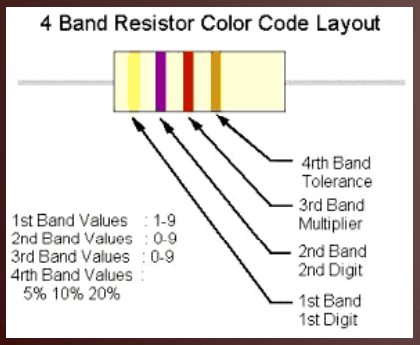
Systems → `Solve({x+y=0,x-y=7},{x,y})`

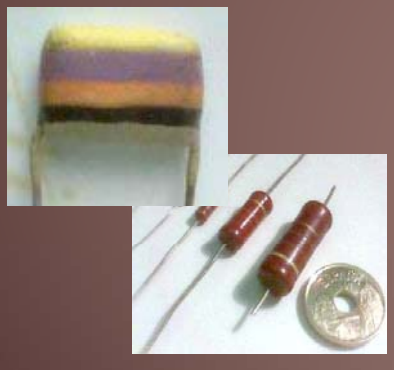
Contents

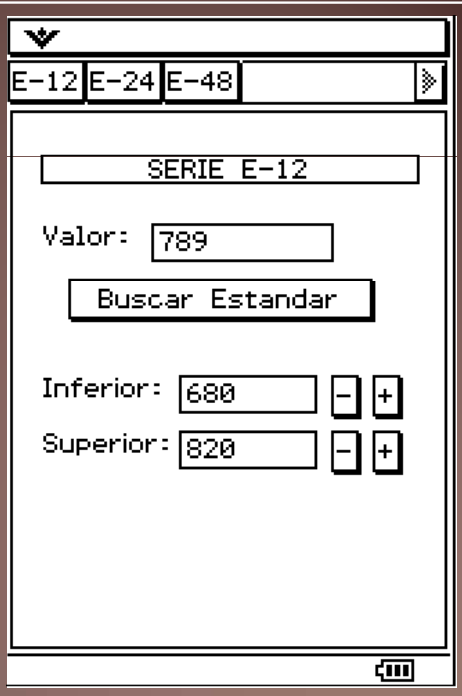
1.- Introduction	• C++ vs Basic • C++ in a calculator
2.- Application structure	• Development environment • Application model • Example
3.- Functions	• Embedded functions • CAS calling
4.- Real application examples	• Laboratory • Result checking • Circuit Analysis
5.- Conclusions	• C++ as future programming language bridging CAS • Advantages and Drawbacks • Final statement

Lab - Standard

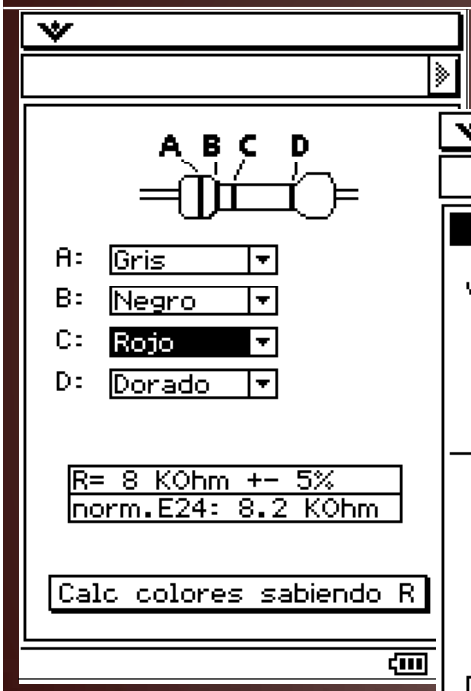
E6	1.0		1.5		2.2		3.3		4.7		6.8	
E12	1.0	1.2	1.5	1.8	2.2	2.7	3.3	3.9	4.7	5.6	6.8	8.2
E24	1.0	1.1	1.2	1.3	1.5	1.6	1.8	2.0	2.2	2.4	2.7	3.0
	3.3	3.6	3.9	4.3	4.7	5.1	5.6	6.2	6.8	7.5	8.2	9.1
E48	1.0	1.05	1.10	1.15	1.21	1.27	1.33	1.40	1.47	1.54	1.62	1.69
	1.78	1.87	1.96	2.05	2.15	2.26	2.37	2.49	2.61	2.74	2.87	3.01
	3.16	3.32	3.48	3.65	3.83	4.02	4.22	4.42	4.64	4.87	5.11	5.36
	5.62	5.90	6.19	6.49	6.81	7.15	7.50	7.87	8.25	8.66	9.09	9.53
Tolerancias de las series : E6 20% - E12 10% - E24 5% - E48 2%												
Valores de las resistencias en , K , M IEC = Comisión eléctrica Internacional												

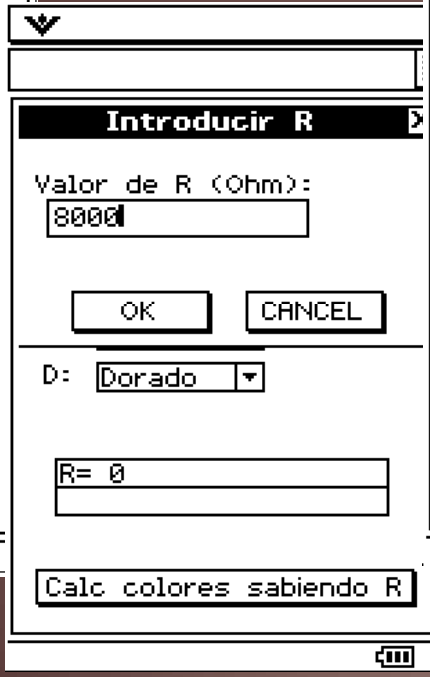


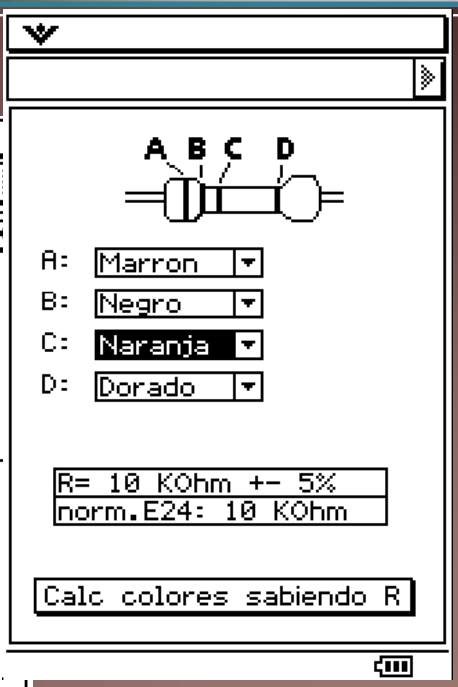




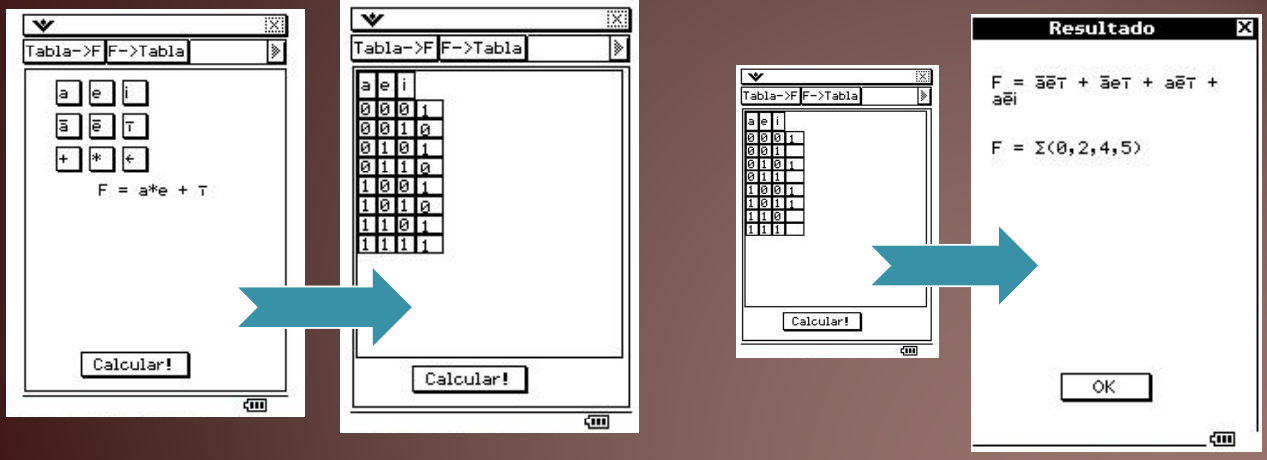
Lab - Resis







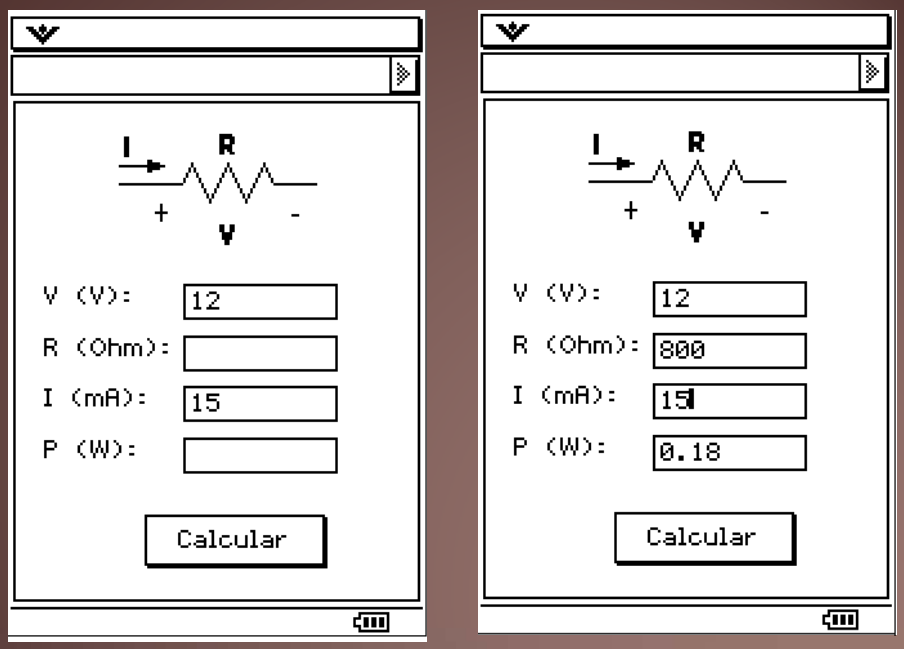
Lab – Logic Functions



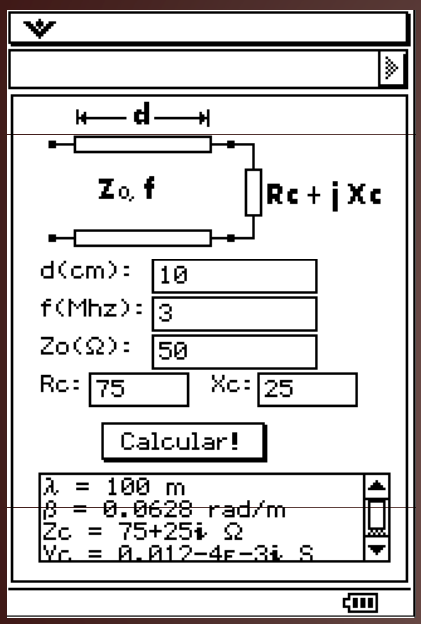
Check – Ohm's Law

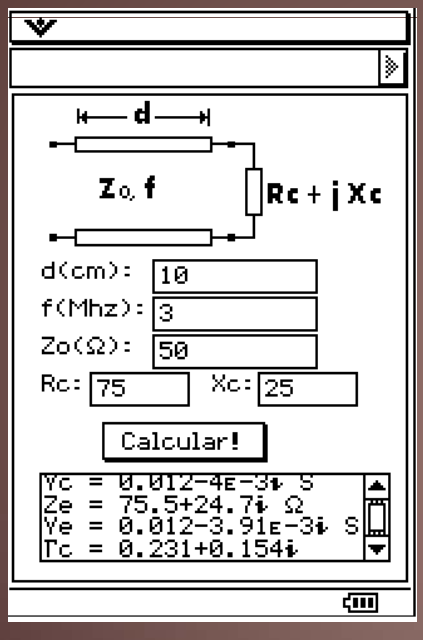
$$V = R \cdot I$$

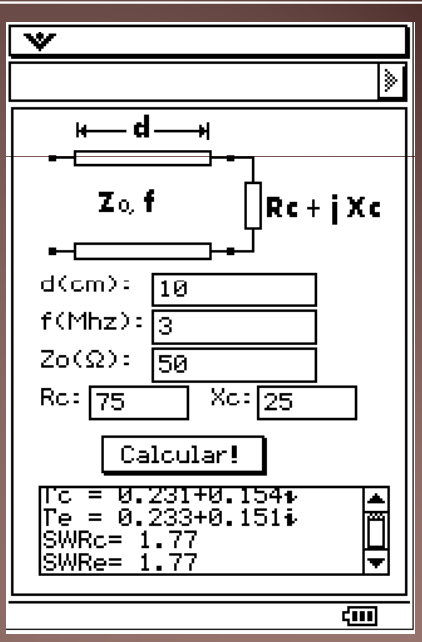
$$P = V \cdot I$$



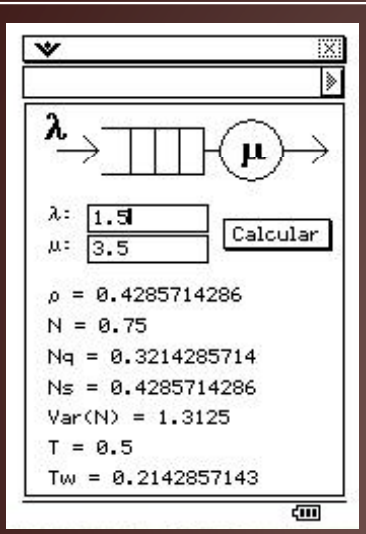
Check – Tx. Lines

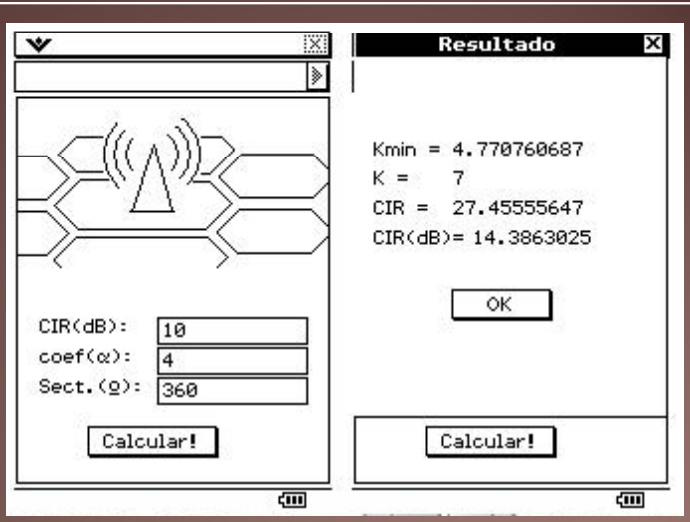






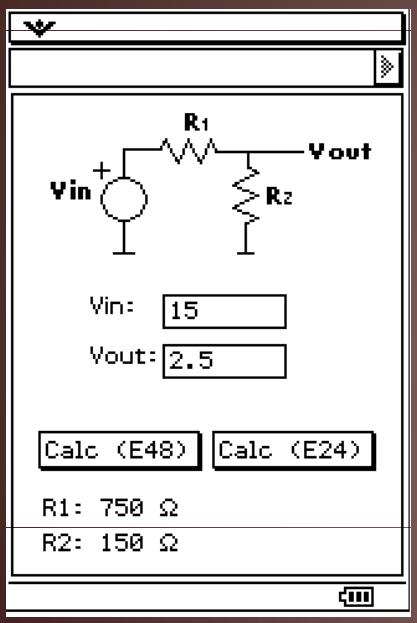
Check - Communications

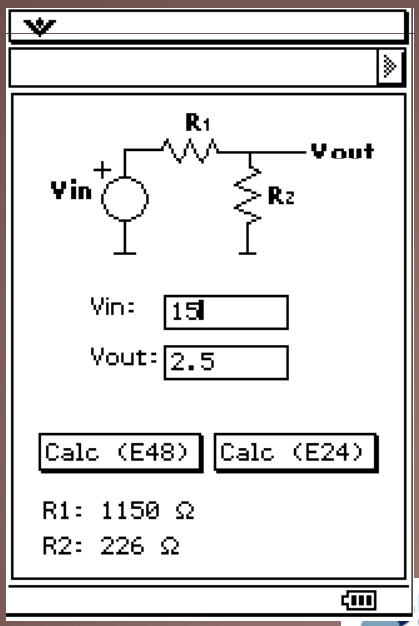




Circ. Analysis - Vdiv

Vo = R2/(R1+R2) · Vi

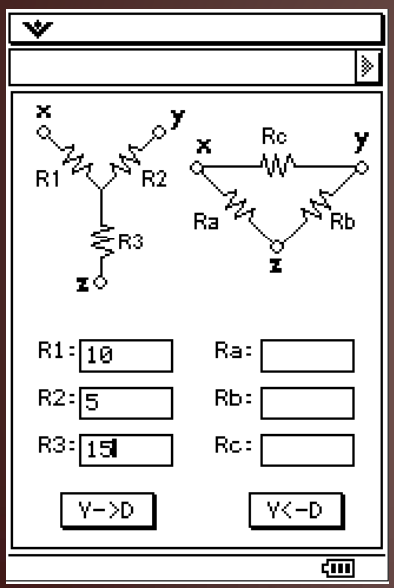


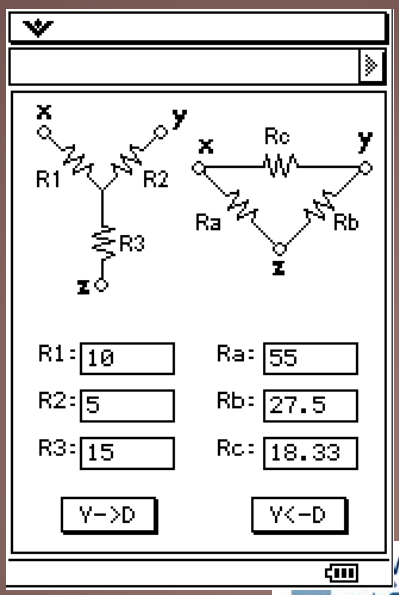


Circ. Analysis - D-Star

R1 = (Ra x Rc) / (Ra + Rb + Rc)
R2 = (Rb x Rc) / (Ra + Rb + Rc)
R3 = (Ra x Rb) / (Ra + Rb + Rc)

Ra = [(R1 x R2) + (R1 x R3) + (R2 x R3)] / R2
Rb = [(R1 x R2) + (R1 x R3) + (R2 x R3)] / R1
Rc = [(R1 x R2) + (R1 x R3) + (R2 x R3)] / R3





Contents

1.- Introduction	• C++ vs Basic • C++ in a calculator
2.- Application structure	• Development environment • Application model • Example
3.- Functions	• Embedded functions • CAS calling
4.- Real application examples	• Laboratory • Result checking • Circuit Analysis
5.- Conclusions	• C++ as future programming language bridging CAS • Advantages and Drawbacks • Final statement

4.2- Advantages & Drawbacks

Easy and powerful framework, versatility and complexity. Intuitive and useful applications.

Real and complex numbers, new language (?)

4.3- Final Statement

- C++ is the natural evolution to program calculator add-ins, providing both powerfulness (embedded functions, CAS calling) and simplicity (modular design).

- Web2.0 philosophy: intuitive applications, easy to share and run and capable of being improved by the community.

Suggestions and comments are welcome

Thank you for you attention