
commands of OllinTS

OllinTS is programmed to resolve cases where necessarily the mole fraction of the mixture of specified which derive the following cases:

1. Pressure Temperature
2. Pressure-Enthalpy
3. Pressure-Vapor Fraction
- Four. Temperature Vapor fraction
5. Temperature- enthalpy

Commands described below must be applied from an object can be a server or a thermodynamic property case. The syntax for these commands is that these should be written after a point separating the object to which point (? Object?..? Command?).

Administrator Commands

This section object to which the command is applied? **Ollin (*Ollin.Solve ()*)** which is the server properties.

- **AddModel (*TO* , *B* , *C*)**: Add a thermodynamic model administrator and returns the address of which can model created can be saved as a variable to enter the model created faster by following these specifications:

TO Specifies the name by which the thermodynamic model is identified and which is stored in the administrator, this name should be a string.

B : specifies the equation status to use thermodynamic model, the options are as follows.

RKS = RedlichKwongSimplicado RK =

RedlichKwong SRK =

SoaveRedlichKwong PR =

PengRobinson

When it omitted East server properties argument uses the default model RK

C : Designates the equation by which the vapor pressure within the thermodynamic model is calculated. OllinTS currently has a database for equations *Antonie Y Harlacher*, even though the latter is not recommended, because with the constants available in our database the results are incorrect. If this argument is omitted equation is used *Antoine*.

- **Add (TO , B)**: adds one or more compounds which are available in the database,

when the component is missing or does not match the database is performed

a search for compounds having some equivalence in the name and

printed on the screen. The name should be written in CAPITAL LETTERS. Because

the components can not be repeated, repeated compounds are ignored.

TO : specifies compounds which they are to be added.

B : Specifies the model thermodynamic to which the component is added.

- **Remove (TO , B)**: eliminates one or more compounds which are available in the model

Thermodynamic, in the absence of the compound an error message is printed. From

Similarly, the name should be capitalized.

TO Specifies the compound or compounds to be removed.

B Specifies the thermodynamic model to which the component is disposed.

- **AddCase (TO , B)**: adds a thermodynamic administrator case and returns the address of

If you can set up saved a variable for faster access, the case

created under the following specifications:

TO : It's the name with which the administrator will identify this case.

B : It is the name of the thermodynamic model which connects the thermodynamic case, when you omit this argument, the administrator automatically connect this with the first thermodynamic model in the manager added.

- **LoadConst (TO)**: Load all constants necessary for the thermodynamic model specified by the argument "TO" , if not specified will load the constant for all models that have been created.
- **Connect (TO , B)**: connect a thermodynamic model [TO] , a thermodynamically case [B] . East command is used when you need to change the thermodynamic model which solves the thermodynamic case.
- **Solve (TO)**: solve If the thermodynamic specified by the argument "TO" . When do not specified, all possible cases are resolved, if they have not defined the conditions balance prints an error.
- **Summary(TO)**: Print the results of a case decided by the argument specifying "TO" , if that is not the case resolved it prints an error message.
- **Comp ()**: prints compounds which are in thermodynamic models

Thermodynamic commands a case

For this section object to which I apply commands is determined by the name assigned to the thermodynamic case.

- **P (TO)**: Specifies the value of the pressure by the argument TO? in Kpa.
- **T (TO)**: Specifies the value of the temperature the argument "TO" in K.
- **FracVap (TO)**: specifies vaporized fraction defined by the argument "TO"
- **SetX (TO)**: specifies he value of the overall mole fraction defined by the argument "TO" , should be in list form, this command normalizes concentration.
- **Rx ()**: calculates fraction mole overall from the vaporized fraction and concentrations of each of the phases
- **reset ()**: Borra all intensive properties that define the conditions used to calculate the Balance phase.

- **CasePrint ()**: Print values general intensive properties.
- **XPrint ()**: Print the values of the equilibrium concentrations
- **Get (TO)**: Came back a fix the values specified by the argument **TO?** , Which

They must be saved in a variable or be used immediately. should not be confused with screen printing results. The names of the variables are listed in the appendix.

As a result we have a server properties able to calculate the phase equilibrium and basic thermodynamic properties that can be used to build more complex programs.

calculation Balance phase-Example

The procedure for the server application properties in an example of calculating the phase equilibrium described.

Example 1. Phase equilibrium pressure and defined temperature. For the hydrocarbon mixture of Table 10 is desired to know the temperature to which must operate so that the flash vaporized fraction is 0.5 to 101,325 KPa pressure.

Table 1. Composition of the mixture of hydrocarbons

compos or	Fraction mol
ethane	0.05
Propane	0.15
N-Butane	0.25
N-Pentane	0.20
N-Hexane	0.35

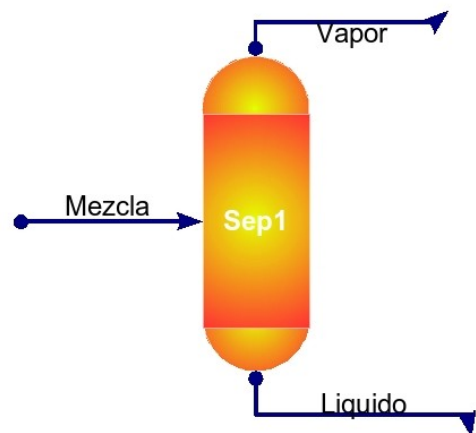


Figure 1. Separador by flash

Application Procedure: Commands can be executed directly on the command console Python *Home> All Programs> Python 2.4> Python (Comand line)* or you can type commands and store in a simple text file with the extension py to run together.

From the console command is invoked Python OllinTS

```
>>> ollin.Administrator.AdmOllin import from Olin
```

Loading Data Base data.db |

Create a thermodynamic models and naming

```
>>> PR = Ollin.AddModel ( "PR", "PR", "Antoine")
```

Add the compounds of the mixture

```
>>> Ollin.Add ([ "ETHANE" "PROPANE", "N-BUTANE", "N-PENTANE", "N-HEXANE"], "PR")
```

*ETHANE component 100 was add to PengRobinson component 132
PROPANE PengRobinson was add to 181 N-BUTANE component was
add to PengRobinson component 223 N-PENTANE PengRobinson was
add to N-HEXANE component 271 was add to PengRobinson*

Thermodynamic creates cases and store them in a variable

```
>>> S1 = Ollin.AddCase ( "S1")
```

Defines the composition of thermodynamic case

```
>>> S1.SetX ([0.05,0.15,0.25,0.20,0.35])
```

Specifies the equilibrium conditions

```
>>> S1.FracVap (0.5)
```

```
>>> S1.P (101,325)
```

Solves the case

```
>>> Ollin.Solve ()
```

*Solving S1 ... Pressure
Defined
...Defined FracVap*

Print results

```
>>> Ollin.Resumen ()
```

...:Summary :...

*FracVap = 0.5000 =
101,325 KPa Press*

Temp K = 295,273 ZL

= 0.006

ZV = 0.977 Z

= 0.492

Enthalpy kJ / kgmol = 2607.51497846 Entropy

KJ / KgmolK = 134.195509151 MolWt Kg /

kgmol = 67,241 MolWt L kg / kgmol = 76,538 V

MolWt kg / kgmol = 57945

```
...: Component :: ...      << >> << Vap Liq Fraction Fraction >>
ETHANE                    ==>  0.0046          | ____ |      .0954
PROPANE                   ==>  0.0395          | ____ |      .2605
N-BUTANE                   ==>  .1585          | ____ |      .3415
N-PENTANE                  ==>  .2335          | ____ |      .1665
N-HEXANE                   ==>  .5640          | ____ |      .1360
```

To access the results of a thermodynamic case? Get? Command is used where you specify the name of the variable. For example, for the values of the molar volume of the gas phase it is in the variable? Vvi? run the following command:

```
>>> S1.Get print (? Vvi?)
```

```
[24.09860111 23.92423415 23.67905276 23.34669897 22.90005727]
```

Results: The temperature at which the equipment must operate flash is 295 273 K, these conditions the mixture the more heavy compounds are concentrated. Table 2.COMPOSITION in the balance

Component	phase liquid composition gas phase	composition
ethane	0.0046	.0954
Propane	0.0395	.2605
N-Butane	.1585	.3415
N-Pentane	.2335	.1665
N-Hexane	.5640	.1360

Case 1. Construction of a diagram

phase

If described as OlliTS it can be used to construct the phase diagram where the evolution of the vaporized fraction of a hydrocarbon mixture is expressed from the bubble point at the dew point. This example is within OllinTS folder in the Examples section under the name *diagrama.py*.

Table 3. Composition of the mixture

Compound	Mol fraction
ethane	0.05
Propane	0.15
N-Butane	0.25
N-Pentane	0.20
N-Hexane	0.35

Application Procedure: The phase diagram is calculated for a pressure of 101,325 kPa, approximately bubble temperature is 240 K and pressure sprayed is 300 K. For the data needed to construct the diagram is calculated balance an interval of 2 K.

Then the source code described to construct the phase diagram:

Invokes OllinTS and pylab

```
ollin.Administrator.AdmOllin from import Ollin from pylab  
import *
```

Creates the thermodynamic model and adds the compounds


```
RK = Ollin.AddModel ( "RK", "RK", "Antoine")
```

```
Ollin.Add ([ "ETHANE" "PROPANE", "N-BUTANE", "N-PENTANE", "N-HEXANE"], "RK")
```

Creates a current and defines the composition and pressure of stream

```
S1 = Ollin.AddCase ( "S1")
```

```
S1.SetX ([0.05,0.15,0.25,0.20,0.35]) S1.P
```

```
(101,325)
```

Sets the range of phase equilibrium calculation

```
plot_x = range (240,300,2)
```

Define the variables where the data is saved

```
plot_y0 = []
```

```
plot_y1 = []
```

```
plot_y2 = []
```

```
plot_y3 = []
```

```
plot_y4 = []
```

```
plot_y5 = []
```

Starts calculating the equilibrium in the balance range

```
for T in plot_x:
```

Defines the temperature and settle the balance

```
S1.T (T)
```

```
Ollin.Solve ()
```

Retrieves the values of the concentration of the gas phase at equilibrium

```
f = S1.Get ( "f")
```

Saves the results in the variables

```
plot_y0.append (f [0])
```

```
plot_y1.append (f [1])
```

```
plot_y2.append (f [2])
```

```
plot_y3.append (f [3])
```

```
plot_y4.append (f [4])
```

```
plot_y5.append (S1.Get ( "FracVap"))
```

End of calculations

Grafica results

```
plot (plot_x, plot_y0) plot  
(plot_x, plot_y1) plot  
(plot_x, plot_y2) plot  
(plot_x, plot_y3) plot  
(plot_x, plot_y4) plot  
(plot_x, plot_y5)
```

Defines the characteristics diagram

```
axis ([244,300,0,1]) grid  
(True)  
titles = RK.library titles.append (  
"FracVap") legend (titles)  
  
title ( "Vapor Fraction Vs T, Y vs. T") xlabel ( 'T  
(K)') ylabel ( 'and FracVap')
```

Diagram shows

```
Show()
```

The resulting plot of the execution of this code represents the evolution of the concentration of hydrocarbons in the gas phase.

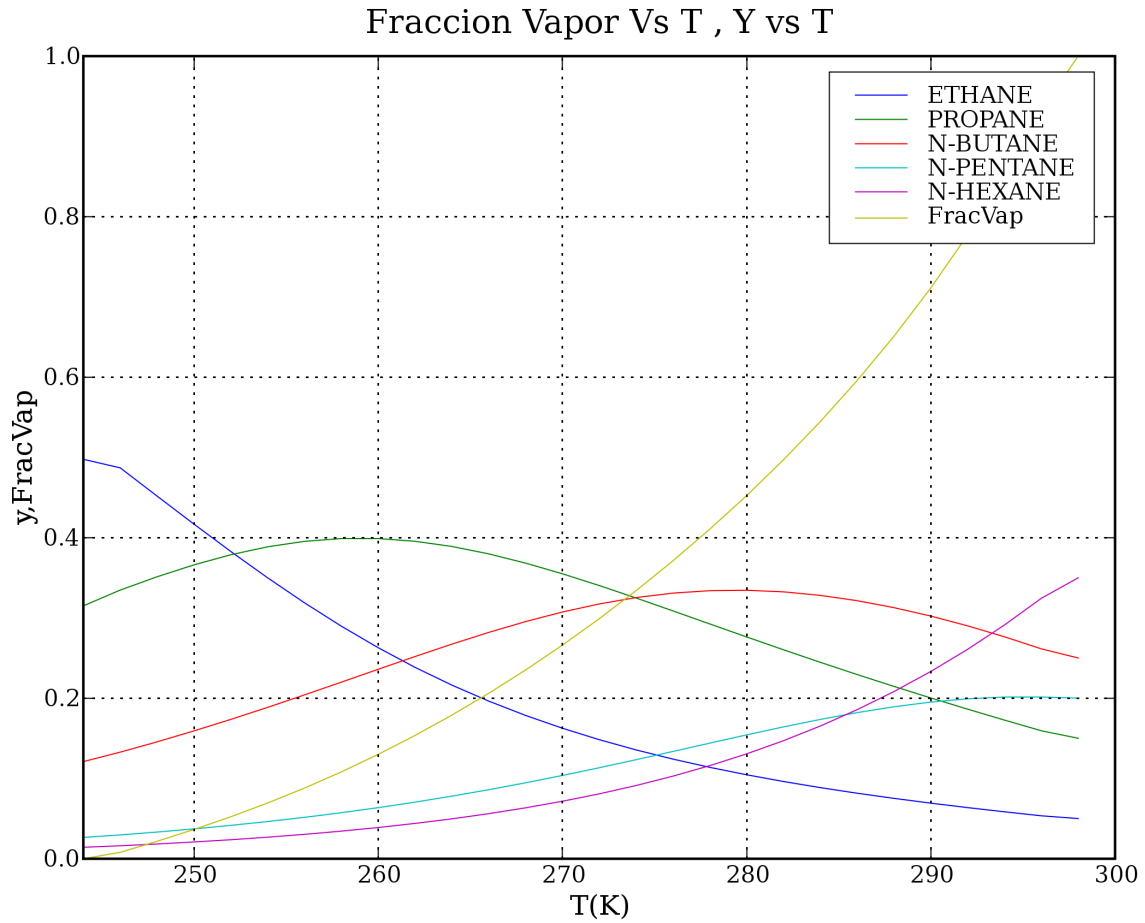


Figure 2. Graph phase equilibria

Results: In the diagram one can see that there is a high presence of ethane in the bubble point and with increasing vaporized fraction and the presence of going up until the time that the composition in the same initial mixing .

Case 2. Calculating the vapor pressure

With the help server properties can calculate the value of the vapor pressure corrected in the following example values of the vapor pressure calculated using the Antoine equation and the equation of Peng-Robinson for N-Butane is graphed and N-Heptane. This example is within OllinTS folder in the Examples section under the name *Presionv.py*.

Application Procedure: For equation Peng-Robinson vapor pressure is defined in a pure compound as the point where the value of fugacities for each phase is the same. The code by which you perform these graphs is written below.

Invoke OllinTS, pylab and lagrange interpolation tool

```
ollin.Administrator.AdmOllin from import Ollin from  
ollin.Tools.tools import lagrange from pylab import *
```

Create a model and a thermodynamic case

```
RK = Ollin.AddModel ( "RK", "PR") S1 =  
Ollin.AddCase ( "S1")  
Ollin.Add ([ "N-HEPTANE"], "RK") S1.SetX  
([1])
```

Defines the temperature range

```
Ti = range (300,450,10)
```

Create the variables to store the results.

```
Ppi = []  
Ppv = []
```

Starts calculating the vapor pressure

```
for T in Ti:
```

Create the variables to store the iterations

```
df = [] P  
= []
```

Define initial conditions and settle the balance

```
S1.P (101,325)  
S1.T (T) Ollin.Solve  
( )
```

Retrieving baseline

```
Pvi = S1.Get ( "PreVap") [0]  
Ppi.append (PVI) fl = S1.Get ( "fl_i")  
[0] = fv S1.Get ( "fv_i") [0]
```

```
P.append (S1.Get ( "P"))  
df.append (fl-fv) S1.P (PVI)  
Ollin.Solve ( )
```

```
fl = S1.Get ( "fl_i") [0] = fv  
S1.Get ( "fv_i") [0]  
P.append (S1.Get ( "P"))  
df.append (fl-fv)
```

Calculating the error value by fugacities

```
E = fl-fv
```

Starts iterations to calculate the vapor pressure

```
while abs (E)> 1e-3: Pi =  
    lagrange (df, P, 0) print S1.P  
    Pi (Pi) Ollin.Solve ( ) fl =  
    S1.Get ( "fl_i") [0] fv = S1.  
    Get ( "fv_i") [0]
```

```
P.append (S1.Get ( "P")) E =  
fl-fv
```

Adds the results to the list of actual values

```
df.append(E)
Ppv.append(Pi)
```

Plot the results and defines the characteristics of the graph

```
plot (Ti, Ppi) plot
(Ti, Ppv) grid
(True)
titles = [ "Antoine", "Peng-Robinson"] legend
(titles)
title ( "vapor pressure of N-Heptane") ylabel ( 'P
(Kpa)')
xlabel ( 'Temperature (K)')
```

Chart shows

Show()

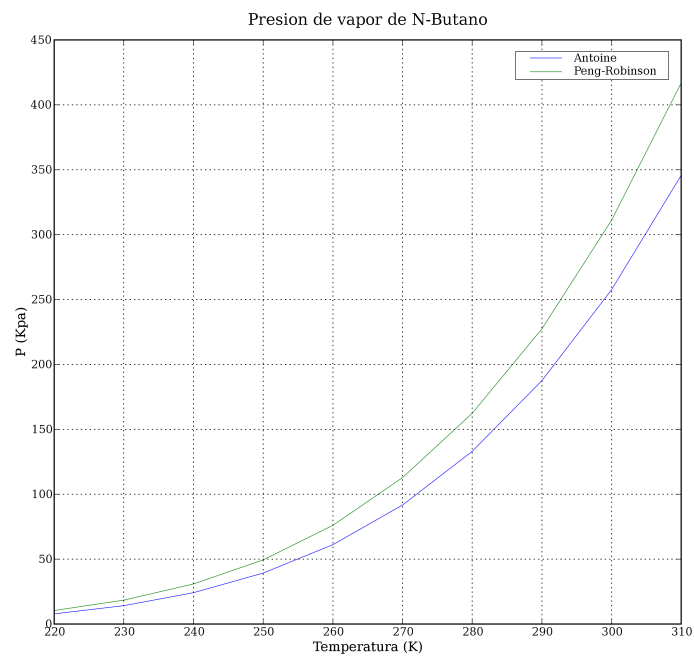


Figure 3. Vapor pressure of N-Butane

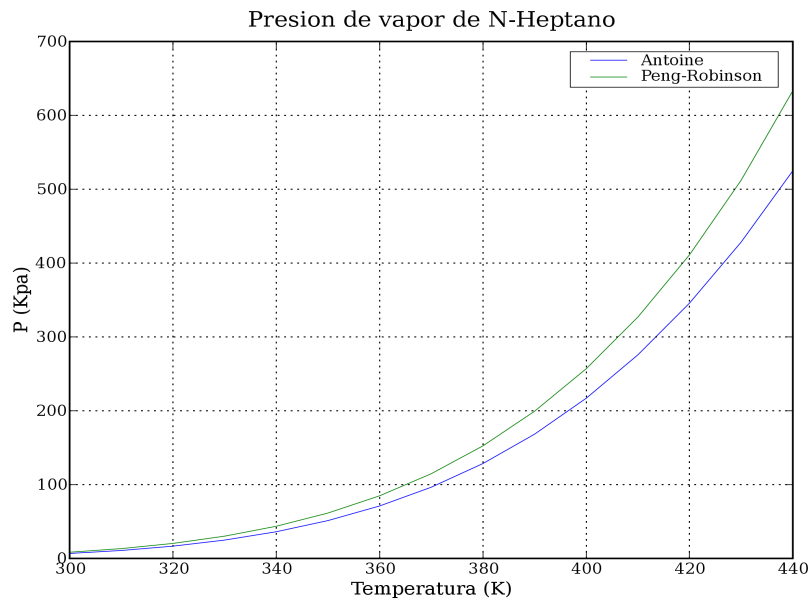


Figure 4. Vapor pressure of N-Heptane

Results: In both graphs shown as the value of the vapor pressure calculated by the equation cubic state is greater than that calculated by the Antoine equation, besides the deviation between both methods is becoming wider as it approaches the critical point.

Case 3. Design a

phase separator

LY

At the exit of a reactor producing benzene from toluene, it has a separator type flash phases of which one wishes to know the vaporized fraction and size of equipment to recover benzene to a flow of the mixture of 1919,605 kgmol / hr to a temperature of 311.15 ° K and a pressure of 3206,062 kPa. The feed composition is as follows:

Table 4. Composition of the mixture of aromatic

Compound	Mol fraction
Hydrogen	0.36602
Methane	0.54813
Benzene	0.062618
toluene	0.021503
diphenyl	0.000945

Application Procedure: Flash tank dimensions are determined by the volume of the liquid being processed, establishing a residence time of 5 minutes. For vertical tank it is recommended that the height of the tank is the height occupied by the liquid plus three times the diameter and height ratio of the diameter is 4. This example is within OllinTS folder in the Examples section under the TanqueFlash name. *py*.

So the length of the flash tank will be:

$$L = 3D \cdot \frac{V_L}{D^2} \quad (18)$$

And the diameter:

$$D = \frac{L}{4} \quad (18)$$

Pooling and operating both equations is obtained:

$$L = \frac{256V_L}{\pi} \quad (18)$$

Where:

L = Length Tank

D = diameter of tank

V_L = Fluid volume residing in the tank

The script to solve the problem posed is described in detail below, in this example OllinTS besides, you are needed to invoke the numerical value of the π

Variable power array and method.

solving OllinTS have this with the following code:

Invoke OllinTS, the constant π , and varying array

```
ollin.Administrator.AdmOllin from Olin import from
Numeric import array, power, pi
```

Create a thermodynamic model and defines the components

```
PR = Ollin.AddModel ( "PR", "PR")
Ollin.Add ([ "HYDROGEN", "METHANE", "BENZENE", "TOLUENE", "DIPHENYL"], "PR")
```

Thermodynamic case creates and defines the conditions

```
S1 = Ollin.AddCase ( "S1")
S1.SetX ([0.366021,0.548913,0.062618,0.021503,0.000945]) S1.T (38 +
273.15) S1.P (3206.062)
```

Thermodynamic solves the case and prints the results

```
Ollin.Solve ( "S1")
```

```
Ollin.Resumen ( "S1")
```

Calculate the liquid flow rate, the volume resident, the length and diameter

tank

```
L = (1-S1.Get ( "FracVap")) * 1919.605
```

```
Gv = (L * S1.Get ( "MolWt_L")) / (S1.Get ( "LiqDen") * 60) Vr = Gv * 5
```

```
Lon = power ((256 * Vr / pi), 0.333333) Dia = lon
```

```
/ 4
```

the results are printed

```
print "longitud" Lon print
```

```
"Diameter" Dia
```

Output the executing this code is as follows:

```
Loading Data Base data.db | .....
```

```
OllinTS has-been loaded
```

```
HYDROGEN component 19 was add to PengRobinson component  
was 61 METHANE add to PengRobinson component was 242  
BENZENE add to PengRobinson component was 286 TOLUENE add  
to PengRobinson component was 429 DIPHENYL add to  
PengRobinson Solving S1 ...
```

```
Defined Temperature
```

```
...Pressure Defined
```

```
...::Summary ...
```

```
FracVap = 0.9138 KPa
```

```
Press Temp = 3206,062
```

```
311,150 Z K = L = 0.132 Z =
```

```
1.031 V
```

$$Z = 0.954$$

Enthalpy KJ / kgmol = 6499.86626874 Entropy

KJ / Kg KgmolK = 236.251421265 MolWt / kgmol

= 16,562 MolWt L Kg / kgmol = 74,482 MolWt V

Kg / kgmol = 11098

...: Component :: .. << >> << Vap Liq Fraction Fraction >> HYDROGEN

==> 0.0164 | ____ | METHANE 0.3990 ==> 0.1051 | ____ | BENZENE

0.5908 ==> 0.6317 | ____ | TOLUENE 0.0089 ==> 0.2359 | ____ | 0.0013

DIPHENYL ==> 0.0110 | ____ | 0.0000

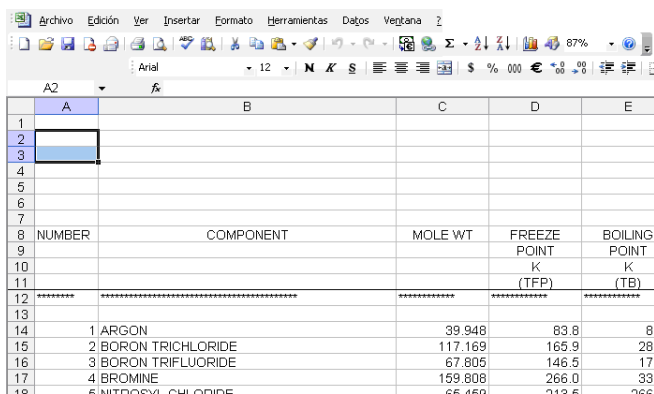
Longitud (m) 4.6729013708

Diameter (m) 1.1682253427

Results: At flash tank operating shows that the highest amount of hydrogen and methane is in the gas phase, so that benzene is in the liquid phase. Also are the dimensions of a diameter of 1.1682 and a height of 4.6729 meters. effectively,

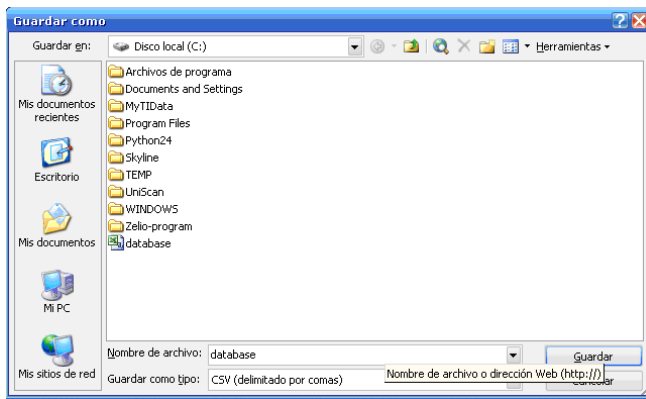
creation of the database

The following describes the procedure to create a database with SQL format from a spreadsheet with the help of a graphical application called SQLite? SQLite Data Browser? [SQLbrow, 2007].

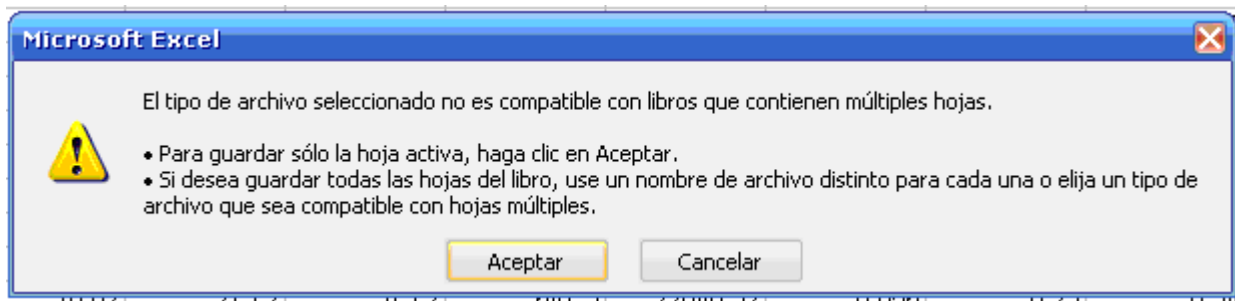


NUMBER	COMPONENT	MOLE WT	FREEZE POINT K (TFP)	BOILING POINT K (TB)
1	ARGON	39.948	83.8	87.3
2	BORON TRICHLORIDE	117.169	165.9	283.1
3	BORON TRIFLUORIDE	67.805	146.5	173.8
4	BROMINE	159.808	266.0	331.1
5	NITROSYL CHLORINE	65.454	213.5	266.1

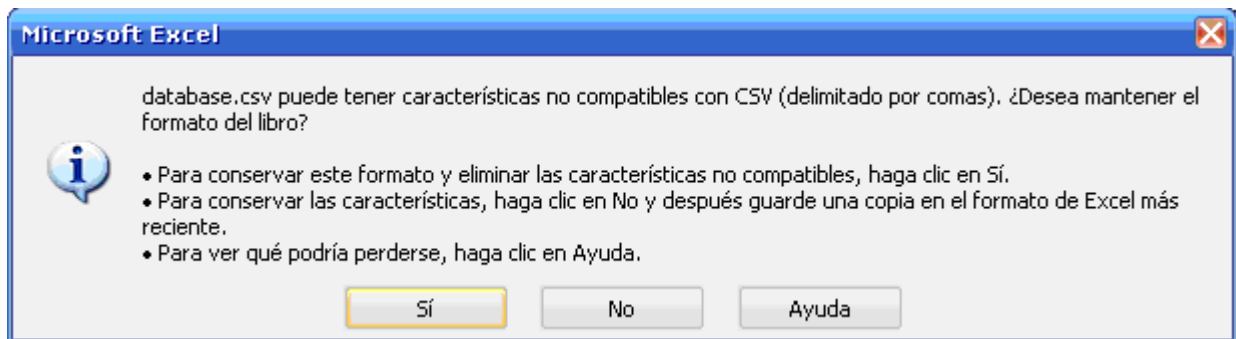
Initially available spreadsheet containing all necessary data which will be stored in a blank worksheet, the columns should not have any extra information. For example the name of the column.



Save the new spreadsheet with the CSV (comma delimited) which can be used by SQLite Database Browser



This window appears to warn us that we are selecting a format that does not support multiple sheets when the spreadsheet is saved. the "OK" option is selected to continue the process.



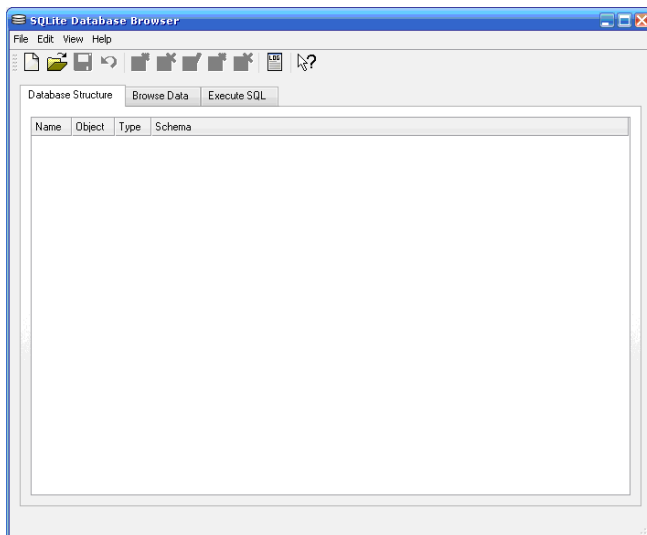
When this sale appears, the "Yes" is pressed to save the database in CSV format SQLite Database Browser.



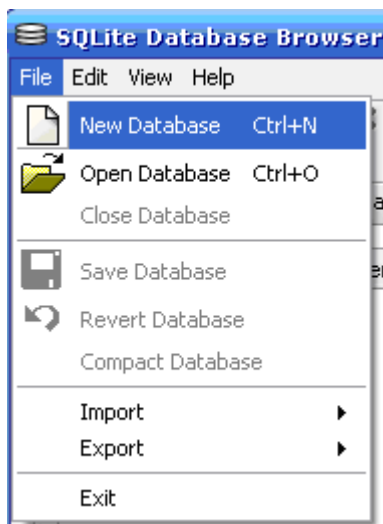
SQLite Database Browser

Once you've created the CSV file is run SQLite Database Browser, which is also an open source program. It can be downloaded from the website

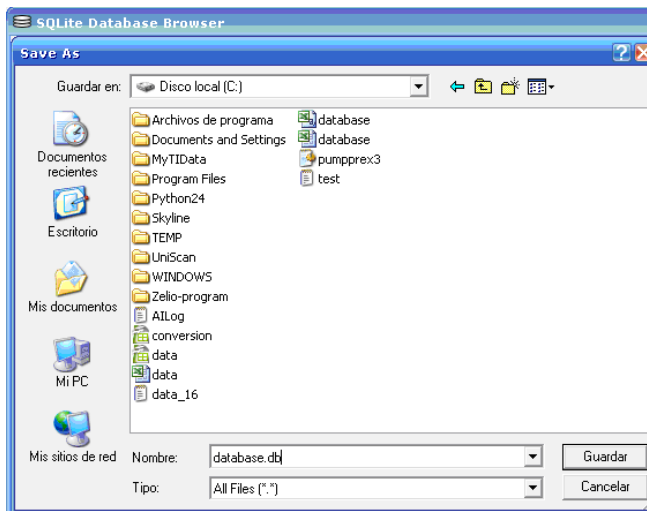
www.souceforge.net



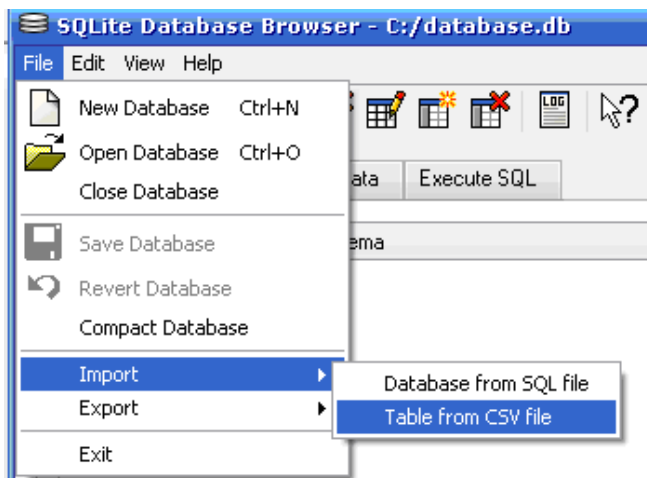
SQLite Database Browser lets you create the database in a very easy way, because they do not need to know SQL. Besides it is very close to the appearance of a spreadsheet.



Here is a database where insert our information is created. This can be done through the menu "File> New DataBase [Ctrl + N]"

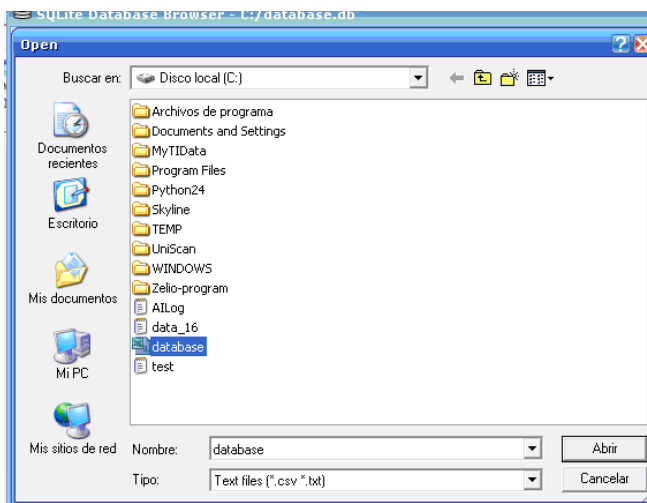


By selecting the Save option this window you can select the directory and name for the database appears. For this example the name database.db used.

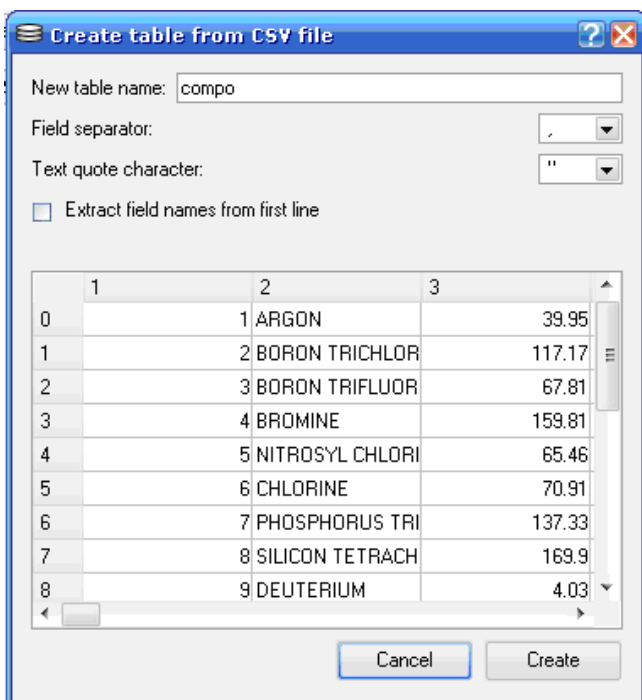


Then the spreadsheet format created previously imported.
To select the directory and file name to import

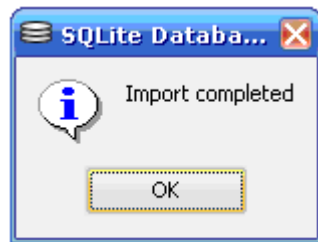
HE select the menu
File> Import> Table from CSV file.



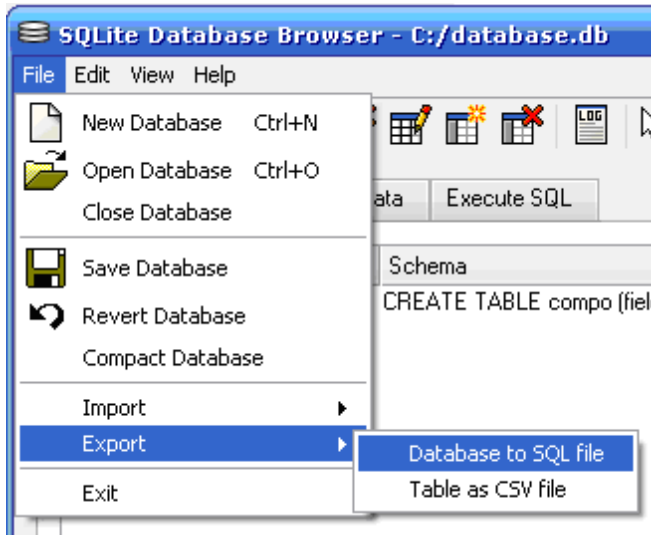
Through this window the directory and file name is
selected. For this example the file has the name
database.csv and finally open option is selected.



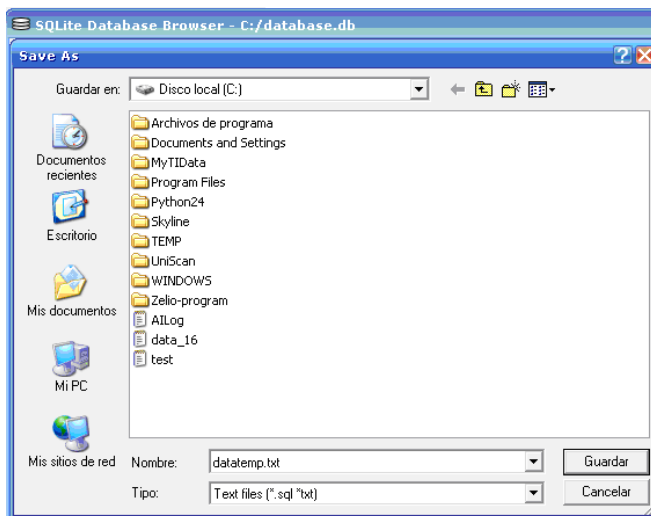
This window shows the result of the process of importing
the CSV file. In the "New table name:" name "compo" is
written, this is the name by which OllinTS accesses the
database. To create the base for imported s data select
the "Create" option



This sale confirms that imported information. Now we proceed to give the correct information to the fields of the database names.



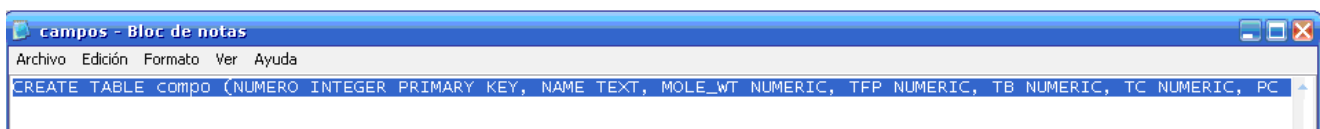
database in SQL format is exported, to change the name of the fields more easily.
 the action:
 File> Export> Database to SQL file.

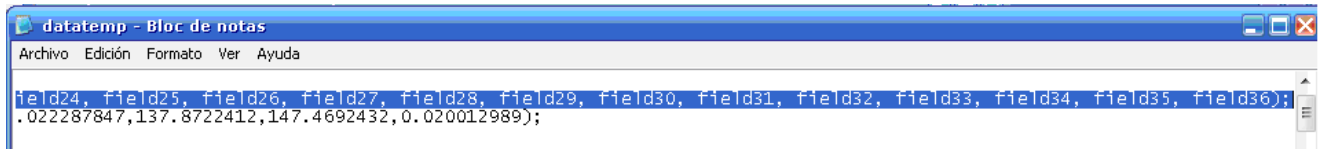


In selling the name given to the database in the format "txt" for this example the name is "datatemp.txt" option and select "Save"

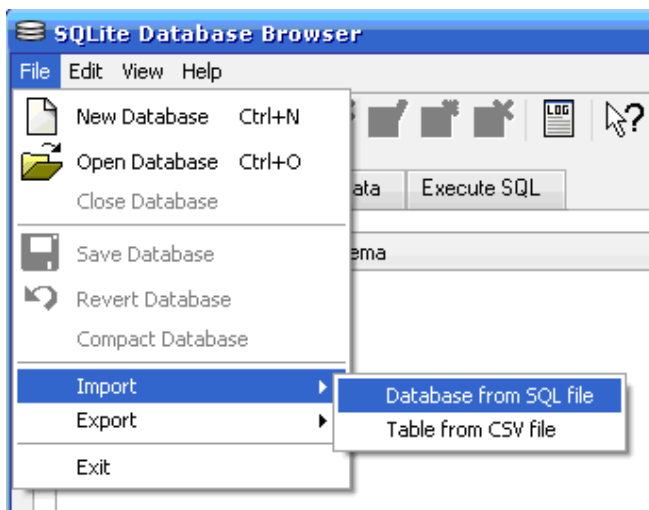


This window indicates that the information is to successfully exported.

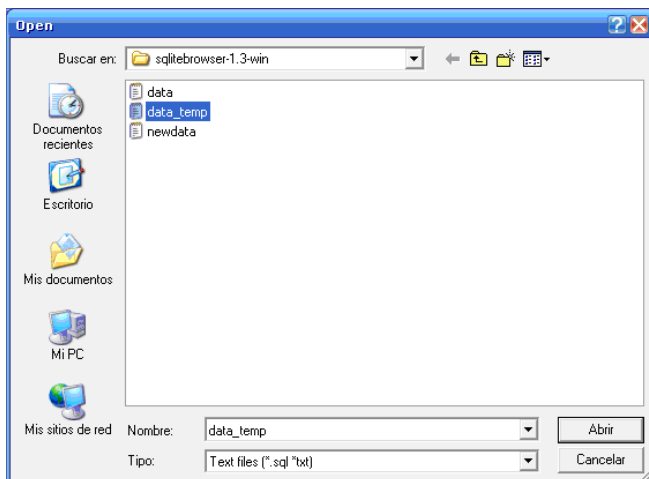




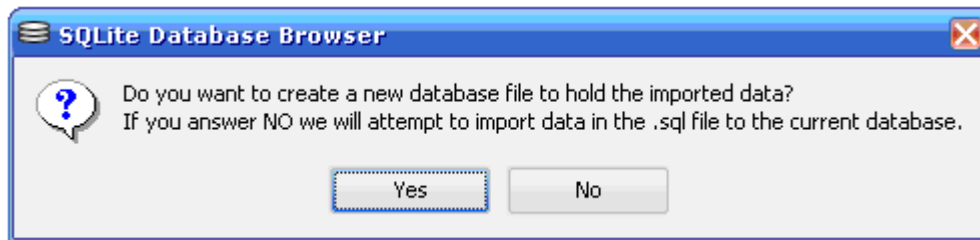
Then the "datatemp.txt" file is opened and the file "campos.txt" containing the names of the fields as the need OllinTS which the line that begins with "CREATE" and ends with "copy; "I'll replace the file" datatemp.txt "by the line that starts with" CREATE "and ends in"; ". At the end we keep the "datatemp.txt" file.



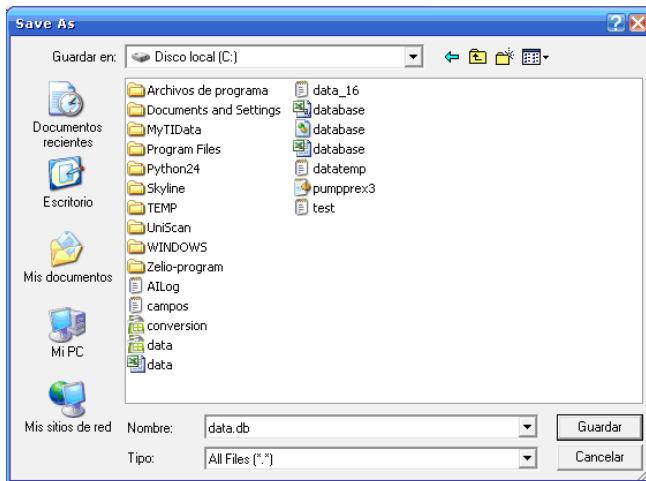
To import the corrected database, select it from the menu SQLite Data Browser: File> Import> DataBase from SQL file.



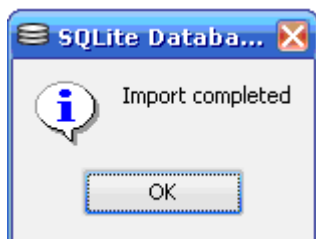
Through this window the modified file "datatemp.txt" is selected, and "Open" option is selected.



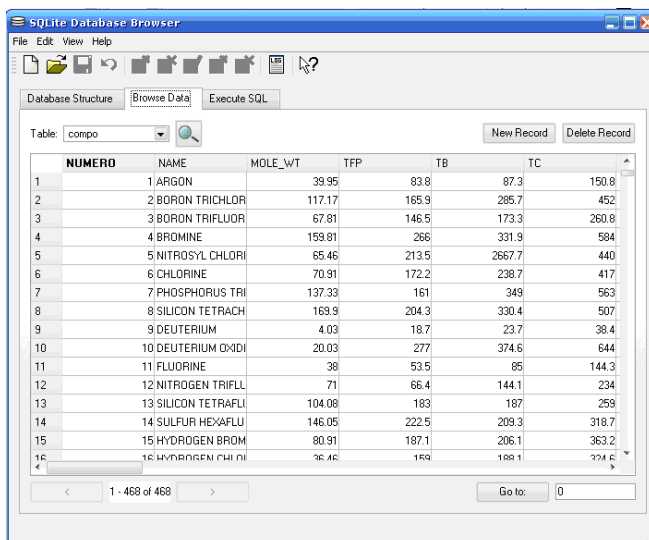
This window only appears when you have open another database, for which "YES" is selected to save the information into a new file.



The new file is given the name "data.db" because this is the name with which OllinTS calls the container file database.



This window confirms that the creation of the database completed successfully.



In the end I could check that the database is complete and no errors in the "Browse Data" tab

The new database must be copied to the DataBase folder inside the folder OllinTS.

for the operating system Windows XP is C: \ Python24 \ OllinTS \ DataBase

Appendix C: Installation Procedure

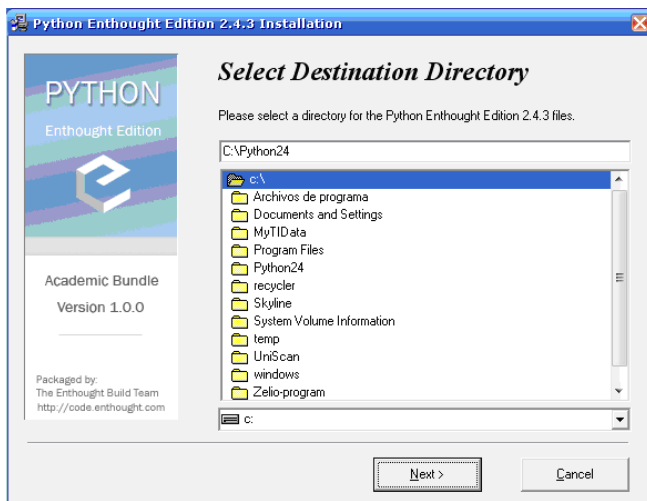
in Windows XP

OllinTS for installing Windows will use a modified version of Python (Python Enthought Edition) which includes all necessary libraries to run OllinTS, this version can be downloaded from the website <http://code.enthought.com/enthon/>. We will describe the process for installing Python and OllinTS.

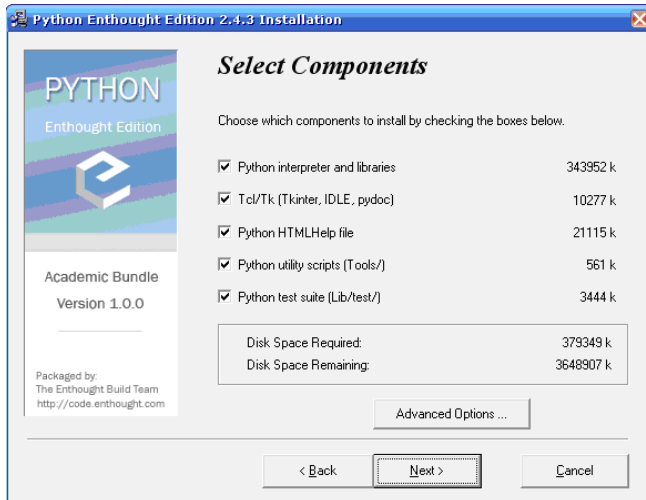


enthon-python2.4-1.0.0

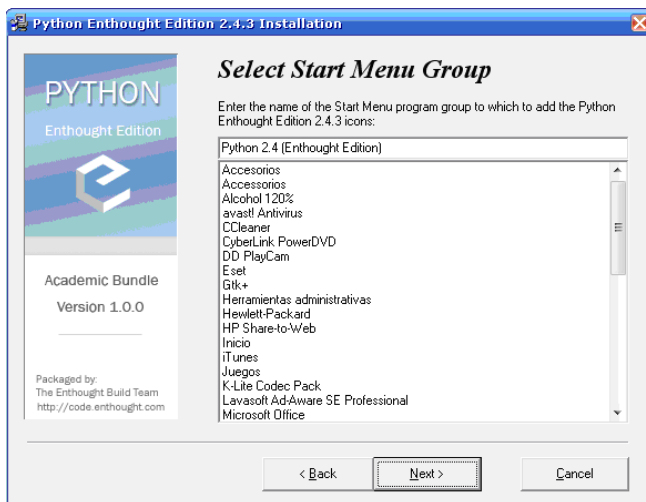
Run the installation program Python with the name "enthonpython2.41.0.0".



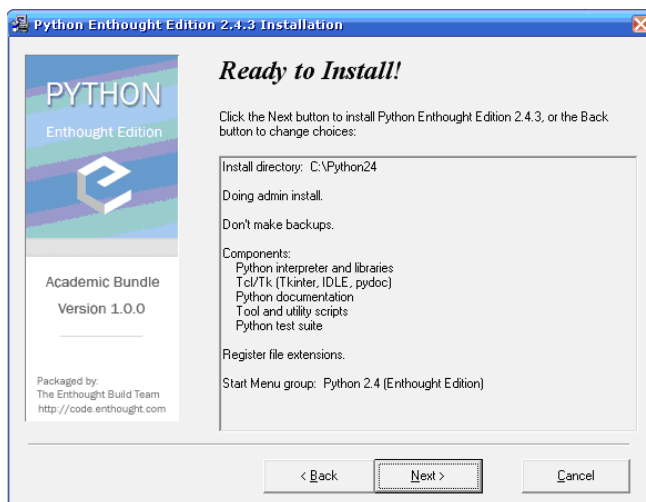
the installation direction is selected, it should be left as is
set (**C: \ Python24**) ,
Select "Next"



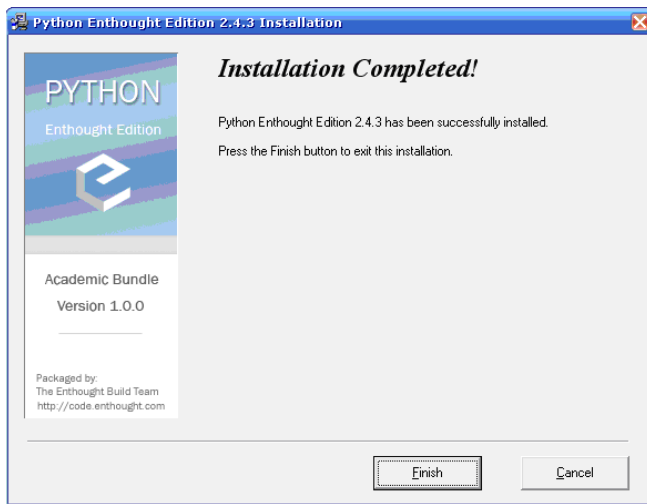
This window components installed with Python are selected, all boxes are selected. Select "Next"



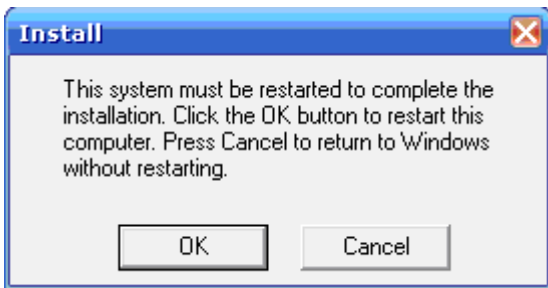
In this sale the group where shortcuts were created is selected, it is recommended to use the default. Select "Next"



In this window selected options before installing shown. Select "Next"



This window confirms that Python is installed.



Finally the computer is rebooted to complete the installation.

After the installation of Python, we proceed to copy the folder to the folder OllinTS Python, which is the route **C:\Python 24 **. Now you can run OllinTS.

OllinTS nomenclature

Table 5. Nomenclature OllinTS

Variable	units	Specifications
T	$^{\circ}K$	Temperature
P	<i>kpa</i>	Pressure
FracVap	-	vaporised fraction
xf	-	Mol fraction liquid fas
f	-	Mol fraction gas phase
x	-	mole fraction of the mixture
zl	-	Compressibility factor liquid phase
zV	-	Gas phase compressibility factor
CoefPureVap	-	Fugacity coefficient of numerical pure compounds gas- phase Arrangement
CoefMixVap	-	Fugacity coefficient of compounds in gas- phase mixture numerical Arrangement
CoefMixVLiq	-	Fugacity coefficient of the compounds in admixture Liquid phase numerical Arrangement
VVI	$\frac{M_3}{kgmol}$	Volume numerical gas- phase pure compounds Arrangement
VLI	$\frac{M_3}{kgmol}$	Volume of Liquid phase pure compounds numerical Arrangement
Vv	$\frac{M_3}{kgmol}$	Gas phase volume

Table 5. Nomenclature OllinTS (continued)

Variable	units	Specifications
vl	$\frac{M_3}{kgmol}$	Volume of liquid phase pure compounds
ActivityVap	-	Activity coefficient of the gas phase pure compounds - numerical Arrangement
ActivityLiq	-	Activity coefficient of liquid phase pure compounds - numerical Arrangement
PreVap	KPa	Vapor pressure - numeric array
Ki	-	Coefficient of distribution numeric array
AlphaT	-	Temperature function
Tr	-	reduced temperature
fw	-	Acentric factor function and reduced temperature - numerical Arrangement
to	-	Factor "a" for the cubic equation of state
TO	-	A factor for the cubic equation of state
B	-	Factor B for the cubic equation of state
DADT	-	First derivative of the factor "a" for the cubic equation of state
d2adT2	-	Second factor derived from "a" to the cubic equation of state
MolWt	$\frac{kg}{kgmol}$	average molecular mass of the mixture
MolWt_l	$\frac{kg}{kgmol}$	average molecular weight of the liquid phase
MolWt_v	$\frac{kg}{kgmol}$	average molecular mass of the vapor phase
LiqDen	$\frac{kg}{M_3}$	average liquid density
Cp_v	$\frac{KJ}{Kgmol^{\circ}K}$	heat capacity at constant pressure vapor-phase numerical Arrangement
Cv_v	$\frac{KJ}{Kgmol^{\circ}K}$	heat capacity at constant volume vapor-phase numerical Arrangement

Table 5. Nomenclature OllinTS (continued)

Variable	units	Specifications
HF	$\frac{KJ}{kgmol}$	Standard Power Training
GF	$\frac{KJ}{kgmol}$	Gibbs free energy of formation
G	$\frac{KJ}{kgmol}$	Gibbs free energy mix
H	$\frac{KJ}{kgmol}$	Enthalpy of mixing
S	$\frac{KJ}{kgmol}$	Entropy mix
OR	$\frac{KJ}{kgmol}$	internal energy of the mixture
Afree	$\frac{KJ}{kgmol}$	Helmholtz free energy
G_v	$\frac{KJ}{kgmol}$	Gibbs free energy of gas phase
G_L	$\frac{KJ}{kgmol}$	Gibbs free energy liquid phase
H_v	$\frac{KJ}{kgmol}$	Gas phase enthalpy
h_l	$\frac{KJ}{kgmol}$	free enthalpy of the liquid phase
S_v	$\frac{KJ}{Kgmol^{\circ}K}$	Entropy gas phase
S_L	$\frac{KJ}{Kgmol^{\circ}K}$	Free entropy of the liquid phase
U_v	$\frac{KJ}{kgmol}$	internal energy of the gas phase
U_l	$\frac{KJ}{kgmol}$	internal energy of the liquid phase
AFree_v	$\frac{KJ}{kgmol}$	Helmholtz free energy of the gas phase

Variable	units	Specifications
AFree_l	$\frac{KJ}{kgmol}$	Helmholtz free energy of the liquid phase

Table 6. Nomenclature database

OllinTS name Ollin.DataBase.Sy sData	Name in the Database	Description	units
ZC	ZC	Critical compressibility factor	-
OMEGA	OMEGA	Pitzer acentric factor	-
<i>LIQDEN</i>	<i>LIQDEN</i> liquid density		
<i>TDEN</i>	<i>TDEN</i>	Temperature of the liquid density	$^{\circ}K$
<i>DIM</i>	<i>DIM</i>	Dipole momentum	-
CP_A	CP_A	Coefficient of heat capacity of ideal gas A	$\frac{KJ}{Kg mol^{\circ}K}$
CP_B	CP_B	Coefficient of heat capacity of ideal gas B	$\frac{KJ}{Kg mol^{\circ}K}$
CP_C	CP_C	Coefficient of heat capacity of ideal gas C	$\frac{KJ}{Kg mol^{\circ}K}$
CP_D	CP_D	Coefficient of heat capacity of ideal gas D	$\frac{KJ}{Kg mol^{\circ}K}$
VL_B	VISC_LIQ_B Viscosity coefficient of fluid B		C_p
<i>VL_C</i>	<i>VISC_LIQ_C</i> Viscosity coefficient of liquid C		C_p
<i>DELHF</i>	<i>DEL_HF</i>	Energy standard training	$\frac{KJ}{kg mol}$
<i>ANTA</i>	<i>ANTOINE_VP</i> <i>_TO</i>	A coefficient for Antoine equation for	$P = mmHg$ $T = ^{\circ}K$

Table 6. Nomenclature database (continued)

OllinTS name Ollin.DataBase.Sy sData	Name in the Database	Description	units
<i>ANT_B</i>	<i>ANTOINE_VP _B</i>	Coefficient B for for Antoine equation	$P = \text{mmHg}$ $T = ^\circ K$
<i>ANT_C</i>	<i>ANTOINE_VP _C</i>	Coefficient C for Antoine equation	$\text{MmHg } P = T =$ $^\circ K$
<i>ANT_MAX</i>	<i>TMAX</i>	maximum temperature for Antoine equation	$^\circ K$
<i>ANT_MIN</i>	<i>TMIN</i>	minimum temperature for Antoine equation	$^\circ K$
<i>HAR_A</i>	<i>HARLACHER_ VP_A</i>	A coefficient for equation for Harlacher	$\text{MmHg } P = T =$ $^\circ K$
<i>HAR_B</i>	<i>HARLACHER_ VP_B</i>	Coefficient B for the equation for Harlacher	$\text{MmHg } P = T$ $= ^\circ K$
<i>HAR_C</i>	<i>HARLACHER_ VP_C</i>	Coefficient C for equation Harlacher	$\text{MmHg } P = T$ $= ^\circ K$
<i>HAR_D</i>	<i>HARLACHAR_ VP_D</i>	Coefficient D for equation Harlacher	$P = \text{mmHg}$ $T = ^\circ K$
<i>HV</i>	<i>HV</i>	Standard vaporization heat	$\frac{\text{KJ}}{\text{kgmol}}$
<i>RK_A</i>	<i>RK_ac</i>	ac constant for the Redlich-Kwong case	-
<i>RK_B</i>	<i>RK_b</i>	b constant for the case Redlich-Kwong	-

Table 6. Nomenclature database (continued)