

# Test Examples Guide

## Introduction:

Simulator is the OpenModelica library created at FOSSEE, IITB. The main objective of this is to create general purpose steady state chemical engineering simulator using OpenModelica.. You can download this library from - <https://github.com/FOSSEE/OMChemSim>

Test examples are available in last section of Simulator. This document is created for giving step by step guidelines of how these test examples can be reproduced. One can learn, how to use this library using this guide. Readers are expected to practice these steps parallelly to learn effectively.

## Table of Contents

1. Material_Stream:	3
2. Heater:	7
3. Cooler:	9
4. Valve:	11
5. Mixer:	13
6. Compound_Separator:	16
7. Shortcut_Column:	18
8. Flash:	20
9. Splitter:	22
10. Centrifugal_Pump:	24
11. Adiabatic_Compressor:	26
12. Adiabatic_Expander:	28

**NOTE: Load Simulator.mo file before creating examples.**

## 1. Material\_Stream:

### msTP:

This is simulation of material stream consisting of ternary system, Methanol, Ethanol and Water at pressure 101325 Pa and temperature 351 K. Mole fractions of these components in mixture is 0.33, 0.33, 0.34 respectively.

1. Create an empty model named "msTP".

2. Create an instance of Chemsep Database.

```
import data = Simulator.Files.Chemsep_Database;
```

3. Create instances of components to be used.

```
parameter data.Methanol meth;  
parameter data.Ethanol eth;  
parameter data.Water wat;
```

4. Extend Material\_Stream model in Streams section of Simulator and specify values of NOC(number of components), comp(components array) and also give start values for some variables.

```
extends Simulator.Streams.Material_Stream(NOC = 3, comp =  
{meth, eth, wat}, compMolFrac(each min = 0.01, each max = 1,  
start = {{0.33, 0.33, 0.34}, {0.32, 0.33, 0.34}, {0.53, 0.32,  
0.14}}), totMolFlo(each start = 50));
```

5. Extend Thermodynamic package Raoult's Law.

```
extends Simulator.Files.Thermodynamic_Packages.Raoult's_Law;
```

6. Specify values of input variables in equation section. Here P is pressure in Pa, T is temperature in K, compMolFrac[1, :] is component mole fractions of mixture and totMolFlo[1] is total molar flow rate of mixture in mol/s.

```
equation  
  P = 101325;  
  T = 351;  
  compMolFrac[1, :] = {0.33, 0.33, 0.34};  
  totMolFlo[1] = 100;
```

7. Simulate this model.

Final code will look like follows:

```
model msTP
  import data = Simulator.Files.Chemsep_Database;
  parameter data.Methanol meth;
  parameter data.Ethanol eth;
  parameter data.Water wat;
  extends Streams.Material_Stream(NOC = 3, comp = {meth, eth, wat},
  compMolFrac(each min = 0.01, each max = 1, start = {{0.33, 0.33,
  0.34}, {0.32, 0.33, 0.34}, {0.53, 0.32, 0.14}}), totMolFlo(each start
  = 50));
  extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;

equation
  P = 101325;
  T = 351;
  compMolFrac[1, :] = {0.33, 0.33, 0.34};
  totMolFlo[1] = 100;
end msTP;
```

Note: P and T are mode variables in this example. Material stream supports four other modes. msTVF, msPVF, msPH and msPS are test examples for other modes. All other code remains same for these modes only input variables specified are different.

- msTVF : Temperature (T) in K and vapor fraction( vapPhasMolFrac) is specified.
- msPVF: Pressure(P) in Pa and vapor fraction(vapPhasMolFrac) is specified.
- msPH: Pressure(P) in Pa and Mixture molar Enthalpy(phasMolEnth[1]) in J/mol is specified.
- msPS: Pressure(P) in Pa and Mixture molar Entropy(phasMolEntr[1]) in J/mol.K is specified.

msTPbbp is TP mode simulation only but in this Temperature of system is below bubble point.

### **cmpstms:**

This is simulation of material stream only but we are creating composite material stream first and then using it in main model. Composite material stream is the model in which only material stream and thermodynamic package is extended. Composite material stream is very useful when we need multiple instantiations of material stream containing same thermodynamic model while creating flowsheets.

1. Create a package named "cmpstms".

2. Create new model named "ms" in this package.
3. Extend Material\_Stream model inside "ms".  

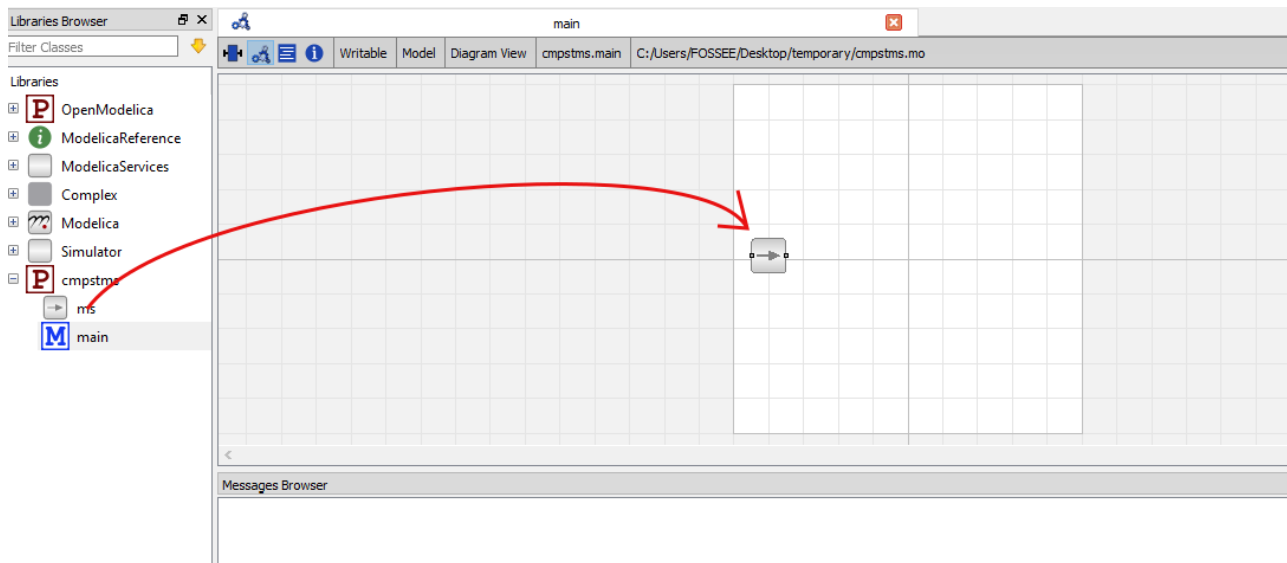
```
extends Simulator.Streams.Material_Stream;
```
4. Extend required thermodynamic package.  

```
extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;
```
5. Create new package in "cmpstms" named "main".
6. Now, inside model main, create an instance of Chemsep Database.  

```
import data = Simulator.Files.Chemsep_Database;
```
7. Create instances of components to be used.  

```
parameter data.Benzene benz;  
parameter data.Toluene tol;
```
8. Create an Integer parameter for NOC(number of components) and create a component array.  

```
parameter Integer NOC = 2;  
parameter data.General_Properties comp[NOC] = {benz, tol};
```
9. Open diagram view of "main" model. Drag and drop ms as shown in fig. Name it as ms1.



10. Switch to text view there will be generated code for instantiation of "ms". Assign values of NOC and comp in bracket.  

```
Simulator.Test.cmpstms.ms ms1(NOC = NOC, comp = comp)  
annotation( ... );
```

11. Assign values of variables in equation section. Note that these variable names are different than msTP.

```
equation
  ms1.P = 101325;
  ms1.T = 368;
  ms1.totMolFlo[1] = 100;
  ms1.compMolFrac[1, :] = {0.5, 0.5};
```

12. Simulate model "main".

Final code will look like follows.

```
package cmpstms
model ms
  extends Simulator.Streams.Material_Stream;
  extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;
end ms;

model main
  import data = Simulator.Files.Chemsep_Database;
  parameter data.Benzene benz;
  parameter data.Toluene tol;
  parameter Integer NOC = 2;
  parameter data.General_Properties comp[NOC] = {benz, tol};
  Simulator.Test.cmpstms.ms ms1(NOC = NOC, comp = comp) annotation(
    Placement(visible = true, transformation(origin = {-80, 2},
  extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  equation
    ms1.P = 101325;
    ms1.T = 368;
    ms1.totMolFlo[1] = 100;
    ms1.compMolFrac[1, :] = {0.5, 0.5};
  end main;
end cmpstms;
```

## 2. Heater:

### heater1:

This is simulation of heat addition (2000 kJ) in ternary mixture of methanol, ethanol and water having mole fractions 0.33, 0.33, 0.34 respectively at 202650 Pa and 320 K. Pressure drop across heater is 101325 Pa.

1. Create package heater1.
2. Create composite material stream model "ms" by extending Material\_Stream model and required thermodynamic package.

```
model ms
  extends Simulator.Streams.Material_Stream;
  extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;
end ms;
```

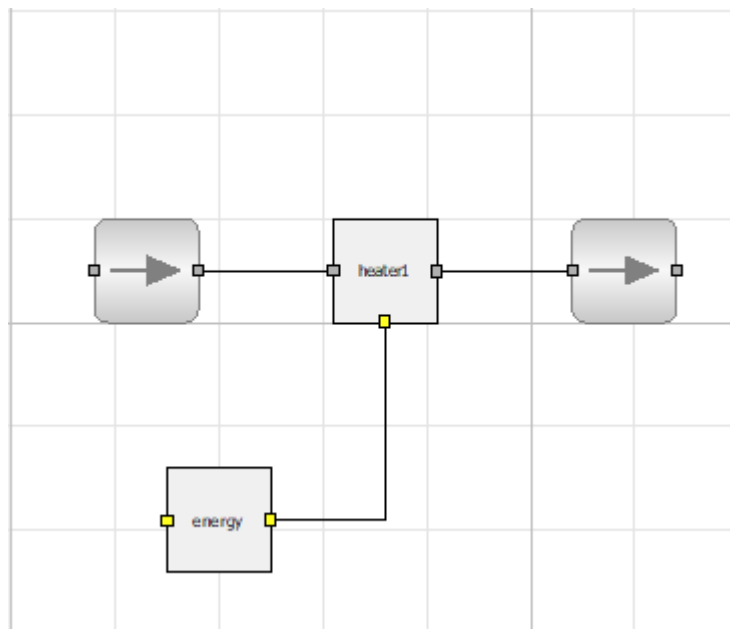
3. Create new model "heat" in "heater1".
4. Now, inside model "heat", create an instance of Chemsep Database and create instances of components to be used.

```
import data = Simulator.Files.Chemsep_Database;
parameter data.Methanol meth;
parameter data.Ethanol eth;
parameter data.Water wat;
```

5. Create an integer parameter for NOC(number of components) and create a component array.

```
parameter Integer NOC = 3;
parameter data.General_Properties comp[NOC] = {meth, eth,
wat};
```

6. Now open diagram view. Create two instances of "ms" as "inlet" and "outlet". Create instance of heater model from Simulator.Unit\_Operations as "heater1" also create instance of Energy stream from Simulator.Streams as "energy". Connect them as shown in fig.



7. Switch to text view and specify parameters in "heater1", "inlet" and "outlet" models. Specify some start values of "outlet" stream.

```
Simulator.Unit_Operations.Heater heater1(pressDrop = 101325,  
eff = 1, NOC = NOC, comp = comp) annotation( ...);  
heater1.ms inlet(NOC = NOC, comp = comp) annotation( ...);  
heater1.ms outlet(NOC = NOC, comp = comp, T(start = 353),  
compMolFrac(start = {{0.33, 0.33, 0.34}, {0.24, 0.31, 0.43}, {0.44,  
0.34, 0.31}}), P(start = 101325)) annotation( ...);
```

8. Specify values of input variables in equation section.

```
equation  
inlet.compMolFrac[1, :] = {0.33, 0.33, 0.34};  
inlet.P = 202650;  
inlet.T = 320;  
inlet.totMolFlo[1] = 100;  
heater1.heatAdd = 2000000;
```

9. Simulate model "heat".

Note: here heatAdd is the mode variable for heater. This heater model supports more modes. We just need to change mode variable to use respective mode. These modes are as follow.

- Outlet temperature: value of outlet temperature(outT) is specified in K.
- Phase Molar Fraction: value of Vapor phase molar fraction of outlet(outVapPhasMolFrac) is specified.
- Temperature increase: value of temperature change(tempInc) is specified.
- Energy Stream: value of enFlo variable from Energy\_Stream(energy.enFlo) is specified in Joules.



### 3. Cooler:

#### cooler1:

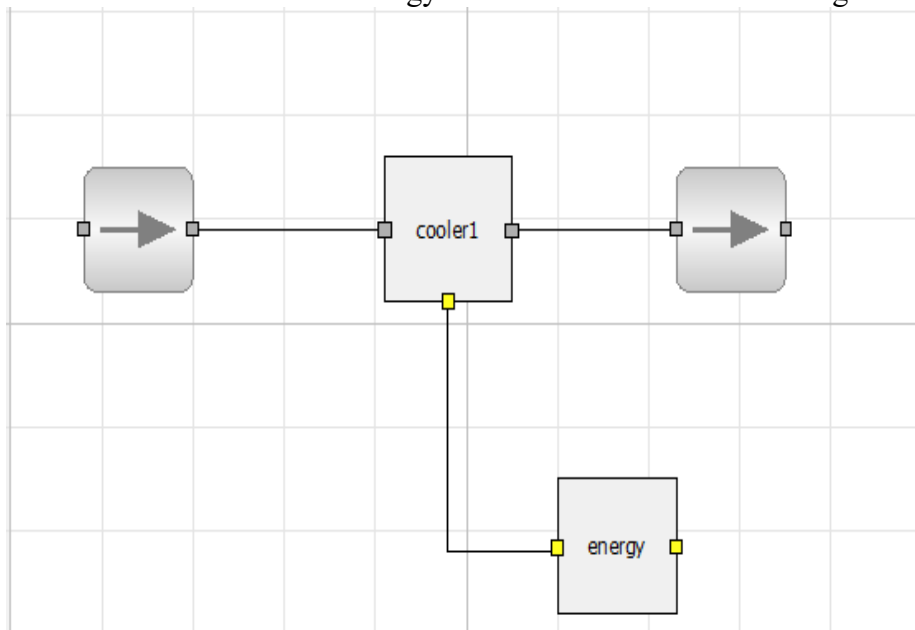
This is simulation of heat removal(2000 kJ) from ternary mixture of methanol, ethanol and water having mole fractions 0.33, 0.33, 0.34 respectively at 101325 Pa and 353 K.

1. Create package cooler1.
2. Create composite material stream model "ms" by extending Material\_Stream model and required thermodynamic package.

```
model ms
  extends Simulator.Streams.Material_Stream;
  extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;
end ms;
```
3. Create new model "cool" in "cooler1".
4. Now, inside model "cool", create an instance of Chemsep Database and create instances of components to be used.

```
import data = Simulator.Files.Chemsep_Database;
parameter data.Methanol meth;
parameter data.Ethanol eth;
parameter data.Water wat;
```
5. Create an Integer parameter for NOC(number of components) and create a component array.

```
parameter Integer NOC = 3;
parameter data.General_Properties comp[NOC] = {meth, eth,
wat};
```
6. Now open diagram view. Create two instances of "ms" as "inlet" and "outlet". Create instance of cooler model from Simulator.Unit\_Operations as "cooler1" also create instance of Energy stream from Simulator.Streams as "energy". Connect them as shown in fig.



7. Switch to text view and specify parameters in "cooler1", "inlet" and "outlet" models. Specify some start values of "outlet" stream.

```
Unit_Operations.Cooler cooler1(pressDrop = 0, eff = 1, NOC =  
3, comp = {meth, eth, wat}) annotation( ...);  
ms inlet(NOC = 3, comp = {meth, eth, wat}) annotation( ...);  
ms outlet(NOC = 3, comp = {meth, eth, wat}, T(start =  
352.6146), compMolFrac(start = {{0.33, 0.33, 0.34}, {0.27, 0.32,  
0.39}, {0.48, 0.33, 0.18}}), P(start = 101325))  
annotation( ...);
```

8. Specify values of input variables in equation section.

```
equation  
inlet.compMolFrac[1, :] = {0.33, 0.33, 0.34};  
inlet.P = 101325;  
inlet.T = 353;  
inlet.totMolFlo[1] = 100;  
cooler1.heatRem = 200000;
```

9. Simulate model "cool".

Note: here heatRem is the mode variable for heater. This cooler model supports more modes. We just need to change mode variable to use respective mode. These modes are as follow.

- Outlet temperature: value of outlet temperature(outT) is specified in K.
- Vapor Phase Molar Fraction: value of Vapor phase molar fraction of outlet(outVapPhasMolFrac) is specified.
- Temperature drop: value of temperature change(tempDrop) is specified.
- Energy Stream: value of enFlo variable from Energy\_Stream(energy.enFlo) is specified in Joules.

#### 4. Valve:

##### valve1:

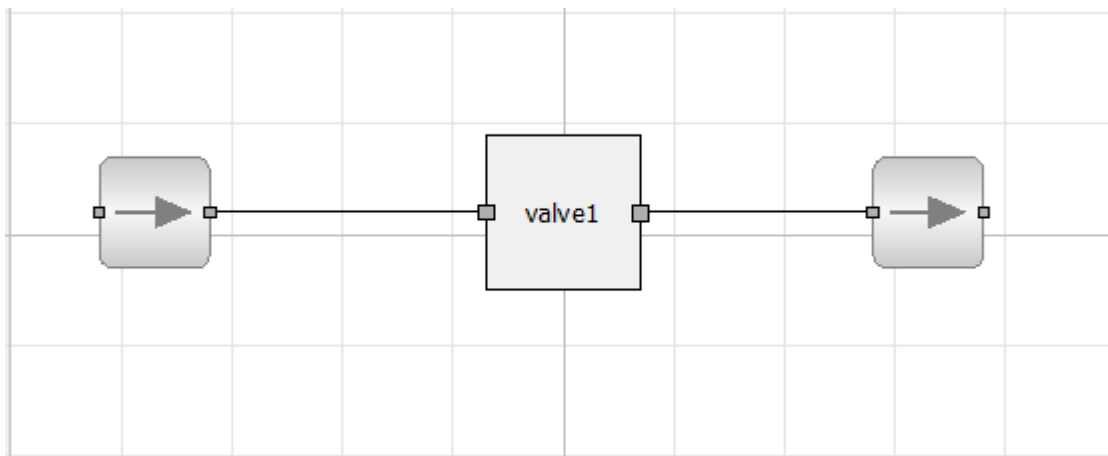
This is simulation of pressure drop(101325 Pa) of stream of ternary mixture containing methanol, ethnaol and water having mole fractions 0.33, 0.33, 0.34 respectively at 202650 Pa and 372 K.

1. Create package valve1.
2. Create composite material stream model "ms" by extending Material\_Stream model and required thermodynamic package.

```
model ms
  extends Simulator.Streams.Material_Stream;
  extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;
end ms;
```
3. Create new model "valve" in "valve1".
4. Now, inside model "valve", create an instance of Chemsep Database and create instances of components to be used.

```
import data = Simulator.Files.Chemsep_Database;
parameter data.Methanol meth;
parameter data.Ethanol eth;
parameter data.Water wat;
```
5. Create an Integer parameter for NOC(number of components) and create a component array.

```
parameter Integer NOC = 3;
parameter data.General_Properties comp[NOC] = {meth, eth,
wat};
```
6. Now open diagram view. Create two instances of "ms" as "inlet" and "outlet". Create instance of valve model from Simulator.Unit\_Operations as valve1. Connect them as shown in fig.



7. Switch to text view and specify parameters in "valve1", "inlet" and "outlet" models. Specify some start values of "outlet" stream.

```
Unit_Operations.Valve valve1(NOC = NOC, comp = comp)
  annotation( ...);
valve1.ms inlet(NOC = NOC, comp = comp) annotation(
...);
valve1.ms outlet(NOC = NOC, comp = comp, T(start = 352),
compMolFrac(start = {{0.33, 0.33, 0.34}, {0.26, 0.32, 0.40}, {0.47,
0.34, 0.18}})) annotation( ...);
```

8. Specify values of input variables in equation section.

```
inlet.compMolFrac[1, :] = {0.33, 0.33, 0.34};
inlet.P = 202650;
inlet.T = 372;
inlet.totMolFlo[1] = 100;
valve1.pressDrop = 101325;
```

9. Simulate model "valve".

Note: here pressDrop is mode variable for valve. Valve also supports one more mode.

- Outlet pressure: value of outlet pressure(outP) is specified.

## 5. Mixer:

### mix1:

This is simulation of mixing of six streams containing mixture of methanol, ethanol water at different composition and different T and P.

Stream 1: P = 101325, T = 353, molar flow = 100, mole fraction = 0.25, 0.25, 0.5

Stream 2: P = 202650, T = 353, molar flow = 100, mole fraction = 0, 0, 1.

Stream 3: P = 126523, T = 353, molar flow = 300, mole fraction = 0.3, 0.3, 0.4.

Stream 4: P = 215365, T = 353, molar flow = 500, mole fraction = 0.25, 0.25, 0.5.

Stream 5: P = 152365, T = 353, molar flow = 400, mole fraction = 0.2, 0.4, 0.4.

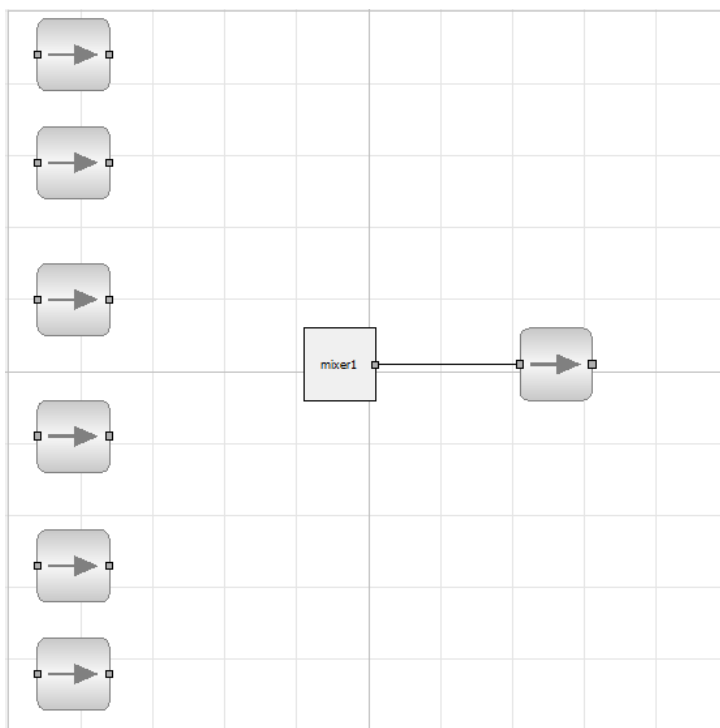
Stream 6: P = 152568, T = 353, molar flow = 200, mole fraction = 0, 1, 0.

1. Create package "mix1".
2. Create composite material stream model "ms" by extending Material\_Stream model and required thermodynamic package.

```
model ms
  extends Simulator.Streams.Material_Stream;
  extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;
end ms;
```
3. Create new model "mix" in "mix1".
4. Now, inside model "mix", create an instance of Chemsep Database and create instances of components to be used.

```
import data = Simulator.Files.Chemsep_Database;
parameter data.Methanol meth;
parameter data.Ethanol eth;
parameter data.Water wat;
```
5. Create an Integer parameter for NOC(number of components) and create a component array.

```
parameter Integer NOC = 3;
parameter data.General_Properties comp[NOC] = {meth, eth,
wat};
```
6. Now open diagram view. Create seven instances of "ms" and name them as "ms1", "ms2", "ms3", "ms4", "ms5", "ms6" and "out1". Create instance of mixer model from Simulator.Unit\_Operations as "mixer1". Connect "mixer1" outlet and inlet of "out1" stream as shown in fig.



7. Switch to text view and specify parameters in "mix1"("NI" is number of mixer inlets, "outPress" is outlet pressure mode), inlets and outlet models. Specify some start values of outlet stream.

```
ms ms1(NOC = NOC, comp = comp) annotation( ...);
ms ms2(NOC = NOC, comp = comp) annotation( ...);
ms ms3(NOC = NOC, comp = comp) annotation( ...);
ms ms4(NOC = NOC, comp = comp) annotation( ...);
ms ms5(NOC = NOC, comp = comp) annotation( ...);
ms ms6(NOC = NOC, comp = comp) annotation( ...);
Unit_Operations.Mixer mixer1(NOC = NOC, NI = 6, comp = comp,
outPress = "Inlet_Average") annotation( ...);
ms out1(NOC = NOC, comp = comp, T(start = 354),
totMolFlo(start = 1600)) annotation( ...);
```

8. Inlet connectors of mixer are instantiated as an array by writing code manually since we need flexibility with number of inputs of mixer. We need to connect them by writing connect equations in equation section.

```
connect(ms1.outlet, mixer1.inlet[1]);
connect(ms2.outlet, mixer1.inlet[2]);
connect(ms3.outlet, mixer1.inlet[3]);
connect(ms4.outlet, mixer1.inlet[4]);
connect(ms5.outlet, mixer1.inlet[5]);
connect(ms6.outlet, mixer1.inlet[6]);
```

9. Specify values of input variables in equation section.

```
ms1.P = 101325;
```

```

ms2.P = 202650;
ms3.P = 126523;
ms4.P = 215365;
ms5.P = 152365;
ms6.P = 152568;
ms1.T = 353;
ms2.T = 353;
ms3.T = 353;
ms4.T = 353;
ms5.T = 353;
ms6.T = 353;
ms1.totMolFlo[1] = 100;
ms2.totMolFlo[1] = 100;
ms3.totMolFlo[1] = 300;
ms4.totMolFlo[1] = 500;
ms5.totMolFlo[1] = 400;
ms6.totMolFlo[1] = 200;
ms1.compMolFrac[1, :] = {0.25, 0.25, 0.5};
ms2.compMolFrac[1, :] = {0, 0, 1};
ms3.compMolFrac[1, :] = {0.3, 0.3, 0.4};
ms4.compMolFrac[1, :] = {0.25, 0.25, 0.5};
ms5.compMolFrac[1, :] = {0.2, 0.4, 0.4};
ms6.compMolFrac[1, :] = {0, 1, 0};

```

#### 10. Simulate model "mix".

## 6. Compound\_Separator:

### comp\_sep1:

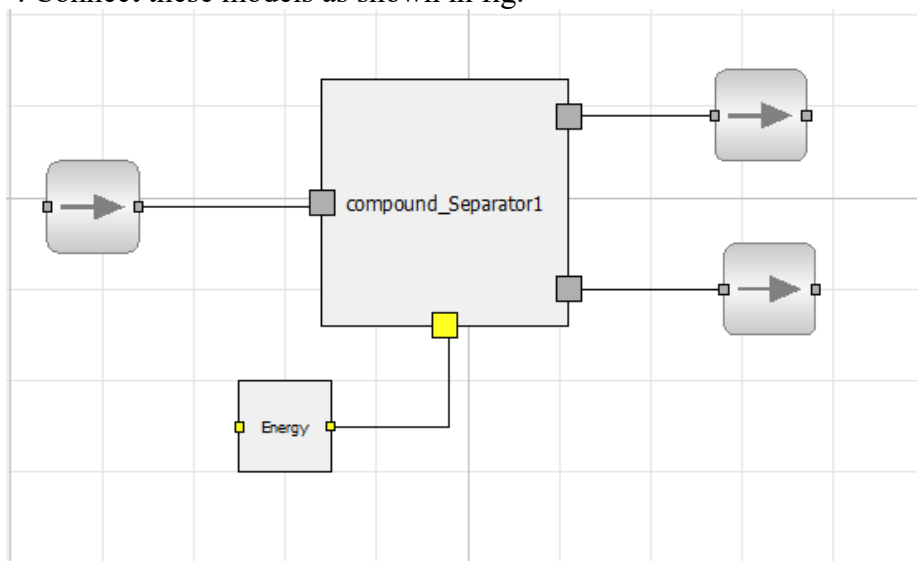
This is simulation of separation of equimolar benzene, toluene mixture in two streams. Where one stream contains 20 moles/s of benzene and 1500 gm/s of toluene. Feed mole flow is 100 mol/s and feed is at 101325 Pa and 298.15 K.

1. Create package comp\_sep1.
2. Create composite material stream model "ms" by extending Material\_Stream model and required thermodynamic package.

```
model ms
  extends Simulator.Streams.Material_Stream;
  extends Simulator.Files.Thermodynamic_Packages.Raoult's_Law;
end ms;
```
3. Create new model "main" in "comp\_sep1".
4. Now, inside model "main", create an instance of Chemsep Database and create instances of components to be used.

```
import data = Simulator.Files.Chemsep_Database;
parameter data.Benzene ben;
parameter data.Toluene tol;
```
5. Create an Integer parameter for NOC(number of components) and create a component array.

```
parameter Integer NOC = 2;
parameter data.General_Properties comp[NOC] = {benz, tol};
```
6. Now open diagram view. Create three instances of "ms" and name them as "Inlet", "Outlet1" and "Outlet2". Create instance of Compound\_Separator model from Simulator.Unit\_Operations as "compound\_Separator1" also instantiate Energy\_Stream form Simulator.Streams as "Energy". Connect these models as shown in fig.





7. Switch to text view and specify parameters in "compound\_Separator1", "Inlet", "Outlet1" and "Outlet2" models.

```
Simulator.Unit_Operations.Compound_Separator
compound_Separator1(NOC = NOC , comp = comp, sepFact =
{"Molar_Flow", "Mass_Flow"}, sepStrm = 2) annotation( ...);
comp_sep1.ms Inlet(NOC = NOC , comp = comp) annotation( ...);
comp_sep1.ms Outlet1(NOC = NOC, comp = comp)
annotation( ...);
comp_sep1.ms Outlet2(NOC = NOC , comp = comp)
annotation( ...);
```

8. Specify values of input variables in equation section.

```
equation
Inlet.P = 101325;
Inlet.T = 298.15;
Inlet.compMolFrac[1, :] = {0.5, 0.5};
Inlet.totMolFlo[1] = 100;
compound_Separator1.sepFactVal = {20, 1500};
```

9. Simulate model "main".

Note: here sepFact are separation factors and sepStrm is stream specified to apply separation factors which are specified in bracket while instantiating component separator. sepFactVal is separation factor values specified in equation section. Separation factors are specified for each component in an array. Following options are available for choosing separation factors.

- "Molar\_Flow": Absolute molar flow value of component in stream in mol/s
- "Mass\_Flow": Absolute mass flow value of component in stream in gm/s
- "Inlet\_Molar\_Flow\_Percent": Molar flow percent of inlet mole flow of respective component
- "Inlet\_Mass\_Flow\_Percent": Mass flow percent of inlet mole flow of respective component

## 7. Shortcut\_Column:

### shortcut1:

This is simulation of column separating euqimolar mixture of benzene toluene. Benzene mole fraction in distillate is 0.01 and toluene mole fraction in bottoms is 0.01. Reflux ratio is 2. Column pressure is 101325.

1. Create package shortcut1.
2. Create composite material stream model ms by extending Material\_Stream model and required thermodynamic package.

```
model ms
  extends Simulator.Streams.Material_Stream;
  extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;
end ms;
```

3. For shortcut column we need to extend thermodynamics. Create a model 'Shortcut' inside package 'shortcut1'. Extend Shortcut\_Column model from Simulator.Unit\_Operations and also extend required thermodynamic package.

```
model Shortcut
  extends Simulator.Unit_Operations.Shortcut_Column;
  extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;
end Shortcut;
```

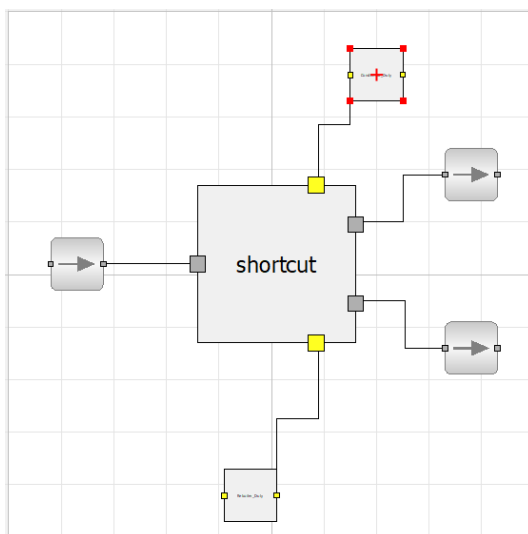
4. Create new model "main" in "shortcut1".
5. Now, inside model "main", create an instance of Chemsep Database and create instances of components to be used.

```
import data = Simulator.Files.Chemsep_Database;
parameter data.Benzene ben;
parameter data.Toluene tol;
```

6. Create an Integer parameter for NOC(number of components) and create a component array.

```
parameter Integer NOC = 2;
parameter data.General_Properties comp[NOC] = {benz, tol};
```

7. Now open diagram view. Create three instances of "ms" as "feed", "distillate" and "bottoms". Instantiate model "Shortcut" as "shortcut". Instantiate Energy\_Stream twice as "Condensor\_Duty" and "Reboiler\_Duty". Connect them as shown in fig.



8. Switch to text view and specify parameters in compound\_Separator1, Inlet, Outlet1 and Outlet2 models. CondType is condensor type – "Partial" and "Total". HKey and LKey are heavy key and light key respectively.

```
shortcut1.Shortcut shortcut(NOC = NOC, comp = comp, condType
= "Partial", HKey = 2, LKey = 1, mixMolFrac(start = {{0.5,
0.5}}, {0.01, 0.99}}, {0.99, 0.01})), minN(start = 9.1859),
theta(start = 1, min = 1), minR(start = 2, min = 0))
annotation( ...);
shortcut1.ms feed(NOC = NOC, comp = comp, compMolFrac(start =
{{0.5, 0.5}}, {0.34, 0.65}}, {0.56, 0.44}))) annotation( ...);
shortcut1.ms bottoms(NOC = NOC, comp = comp, T(start =
383.08), compMolFrac(start = {{0.01, 0.99}}, {0.01, 0.99}}, {0, 0})))
annotation( ...);
shortcut1.ms distillate(NOC = NOC, comp = comp, T(start =
353.83), compMolFrac(start = {{0.99, 0.01}}, {0.99, 0.01}}, {0, 0})))
annotation( ...);
```

9. Specify values of input variables in equation section.

```
equation
feed.P = 101325;
feed.T = 370;
feed.compMolFrac[1, :] = {0.5, 0.5};
feed.totMolFlo[1] = 100;
shortcut.shortP[2] = 101325;
shortcut.shortP[3] = 101325;
shortcut.mixMolFrac[2, shortcut.LKey] = 0.01;
shortcut.mixMolFrac[3, shortcut.HKey] = 0.01;
shortcut.actR = 2;
```

10. Simulate model "main".

## 8. Flash:

### flash:

This is simulation of flashing of equimolar mixture of benzene and toluene at 101325 Pa and 368 K.

1. Create package "flash".
2. Create composite material stream model ms by extending Material\_Stream model and required thermodynamic package.

```
model ms
  extends Simulator.Streams.Material_Stream;
  extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;
end ms;
```

3. For flash also we need to extend thermodynamics. Create a model 'fls' inside package 'flash'. Extend Flash model from Simulator.Unit\_Operations and also extend required thermodynamic package.

```
model fls
  extends Simulator.Unit_Operations.Flash;
  extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;
end fls;
```

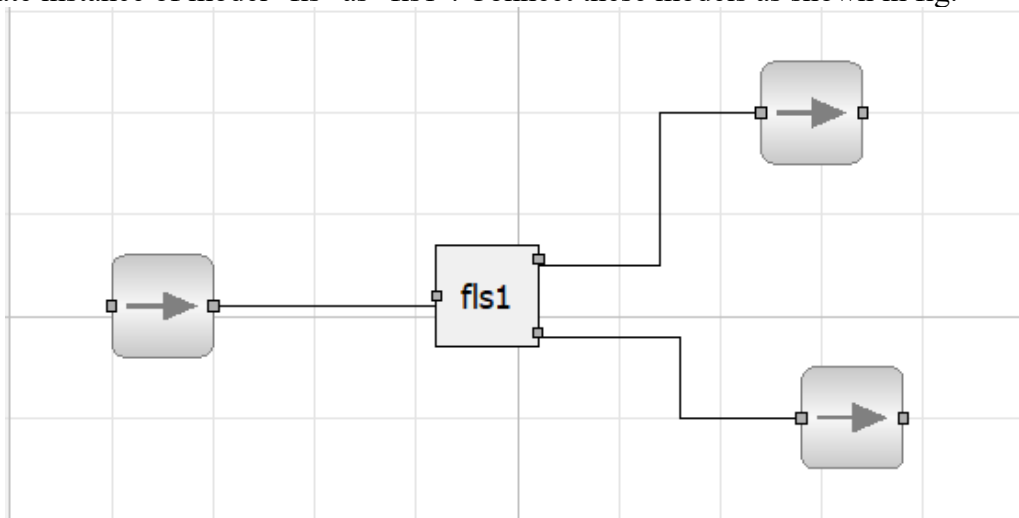
4. Create new model "test" in "flash".
5. Now, inside model "test", create an instance of Chemsep Database and create instances of components to be used.

```
import data = Simulator.Files.Chemsep_Database;
parameter data.Benzene ben;
parameter data.Toluene tol;
```

6. Create an Integer parameter for NOC(number of components) and create a component array.

```
parameter Integer NOC = 2;
parameter data.General_Properties comp[NOC] = {benz, tol};
```

7. Now switch to diagram view. Create three instances of ms as "inlet", "outlet1" and "outlet2". Create instance of model "fls" as "fls1". Connect these models as shown in fig.



8. Switch to text view and specify parameters in "fls1", "inlet", "outlet1" and "outlet2" models.

```
flash.ms inlet(NOC = NOC, comp = comp) annotation( ...);  
flash.fls fls1(NOC = NOC, comp = comp) annotation( ...);  
flash.ms outlet1(NOC = NOC, comp = comp) annotation( ...);  
flash.ms outlet2(NOC = NOC, comp = comp) annotation( ...);
```

9. Specify values of input variables in equation section.

```
equation  
  inlet.P = 101325;  
  inlet.T = 368;  
  inlet.compMolFrac[1, :] = {0.5, 0.5};  
  inlet.totMolFlo[1] = 100;
```

10. Simulate model "test".

## 9. Splitter:

### split:

This is simulation of stream splitter where splitter splits the stream of equimolar stream of benzene and toluene having molar flow 100 mol/s. Molar flows of outlet streams are 20 mol/s and 80 mol/s.

1. Create a package "split".
2. Create composite material stream model "ms" by extending Material\_Stream model and required thermodynamic package.

```
model ms
  extends Simulator.Streams.Material_Stream;
  extends Simulator.Files.Thermodynamic_Packages.Raoult_Law;
end ms;
```

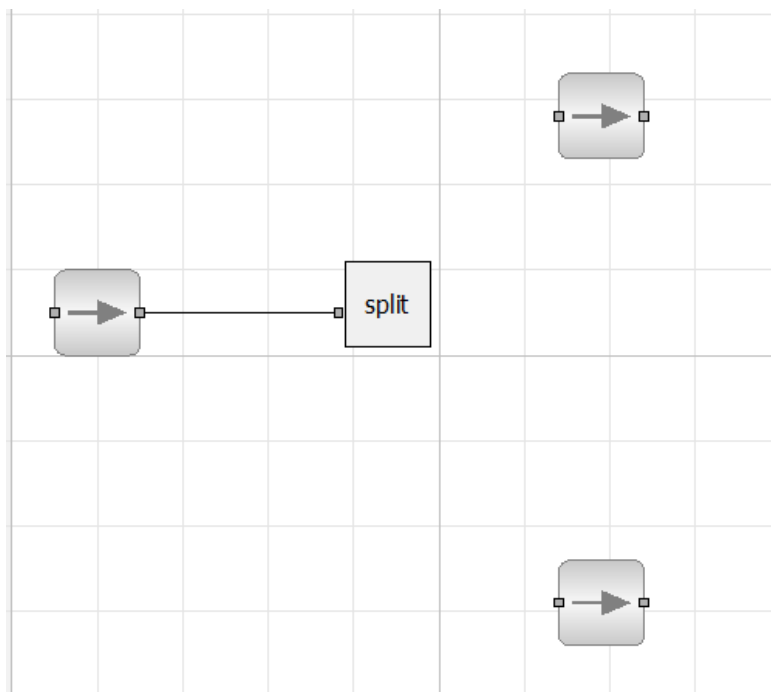
3. Create new model "main" in "split".
4. Now, inside model "main", create an instance of Chemsep Database and create instances of components to be used.

```
import data = Simulator.Files.Chemsep_Database;
parameter data.Benzene ben;
parameter data.Toluene tol;
```

5. Create an Integer parameter for NOC(number of components) and create a component array.

```
parameter Integer NOC = 2;
parameter data.General_Properties comp[NOC] = {benz, tol};
```

6. Now switch to diagram view. Create three instances of ms as "inlet", "outlet1" and "outlet2". Create instance of model "Splitter" from Simulator.Unit\_Operations as "split". Connect these models as shown in fig.



7. Outlet connectors of splitter are instantiated by writing code manually since we need flexibility of having any number of outputs in splitter. Switch to text view and write connect equations for outlets in equation section.

```
connect(split.outlet[1], outlet1.inlet);  
connect(split.outlet[2], outlet2.inlet);
```

8. Specify parameters of "inlet", "outlet1", "outlet2" and "split" in bracket. NO are number of outputs and calcType is calculation type in splitter.

```
ms inlet(NOC = NOC, comp = comp) annotation( ...);  
ms outlet1(NOC = NOC, comp = comp) annotation( ...);  
ms outlet2(NOC = NOC, comp = comp) annotation( ...);  
Simulator.Unit_Operations.Splitter split(NOC = NOC, comp =  
comp, NO = 2, calcType = "Molar_Flow") annotation( ...);
```

9. Specify values of input variables in equation section.

```
inlet.P = 101325;  
inlet.T = 300;  
inlet.compMolFrac[1, :] = {0.5, 0.5};  
inlet.totMolFlo[1] = 100;  
split.specVal = {20, 80};
```

10. Simulate model "main".

Note: Split\_Ratio, Mass\_Flow and Molar\_Flow are available as calcType in splitter.

## 10. Centrifugal\_Pump:

### pump:

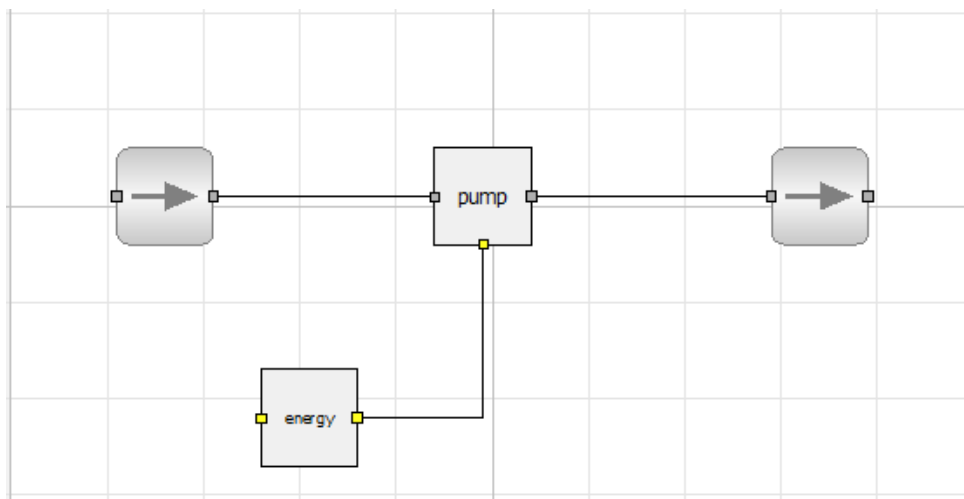
This is simulation of centrifugal pump where equimolar mixture of benzene and toluene at 101325 Pa and 300 K is pumped where pressure is increased by 101325 Pa.

1. Create a package "pump".
2. Create composite material stream model "ms" by extending Material\_Stream model and required thermodynamic package.

```
model ms
  extends Simulator.Streams.Material_Stream;
  extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;
end ms;
```
3. Create new model "main" in "pump".
4. Now, inside model "main", create an instance of Chemsep Database and create instances of components to be used.

```
import data = Simulator.Files.Chemsep_Database;
parameter data.Benzene ben;
parameter data.Toluene tol;
```
5. Create an Integer parameter for NOC(number of components) and create a component array.

```
parameter Integer NOC = 2;
parameter data.General_Properties comp[NOC] = {benz, tol};
```
6. Now switch to diagram view. Create two instances of ms as "inlet" and "outlet". Create instance of model "Centrifugal\_Pump" from Simulator.Unit\_Operations as "pump". Create one instance of "Energy\_Stream" from Simulator.Streams as "energy". Connect these models as shown in fig.





7. Specify parameters of "inlet", "outlet" and "pump" in bracket. eff is efficiency of pump.

```
pump.ms inlet(NOC = 2, comp = {benz, tol}) annotation( ...);  
Unit_Operations.Centrifugal_Pump pump(comp = {benz, tol}, NOC  
= 2, eff = 0.75) annotation( ...);  
pump.ms outlet(NOC = 2, comp = {benz, tol}, T(start =  
300.089), compMolFrac(start = {{0.5, 0.5}}, {0.5, 0.5}, {0, 0})),  
totMolFlo(start = 100)) annotation( ...);
```

8. Specify values of input variables in equation section.

```
inlet.totMolFlo[1] = 100;  
inlet.compMolFrac[1, :] = {0.5, 0.5};  
inlet.P = 101325;  
inlet.T = 300;  
pump.pressInc = 101325;
```

9. Simulate model "main".

Note: pressInc is mode variable for centrifugal pump. Centrifugal pump also supports following modes.

- outlet Pressure: Value of outlet pressure(outP) in Pa is specified.
- power required: Value of power required(reqPow) in Watts is specified.
- Energy Stream: Value of energy flow in energy stream (Energy\_Stream\_Name.enFlo) in Watts is specified.

## 11. Adiabatic\_Compressor:

### adia\_comp1:

This is simulation of adiabatic compression of equimolar benzene and toluene mixture at 202650 Pa and 372 K. Pressure increase is 10000 Pa.

1. Create a package adia\_comp1.
2. Create composite material stream model "ms" by extending Material\_Stream model and required thermodynamic package.

```
model ms
  extends Simulator.Streams.Material_Stream;
  extends Simulator.Files.Thermodynamic_Packages.Raoultz_Law;
end ms;
```

3. For adiabatic compressor also we need to extend thermodynamic package. Create model "compres" in adia\_comp1. Extend Adiabatic\_Compressor from Simulator.Unit\_Operations also extend required thermodynamic package from Simulator.Files.Themoydynamic\_Packages.

```
model compres
  extends Unit_Operations.Adiabatic_Compressor;
  extends Files.Thermodynamic_Packages.Raoultz_Law;
end compres;
```

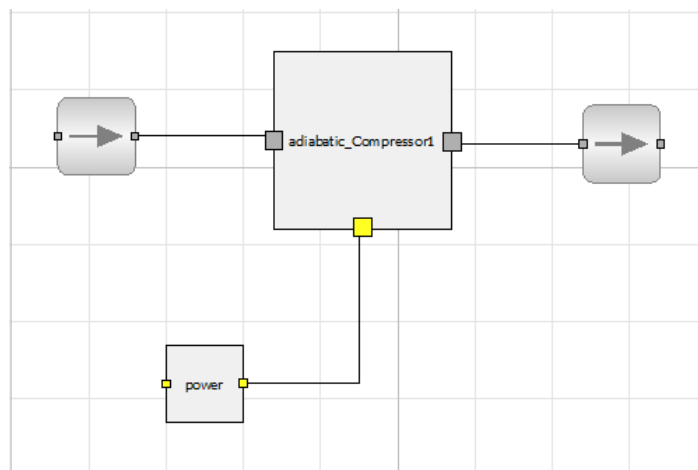
4. Create model "main" inside "adia\_comp1".
5. Now, inside model "main", create an instance of Chemsep Database and create instances of components to be used.

```
import data = Simulator.Files.Chemsep_Database;
parameter data.Benzene ben;
parameter data.Toluene tol;
```

6. Create an Integer parameter for NOC(number of components) and create a component array.

```
parameter Integer NOC = 2;
parameter data.General_Properties comp[NOC] = {benz, tol};
```

7. Now switch to diagram view. Create two instances of ms as "inlet" and "outlet". Create instance of model "compres" as "adiabatic\_Compressor1". Create one instance of "Energy\_Stream" from Simulator.Streams as "power". Connect these models as shown in fig.



8. Specify parameters of "inlet", "outlet" and "adiabatic\_Compressor1" in bracket. eff is efficiency of compressor.

```
compres adiabatic_Compressor1(NOC = NOC, comp = comp, eff =  
0.75) annotation( ...);  
adia_comp1.ms inlet(NOC = NOC, comp = comp) annotation( ...);  
ms outlet(NOC = NOC, comp = comp, T(start = 374)) annotation(  
...);
```

9. Specify input variables in equation section.

```
inlet.compMolFrac[1, :] = {0.5, 0.5};  
inlet.P = 202650;  
inlet.T = 372;  
inlet.totMolFlo[1] = 100;  
adiabatic_Compressor1.pressInc = 10000;
```

10. Simulate model "main".

Note: pressInc is mode variable for adiabatic compressor. Other modes supported by adiabatic compressor are as follows.

- outlet Pressure: Value of outlet pressure(outP) in Pa is specified.
- power required: Value of power required(reqPow) in Watts is specified

## 12. Adiabatic\_Expander:

### adia\_exp1:

This is simulation of adiabatic expansion of equimolar benzene and toluene mixture at 131325 Pa and 372 K. Pressure drop is 10000 Pa.

1. Create a package adia\_exp1.
2. Create composite material stream model "ms" by extending Material\_Stream model and required thermodynamic package.

```
model ms
  extends Simulator.Streams.Material_Stream;
  extends Simulator.Files.Thermodynamic_Packages.Raoults_Law;
end ms;
```

3. For adiabatic expander, we need to extend thermodynamic package. Create model "exp" in adia\_exp1. Extend Adiabatic\_Expander from Simulator.Unit\_Operations also extend required thermodynamic package from Simulator.Files.Themoydnnamic\_Packages.

```
model exp
  extends Unit_Operations.Adiabatic_Expander;
  extends Files.Thermodynamic_Packages.Raoults_Law;
end exp;
```

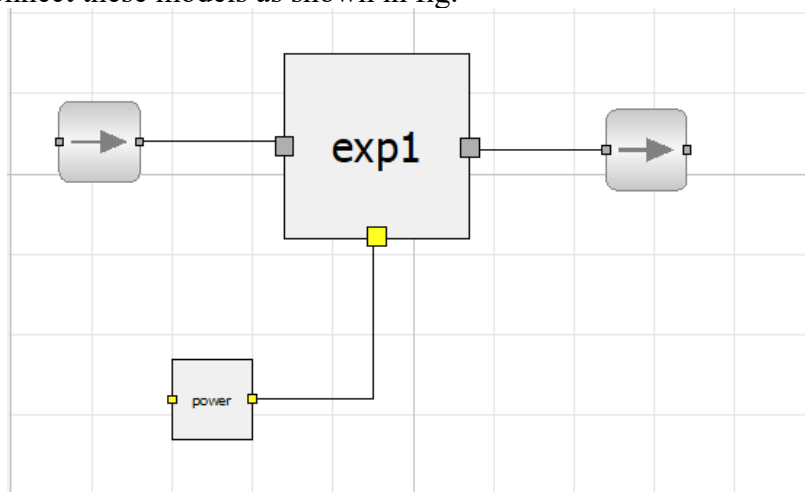
4. Create model "main" inside "adia\_exp1".
5. Now, inside model "main", create an instance of Chemsep Database and create instances of components to be used.

```
import data = Simulator.Files.Chemsep_Database;
parameter data.Benzene ben;
parameter data.Toluene tol;
```

6. Create an Integer parameter for NOC(number of components) and create a component array.

```
parameter Integer NOC = 2;
parameter data.General_Properties comp[NOC] = {benz, tol};
```

7. Now switch to diagram view. Create two instances of ms as "inlet" and "outlet". Create instance of model "exp" as "exp1". Create one instance of "Energy\_Stream" from Simulator.Streams as "power". Connect these models as shown in fig.



8. Specify parameters of "inlet", "outlet" and "adiabatic\_Expander1" in bracket. eff is efficiency of compressor.

```
exp expl(NOC = NOC, comp = comp, eff = 0.75)
annotation( ...);
adia_comp1.ms inlet(NOC = NOC, comp = comp)
annotation( ...);
ms outlet(NOC = NOC, comp = comp, T(start = 374))
annotation( ...);
```

9. Specify input variables in equation section.

```
inlet.compMolFrac[1, :] = {0.5, 0.5};
inlet.P = 131325;
inlet.T = 372;
inlet.totMolFlo[1] = 100;
expl.pressDrop = 10000;
```

10. Simulate model "main".

Note: pressDrop is mode variable for adiabatic compressor. Other modes supported by adiabatic compressor are as follows.

- outlet Pressure: Value of outlet pressure(outP) in Pa is specified.
- power generated: Value of power generated(genPow) in Watts is specified