Report 2 Demos:

Demo1- Filling multiple unbounded output reservoirs

In[ ]:=
```
(* create origin seq, set its height, put an inputR and a
 infPump on it. Then set the initial flowRate using the pump *)
createSeq[Null, Null];
setHeight["seq1", Quantity[1, "Meters"]];
mapNewInfInputReservoir ["seq1",
  Quantity[0.5, "Meters"], 0, globalConstants["densityWater"]];
addPipes["seq1", 1, Quantity[6, "Centimeters"]];
addInfPump["seq1", Quantity[0.5, "Meters"^3 / "Seconds"]];
```

In[ ]:=
```
(* create a split. connect it. then put
 output reservoirs on the resulting sequences *)
createSpl[Null, {1, 1, 1}, Null];
setStart["spl2", "seq1"];
setEnd["seq1", "spl2"];
```

In[ ]:=
```
(* add pipes to the sequences, except for seq6 *)
addPipes["seq3", 1, Quantity[5, "Centimeters"]];
addPipes["seq4", 1, Quantity[5, "Centimeters"]];
addPipes["seq5", 1, Quantity[5, "Centimeters"]];
```

In[ ]:=
```
(* put an unbounded output reservoir on seq2 *)
mapNewUnboundedOutputReservoir ["seq3", Quantity[0.5, "Meters"], 0, 0]
mapNewUnboundedOutputReservoir ["seq4", Quantity[0.5, "Meters"], 0, 0]
mapNewUnboundedOutputReservoir ["seq5", Quantity[0.5, "Meters"], 0, 0]
```

In[ ]:=
```
(* process the flowRates, avgDiameters, and avgVelocities *)
processSysMain[];

(* advance by 2 seconds, fill the outputs. Look at the end references for seq3,
4, and 5. The water volume fills! *)
```

In[ ]:=
```
fillUnOutputs[Quantity[2, "Seconds"]];
```

Demo2- Generalized pump/reservoir addition and modification

```
In[ ]:= (* create origin seq, set its height, put the reservoir and pump on it *)
createSeq[Null, Null];
setHeight["seq1", Quantity[1, "Meters"]];
mapNewGenInputReservoir ["seq1", Quantity[0.5, "Meters"], Quantity[0.25, "Meters"],
  0, Quantity[1, "Meters"^3], 0, globalConstants["densityWater"], 0];
addPipes["seq1", 1, Quantity[0.6, "Centimeters"]];
addGenPump["seq1", 0.75, Quantity[100, "Watts"], Quantity[5, "Meters"]];
```

```
In[ ]:= (* add another sequence, put an output reservoir on it *)
createSeq[Null, "seq1"];
setEnd["seq1", "seq2"];
mapNewGenOutputReservoir ["seq2", Quantity[0.5 , "Meters"],
  Quantity[0.25, "Meters"], 0, Quantity[2, "Meters"^3], 0, Quantity[3, "Meters"^3], 0];

(* modify the pump's head value, supplied electricity, and efficiency *)
modifyGenPump["seq1", 0.80, Quantity[1000, "Watts"], Quantity[2, "Meters"]];
(* The pump's flow rate out has changed because you modified the efficiency,
electricity, and head value *)
```

Demo3- Required head value determination

```
In[ ]:= (* create origin seq, set its height, put the reservoir and pump on it *)
createSeq[Null, Null];
setHeight["seq1", Quantity[2, "Meters"]];
mapNewGenInputReservoir ["seq1", Quantity[0.5, "Meters"], Quantity[0.25, "Meters"],
  Quantity[0, "Pascals"], Quantity[1, "Meters"^3], Quantity[0, "Meters"^3],
  globalConstants["densityWater"], Quantity[0, "Kilograms"/"Meters"^3]];
addPipes["seq1", 1, Quantity[0.6, "Centimeters"]];
addGenPump["seq1", 0.75, Quantity[100, "Watts"], Quantity[5, "Meters"]];
```

```
In[ ]:= (* add a split, put output reservoirs on all 3 of its derivative sequences *)
createSpl["seq1", {1, 1, 1}, Null];
setEnd["seq1", "spl2"];

mapNewGenOutputReservoir ["seq3", Quantity[0.5 , "Meters"], Quantity[0.25, "Meters"],
  Quantity[0, "Pascals"], Quantity[0, "Meters"^3], Quantity[0, "Meters"^3],
  Quantity[0, "Meters"^3], Quantity[0, "Kilograms" / "Meters"^3]];
mapNewGenOutputReservoir ["seq4", Quantity[0.75, "Meters"], Quantity[0.5, "Meters"],
  Quantity[2000, "Pascals"], Quantity[0, "Meters"^3], Quantity[0, "Meters"^3],
  Quantity[0, "Meters"^3], Quantity[2000, "Kilograms" / "Meters"^3]];
mapNewGenOutputReservoir ["seq5", Quantity[0.4, "Meters"], Quantity[0.4, "Meters"],
  Quantity[1500, "Pascals"], Quantity[0, "Meters"^3], Quantity[0, "Meters"^3],
  Quantity[0, "Meters"^3], Quantity[500, "Kilograms" / "Meters"^3]];
(*NOTE- in the previous example, 0's were used in some of the parameter
 spots. We can't do that anymore because now we're processing that
 information. The 0 with its respective quantity has to be entered. Otherwise,
the unit mismatches will cause issues *)
```

```
In[ ]:= (* set heights for the sequences*)
setHeight["seq3", Quantity[4, "Meters"]];
setHeight["seq4", Quantity[3, "Meters"]];
setHeight["seq5", Quantity[1, "Meters"]];
```

```
In[ ]:= (* Uses the input and output reservoir data to figure out what
 the head value (capacity to do work) should be for the pump *)
(* The height of the output reservoirs affects this value,
along with the pressure exerted on the reservoirs by
 pressure gauges and different fluids floating above the water *)
findHeadVal []
```

```
Out[ ]= 2.20408 m
```

Demo4- Generalized water transfer

```
In[ ]:= (* create origin seq, set its height, put the reservoir and pump on it *)
createSeq[Null, Null];
setHeight["seq1", Quantity[2, "Meters"]];
mapNewGenInputReservoir ["seq1", Quantity[0.5, "Meters"], Quantity[0.25, "Meters"],
  Quantity[0, "Pascals"], Quantity[1, "Meters"^3], Quantity[0, "Meters"^3],
  globalConstants["densityWater"], Quantity[0, "Kilograms" / "Meters"^3]];
addPipes["seq1", 1, Quantity[0.6, "Centimeters"]];
addGenPump["seq1", 0.75, Quantity[10 000, "Watts"], Quantity[2, "Meters"]];
```

```
In[ ]:= (* add a split, put output reservoirs on all 3 of its derivative sequences *)
createSpl["seq1", {1, 1, 1}, Null];
setEnd["seq1", "spl2"];

mapNewGenOutputReservoir ["seq3", Quantity[0.5 , "Meters"], Quantity[0.25, "Meters"],
  Quantity[0, "Pascals"], Quantity[1, "Meters"^3], Quantity[0, "Meters"^3],
  Quantity[3, "Meters"^3], Quantity[0, "Kilograms" / "Meters"^3]];
mapNewGenOutputReservoir ["seq4", Quantity[0.75, "Meters"], Quantity[0.5, "Meters"],
  Quantity[2000, "Pascals"], Quantity[2, "Meters"^3], Quantity[0, "Meters"^3],
  Quantity[4, "Meters"^3], Quantity[2000, "Kilograms" / "Meters"^3]];
mapNewGenOutputReservoir ["seq5", Quantity[0.4, "Meters"], Quantity[0.4, "Meters"],
  Quantity[1500, "Pascals"], Quantity[0, "Meters"^3], Quantity[0, "Meters"^3],
  Quantity[5, "Meters"^3], Quantity[500, "Kilograms" / "Meters"^3]];
(*NOTE- in the previous example, 0's were used in some of the parameter
 spots. We can't do that anymore because now we're processing that
 information. The 0 with its respective quantity has to be entered. Otherwise,
the unit mismatches will cause issues *)

In[ ]:= (* set heights for the sequences*)
setHeight["seq3", Quantity[4, "Meters"]];
setHeight["seq4", Quantity[3, "Meters"]];
setHeight["seq5", Quantity[1, "Meters"]];

In[ ]:= (* add pipes *)
addPipes["seq3", 1, Quantity[5, "Centimeters"]];
addPipes["seq4", 1, Quantity[5, "Centimeters"]];
addPipes["seq5", 1, Quantity[5, "Centimeters"]];

In[ ]:= (* process system attributes *)
processSysMain []

In[ ]:= (* Is the head value of the pump high enough? *)
headValCheck []
(* No! Ok... let's change it *)

Out[ ]= False
```

```
In[ ]:= (* Keep the original pump data,
 except for the head value. Change it from 2m to 10m. Then run a
  check. The head value is now high enough, so we can pump water! *)
modifyGenPump["seq1", 0.75, Quantity[10000, "Watts"], Quantity[10, "Meters"]];
headValCheck[]
processSysMain[]
(* NOTE- flowRate decreases in seq1 after raising
   the head value because the supplied electricity stayed
   the same. If a pump has a higher capacity to do work,
more electricity must be supplied to it. This is why it's recommended
 to match the system's head value. Using the modifyGenPump command can
 simulate stepping or stepping down the head value or supplying more power. *)
```

```
Out[ ]= True
```

```
(* transfer water for 1 minutes *)
(* the output reservoir information can
 be found in the end pointer for seq3, 4, and 5 *)
(* Before fillling water:
   the first outputR had 1m^3 of water out of 3m^3
    the second outputR had 2m^3 of water out of 4m^3
    the third outputR had 0m^3 of water out of 5m^3 *)
fillGenOutputs[Quantity[60, "Seconds"]];
(* the first outputR has 2.53061 m^3 of water out of 3m^3
 the second outputR has 3.53061 m^3 of water out of 4m^3
 the third outputR has 1.53061 m^3 of water out of 5m^3  *)
```

```
In[ ]:= (* transfer water for another minute *)
fillGenOutputs[Quantity[60, "Seconds"]];
(* the first and second outputR filled
 up and the third  has 3.06122 m^3 out of 5m^3 *)
```

Demo5- Change gauge pressure! + its effects

```
In[ ]:= (* create origin seq, set its height, put the reservoir and pump on it *)
createSeq[Null, Null];
setHeight["seq1", Quantity[2, "Meters"]];
mapNewGenInputReservoir ["seq1", Quantity[0.5, "Meters"], Quantity[0.25, "Meters"],
  Quantity[0, "Pascals"], Quantity[1, "Meters"^3], Quantity[0, "Meters"^3],
  globalConstants["densityWater"], Quantity[0, "Kilograms" / "Meters"^3]];
addPipes["seq1", 1, Quantity[0.6, "Centimeters"]];
addGenPump["seq1", 0.75, Quantity[10000, "Watts"], Quantity[2, "Meters"]];
```

```
In[ ]:= (* add a split, put output reservoirs on all 3 of its derivative sequences *)
createSpl["seq1", {1, 1, 1}, Null];
setEnd["seq1", "spl2"];

mapNewGenOutputReservoir ["seq3", Quantity[0.5 , "Meters"], Quantity[0.25, "Meters"],
  Quantity[0, "Pascals"], Quantity[1, "Meters"^3], Quantity[0, "Meters"^3],
  Quantity[3, "Meters"^3], Quantity[0, "Kilograms" / "Meters"^3]];
mapNewGenOutputReservoir ["seq4", Quantity[0.75, "Meters"], Quantity[0.5, "Meters"],
  Quantity[2000, "Pascals"], Quantity[2, "Meters"^3], Quantity[0, "Meters"^3],
  Quantity[4, "Meters"^3], Quantity[2000, "Kilograms" / "Meters"^3]];
mapNewGenOutputReservoir ["seq5", Quantity[0.4, "Meters"], Quantity[0.4, "Meters"],
  Quantity[1500, "Pascals"], Quantity[0, "Meters"^3], Quantity[0, "Meters"^3],
  Quantity[5, "Meters"^3], Quantity[500, "Kilograms" / "Meters"^3]];

In[ ]:= (* set heights for the sequences *)
setHeight["seq3", Quantity[4, "Meters"]];
setHeight["seq4", Quantity[3, "Meters"]];
setHeight["seq5", Quantity[1, "Meters"]];

In[ ]:= (* add pipes *)
addPipes["seq3", 1, Quantity[5, "Centimeters"]];
addPipes["seq4", 1, Quantity[5, "Centimeters"]];
addPipes["seq5", 1, Quantity[5, "Centimeters"]];

In[ ]:= (* process system attributes *)
processSysMain []

In[ ]:= (* required head value before gauge pressure change *)
findHeadVal []

Out[ ]= 2.20408 m

In[ ]:= (* change gauge pressure on input *)
changeGaugePressureGen ["seq1", Quantity[5000, "Pascals"]];
(* required head value after gauge pressure change *)
(* The required head value decreased because more
 pressure on the fluid in the input reservoir helps it travel *)
findHeadVal []

Out[ ]= 2 m
```

In[ ]:=
```
(* change gauge pressure on seq5's outputReservoir *)
(* the head value needs to be higher because of resistance from the outputR *)
changeGaugePressureGen ["seq5", Quantity[80 000, "Pascals"]];
findHeadVal []
```

Out[ ]= 6.85306 m

Demo6- View testing! Uses controller functions!

```
(* Instructions : run the test 1 cell, then go down to selected,
initialize it to Null, then run the dynamic readout for selected.
  Now you can run the displayNetwork command
 and it will present you with a view for test 1.
     Run the other tests, execute the displayNetwork for each.
      Clicking the sequence buttons changes the selected sequence
*)
```

In[ ]:=
```
(* test 1 *)
resetMasterPathList [];
createSeq[Null, Null];
createSeq[Null, "seq1"];
createSpl["seq2", {3, 1, 2}, {False, False, True, False, True}];
setEnd["seq1", "seq2"];
setEnd["seq2", "spl3"];
setHeight["seq1", Quantity[10, "Meters"]];
setHeight["seq2", Quantity[11, "Meters"]];
setHeight["seq4", Quantity[20, "Meters"]];
setHeight["seq5", Quantity[18, "Meters"]];
setHeight["seq6", Quantity[16, "Meters"]];
setHeight["seq7", Quantity[11, "Meters"]];
setHeight["seq8", Quantity[9, "Meters"]];
setHeight["seq9", Quantity[7, "Meters"]];
setHeight["seq11", Quantity[14, "Meters"]];
setHeight["seq13", Quantity[8, "Meters"]];
mapNewGenInputReservoir ["seq1", 1, 1, 1, 1, 1, 1, 1];
addGenPump["seq1", 1, 1, 1];
mapNewGenOutputReservoir ["seq4", 1, 1, 1, 1, 1, 1, 1];
mapNewGenOutputReservoir ["seq5", 1, 1, 1, 1, 1, 1, 1];
mapNewGenOutputReservoir ["seq11", 1, 1, 1, 1, 1, 1, 1];
mapNewGenOutputReservoir ["seq13", 1, 1, 1, 1, 1, 1, 1];
(* The error thrown below doesn't matter. We only care
 about the structure when testing to make sure the view works *)
```

Quantity : $\frac{\text{Seconds}^2}{\text{Meters}}$ and $\frac{\text{Meters}^3}{\text{Seconds}}$ are incompatible units

In[ ]:=
```
(* test 2*)
resetMasterPathList [];
createSeq[Null, Null];
createSpl["seq1", {1, 1, 1}, {True}];
setHeight["seq1", Quantity[0, "Meters"]];
setHeight["seq3", Quantity[1, "Meters"]];
setHeight["seq4", Quantity[0, "Meters"]];
setHeight["seq5", Quantity[-1, "Meters"]];
setHeight["seq7", Quantity[5, "Meters"]];
createSpl["seq7", {0, 0, 4}, {False, True, True}];
setHeight["seq9", Quantity[4, "Meters"]];
setHeight["seq10", Quantity[3, "Meters"]];
setHeight["seq11", Quantity[2, "Meters"]];
setHeight["seq12", Quantity[1, "Meters"]];
setHeight["seq14", Quantity[0, "Meters"]];
```

In[ ]:=
```
(* test 3, shows pump and output *)
resetMasterPathList [];
createSeq[Null, Null];
createSpl["seq1", {1, 0, 2}, {True, True}];
createSeq[Null, "seq7"];
createSpl["seq8", {2, 1, 1}, {False, False, True}]
setEnd["seq1", "spl2"];
setEnd["seq7", "seq8"];
setEnd["seq8", "spl9"];
setHeight["seq1", Quantity[10, "Meters"]];
setHeight["seq3", Quantity[20, "Meters"]];
setHeight["seq4", Quantity[8, "Meters"]];
setHeight["seq5", Quantity[6, "Meters"]];
setHeight["seq7", Quantity[8, "Meters"]];
setHeight["seq8", Quantity[10, "Meters"]];
setHeight["seq10", Quantity[15, "Meters"]];
setHeight["seq11", Quantity[13, "Meters"]];
setHeight["seq12", Quantity[10, "Meters"]];
setHeight["seq13", Quantity[7, "Meters"]];
setHeight["seq15", Quantity[5, "Meters"]];
mapNewGenOutputReservoir ["seq10", 1, 1, 1, 1, 1, 1, 1];
mapNewGenInputReservoir ["seq1", 1, 1, 1, 1, 1, 1, 1];
addInfPump["seq1", 1];
```

In[ ]:=
```
(* test 4, basic input→output *)
resetMasterPathList [];
createSeq[Null, Null];
setHeight["seq1", Quantity[5, "Meters"]];
mapNewInfInputReservoir ["seq1", 1, 1, 1];
addInfPump["seq1", 1];
createSeq[Null, "seq1"];
setHeight["seq2", Quantity[2, "Meters"]];
mapNewUnboundedOutputReservoir ["seq2", 1, 1, 1];
```

```
(* set the currently selected sequence to Null to initialize it *)
selected = Null;

(* see the selected sequence update when you click on the buttons *)
Dynamic[selected]
```

Out[ ]=
```
selected
```

In[ ]:=
```
(* display is modular: units, fontSize, fontType, line thickness *)
(* warning: changing the parameters might produce bad displays
   example: the display is proportional, so using "Centimeters" as the
  units will require lots of scrolling when the view is displayed *)
(* don't mess with the black options. building the network and modifying it
  should be done through the UI because it has an organized system setup *)
displayNetwork["Meters", 16, "Comic Sans MS", 0.01]
```