# Tests

## Scenario 1

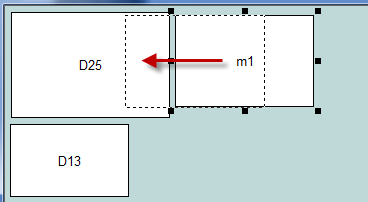
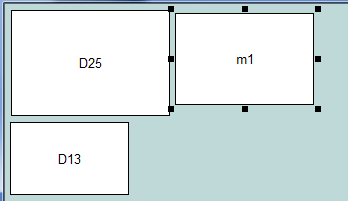
Basic tests. Note that the algorithm is tuned to move the minimum needed, so if moving Y is less distance than moving X then a vertical movement will be applied.

{'type':'node', 'id':'D25', 'x':6, 'y':7, 'width':159, 'height':106}

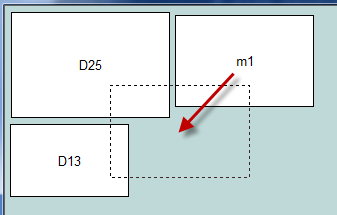
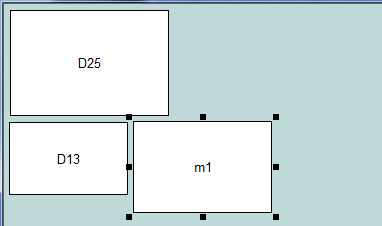
{'type':'node', 'id':'D13', 'x':6, 'y':119, 'width':119, 'height':73}

{'type':'node', 'id':'m1', 'x':170, 'y':9, 'width':139, 'height':92}

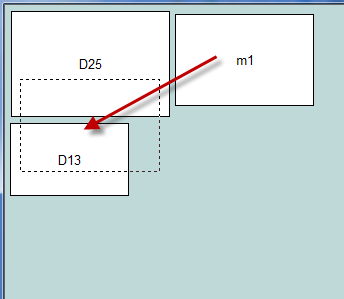
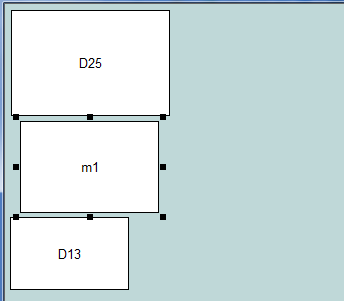
### test1\_1MoveLeftPushedBackHorizontally01

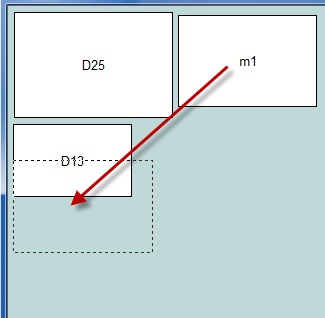
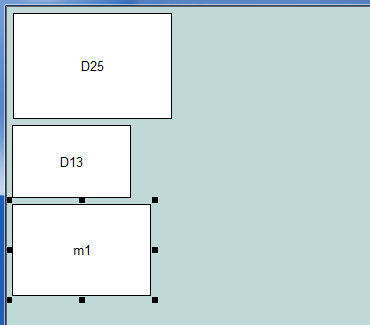
### test1\_2MoveLeftPushedBackDownRight02

### test1\_3MoveInsertedVertically1

### test1\_4MovePushedVertically2

## Scenario 2

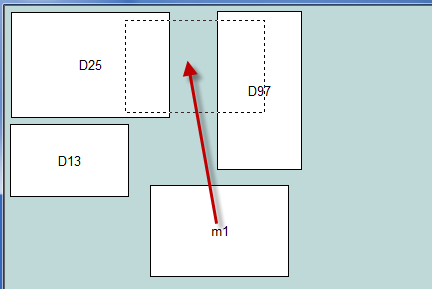
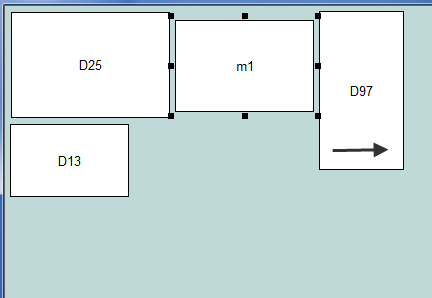
{'type':'node', 'id':'D25', 'x':7, 'y':6, 'width':159, 'height':106}

{'type':'node', 'id':'D13', 'x':6, 'y':119, 'width':119, 'height':73}

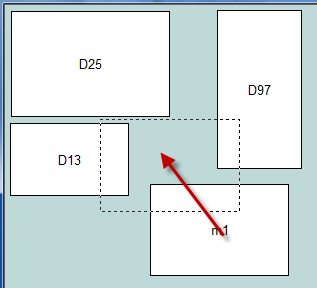
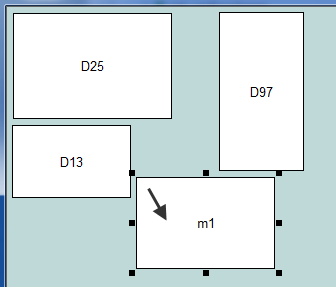
{'type':'node', 'id':'m1', 'x':146, 'y':179, 'width':139, 'height':92}

{'type':'node', 'id':'D97', 'x':213, 'y':6, 'width':85, 'height':159}

### Test2\_1InsertAndPushedRightHorizontally

### test2\_2PushedRightAndDownNicely

## Scenario 3

{'type':'node', 'id':'D25', 'x':7, 'y':6, 'width':159, 'height':106}

{'type':'node', 'id':'D13', 'x':6, 'y':119, 'width':119, 'height':73}

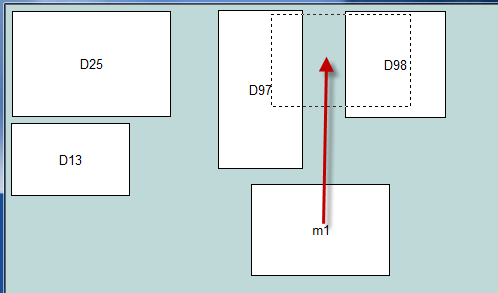
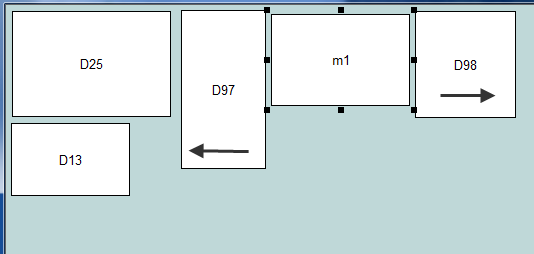
{'type':'node', 'id':'m1', 'x':246, 'y':179, 'width':139, 'height':92}

{'type':'node', 'id':'D97', 'x':213, 'y':6, 'width':85, 'height':159}

{'type':'node', 'id':'D98', 'x':340, 'y':7, 'width':101, 'height':107}

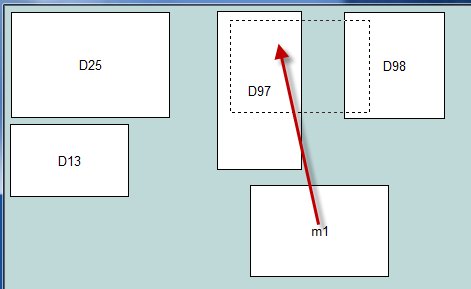
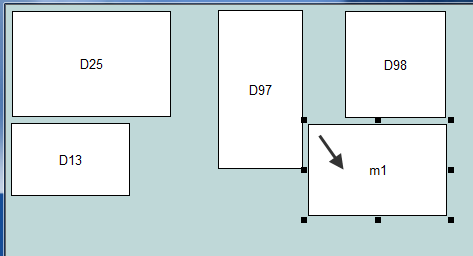
### test3\_1PushedBetweenLeftAndRight

assert m1 has been pushed between two nodes, horizontally. Both left and right nodes moved left and right respectively.

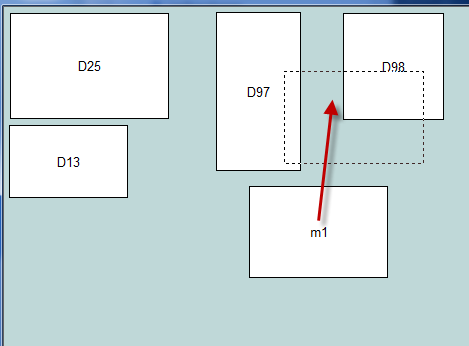
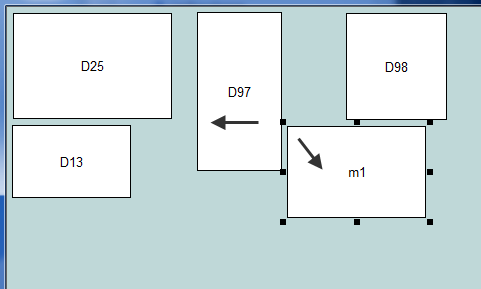
### test3\_2PushedBetweenLeftAndRightRefused

assert m1 has been not been inserted - refused and snuggled instead

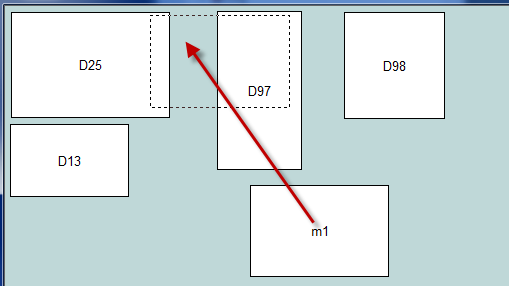
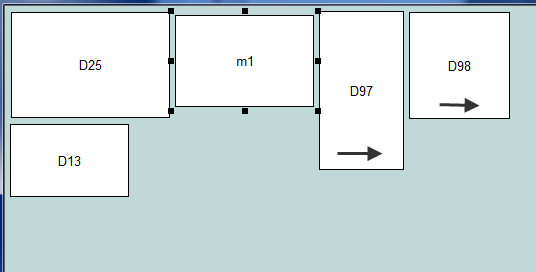
### test3\_2aPushedRefusedButLeftMovedAnyway

m1 has been refused insertion, but left (D97) moved leftwards cos there is room. m1 snuggled below and to the right.

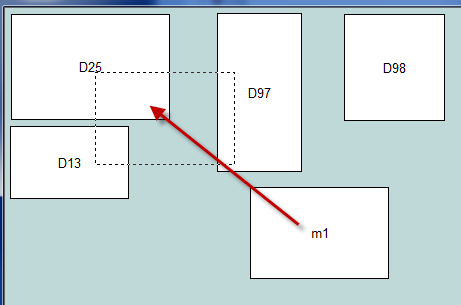
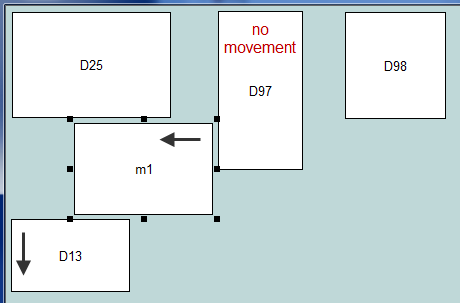
### test3\_3InsertedAndTwoPushedRight

assert m1 has been inserted - and two nodes pushed right

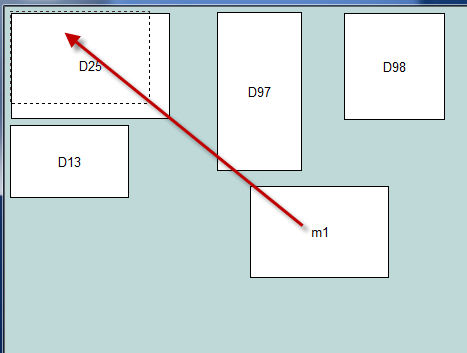
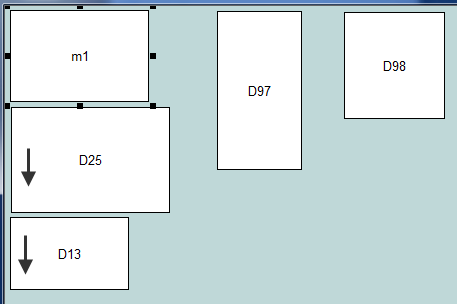
 

### test3\_4InsertedVerticallyNothingPushedRight

assert m1 has been inserted vertically - one node pushed down, NO nodes pushed right

### test3\_5InsertedVerticallyTwoPushedDown

## Scenario 4

{'type':'node', 'id':'D25', 'x':7, 'y':6, 'width':159, 'height':106}

{'type':'node', 'id':'D13', 'x':6, 'y':119, 'width':119, 'height':73}

{'type':'node', 'id':'m1', 'x':6, 'y':214, 'width':139, 'height':92}

{'type':'node', 'id':'D97', 'x':213, 'y':6, 'width':85, 'height':159}

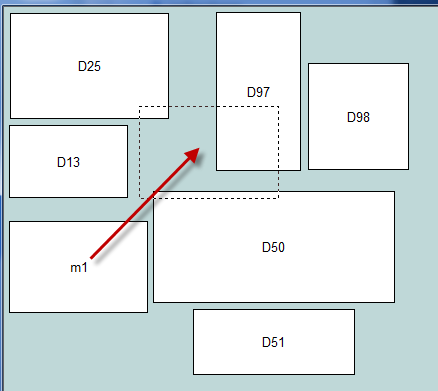
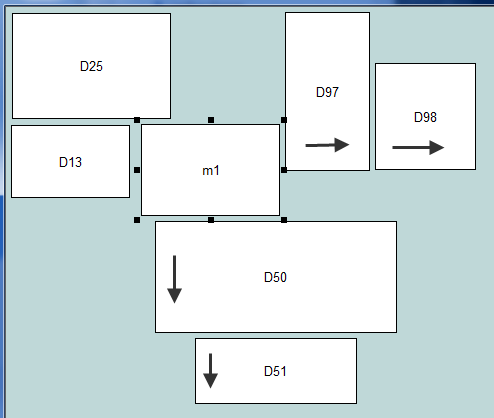
{'type':'node', 'id':'D98', 'x':305, 'y':57, 'width':101, 'height':107}

{'type':'node', 'id':'D50', 'x':149, 'y':184, 'width':242, 'height':112}

{'type':'node', 'id':'D51', 'x':189, 'y':302, 'width':162, 'height':66}

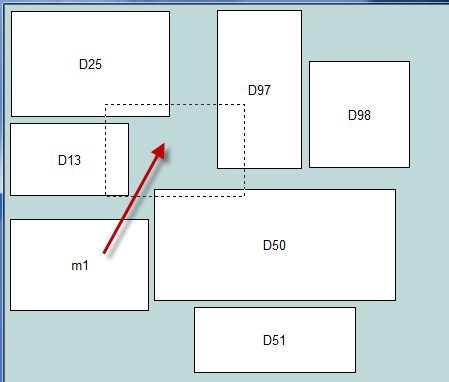
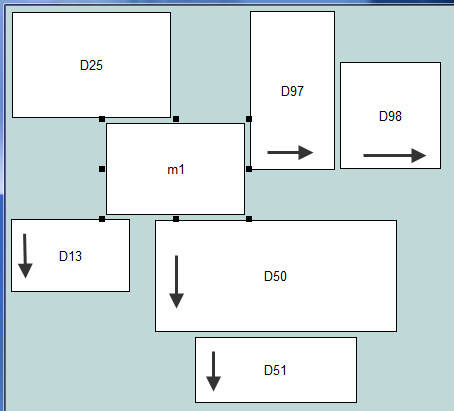
### test4\_1InsertedTwoPushedRightTwoPushedDown

assert m1 has been inserted - two pushed right, two pushed down

### test4\_2InsertedTwoPushedRightThreePushedDown

assert m1 has been inserted - two pushed right, two pushed down, and extra D13 pushed down because m1 overlaps/attacks D13 (and there is room for D13 to move downwards I guess)

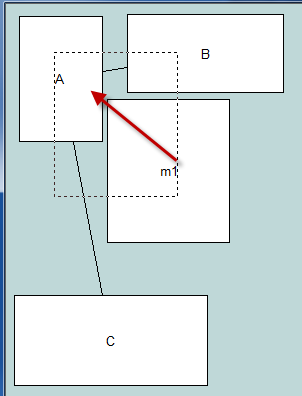
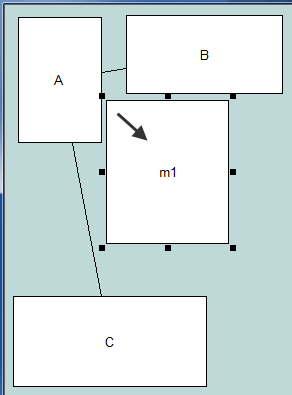
## Scenario 5 – stress tests

Run a complex jumble repeatedly till it fails. I increased MAX\_CYCLES to 20 to improve success.

## Scenario 6 – line crossing enhancements

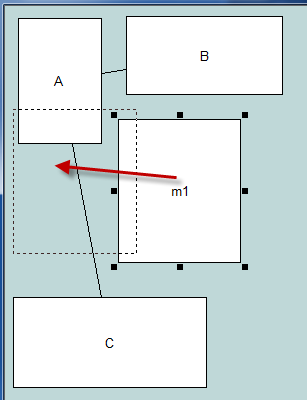
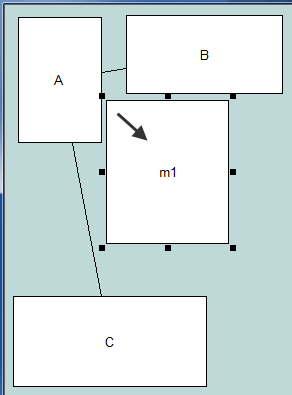
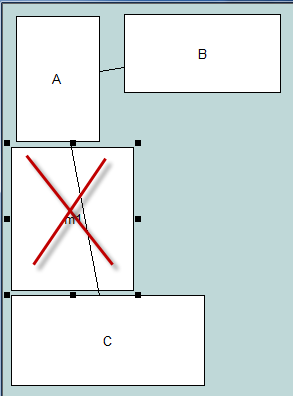
### test6\_1LineCrossingNotNeeded

Usual repulsion and snuggling.

### test6\_2LineCrossingAvoided

assert m1 has been repulsed and snuggeled, and line not crossed - same results as ABOVE

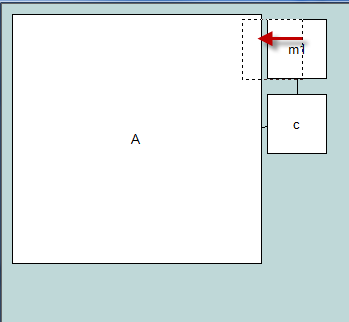
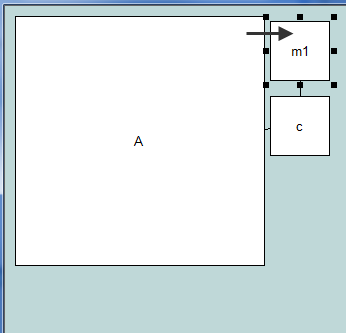
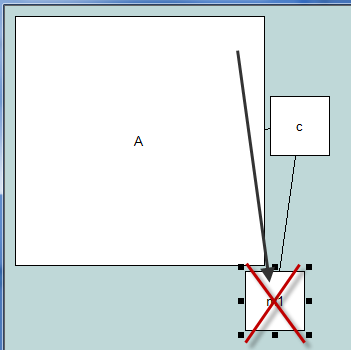
  **not** 

TODO?

## Scenario 7 – weird behaviour when edges involved

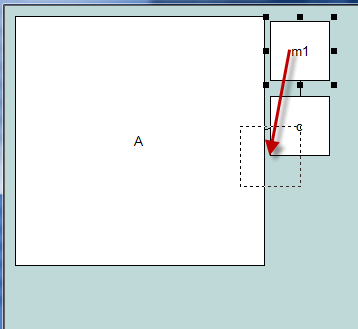
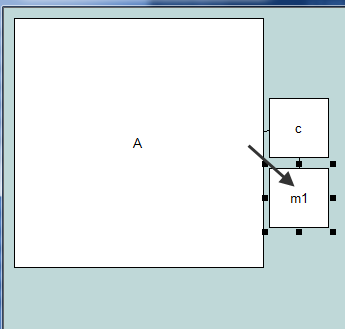
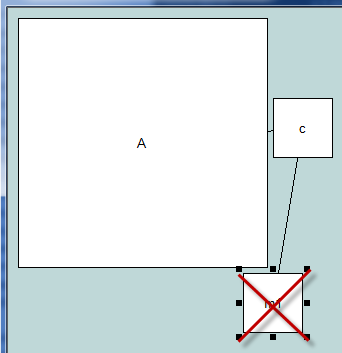
### test7\_1DontJumpTooFarY

assert m1 has been pushed to the right. don't see why edges should make any difference. initially found m1 was being pushed way too far down in the Y direction!

  **not **

### test7\_2DontJumpTooFarY

assert m1 has been pushed to the right and down – snugly – not pushed way too far down in the Y direction!

  **not **