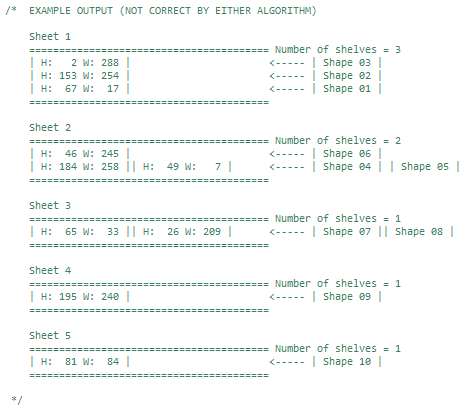
**Test Results**

Task 4

To test this program, I create a series of print line methods in the class CorrectnessTest. These methods break down the rules listed in the specification and then tests that the output is expected. The input is one I created that I knew would satisfy the rule. I then compared the values, such as number of sheets, shelves or shapes depending on the rule. I test both algorithms in one method and output a true or false statement for if that algorithm produced the expected output. In addition, I also implemented a print method within the Algorithms class that shows the layout of the resulting List in a way that matches how it would be drawn to make it easier to compare the results. The print method works as shown below – please not that the example in the image is made up and isn’t correct by either algorithm



Below are the inputs, expected outcome and actual outcome of the correctness testing

|  |  |  |
| --- | --- | --- |
| BOTHtestRuleAPart1() | | |
| Input |  | |
| Expected Outcome |  | Sheets = 1  Shelves in Sheet1 = 1  Shapes on Shelf1 = 1 |
| Actual Outcome | RULE A PART 1:  A shape is placed at the bottom left corner of a  sheet (starting the first shelf on the sheet)  Next Fit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 100 |  ======================================================================  First Fit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 100 |  ======================================================================  Using hand traces and comparing the results:  NextFit Passed = true  FirstFit Passed = true  ~~~~~~~~~~~~~~~~~~~~~~ | |

|  |  |  |
| --- | --- | --- |
| BOTHtestRuleAPart2() | | |
| Input |  | |
| Expected Outcome |  | Sheets = 1  Shelves in Sheet1 = 1  Shapes on Shelf1 = 2 |
| Actual Outcome | RULE A PART 2:  A shape is placed to the right of another shape, if  there is sufficient space in the shelf  NextFit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 100 || H: 100 W: 150 |  ======================================================================  NextFit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 100 || H: 100 W: 150 |  ======================================================================  Using hand traces and comparing the results:  NextFit Passed = true  FirstFit Passed = true  ~~~~~~~~~~~~~~~~~~~~~~ | |

|  |  |  |
| --- | --- | --- |
| BOTHtestRuleB() | | |
| Input |  | |
| Expected Outcome |  | Sheets = 1  Shelves in Sheet1 = 1  Shapes on Shelf1 = 3  Height of Last Shape = 75 |
| Actual Outcome | RULE B:  If a shape does not fit in a shelf, one rotates the  shape and tries to fit it in the shelf  NextFit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 100 || H: 100 W: 150 || H: 75 W: 25 |  ======================================================================  FirstFit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 100 || H: 100 W: 150 || H: 75 W: 25 |  ======================================================================  Using hand traces and comparing the results:  NextFit Passed = true  FirstFit Passed = true  ~~~~~~~~~~~~~~~~~~~~~~ | |

|  |  |  |
| --- | --- | --- |
| BOTHtestRuleCPart1() | | |
| Input |  | |
| Expected Outcome |  | Sheets = 1  Shelves in Sheet1 = 2  Shapes on Shelf1 = 2  Shapes on Shelf2 = 1  Height of Last Shape = 100 |
| Actual Outcome | RULE C PART 1:  If a shape still does not fit in a shelf, one can start a new shelf  directly on top of the current shelf against the left side of the sheet,  if there is enough space in the sheet. First, try to create a new shelf with the  shape in its original orientation  NextFit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 150 |  | H: 100 W: 100 || H: 100 W: 150 |  ======================================================================  FirstFit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 150 |  | H: 100 W: 100 || H: 100 W: 150 |  ======================================================================  Using hand traces and comparing the results:  NextFit Passed = true  FirstFit Passed = true  ~~~~~~~~~~~~~~~~~~~~~~ | |

|  |  |  |
| --- | --- | --- |
| BOTHtestRuleCPart2() | | |
| Input |  | |
| Expected Outcome |  | Sheets = 1  Shelves in Sheet1 = 2  Shapes on Shelf1 = 2  Shapes on Shelf2 = 1  Height of Last Shape = 100 |
| Actual Outcome | RULE C PART 2:  If shape in original orientation is too big to create new shelf, rotate it and try again  NextFit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 200 |  | H: 100 W: 100 || H: 100 W: 150 |  ======================================================================  FirstFit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 200 |  | H: 100 W: 100 || H: 100 W: 150 |  ======================================================================  Using hand traces and comparing the results:  NextFit Passed = true  FirstFit Passed = true  ~~~~~~~~~~~~~~~~~~~~~~ | |

|  |  |  |
| --- | --- | --- |
| BOTHtestRuleCPart3() | | |
| Input |  | |
| Expected Outcome |  | Sheets = 2  Shelves in Sheet1 = 1  Shelves in Sheet2 = 2  Shapes on Sheet1Shelf1 = 1 |
| Actual Outcome | RULE C PART 3:  If both original and rotated version of the shape are too big to create new shelf, create new sheet  NextFit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 100 || H: 100 W: 150 |  ======================================================================  Sheet 2  ======================================================================  | H: 200 W: 250 |  ======================================================================  FirstFit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 100 || H: 100 W: 150 |  ======================================================================  Sheet 2  ======================================================================  | H: 200 W: 250 |  ======================================================================  Using hand traces and comparing the results:  NextFit Passed = true  FirstFit Passed = true  ~~~~~~~~~~~~~~~~~~~~~~ | |

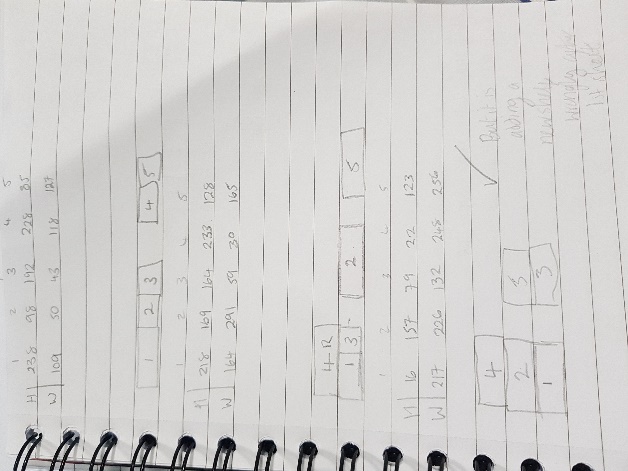
|  |  |  |
| --- | --- | --- |
| BOTHtestRuleE() | | |
| Input |  | |
| Expected Outcome |  | Height of sheet 1 = 100  Height of sheet 2 = 200 |
| Actual Outcome | RULE E:  Total height of all shelves in a sheet must be <= to the max (250 in this case)  NextFit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 100 || H: 100 W: 150 |  ======================================================================  Sheet 2  ======================================================================  | H: 200 W: 250 |  ======================================================================  FirstFit Visual Output:  Sheet 1  ======================================================================  | H: 100 W: 100 || H: 100 W: 150 |  ======================================================================  Sheet 2  ======================================================================  | H: 200 W: 250 |  ======================================================================  Using hand traces and comparing the results:  NextFit Passed = true  FirstFit Passed = true  ~~~~~~~~~~~~~~~~~~~~~~ | |

|  |  |  |
| --- | --- | --- |
| BOTHtestRuleF() | | |
| Input |  | |
| Expected Outcome |  | Sheets = 2 |
| Actual Outcome | RULE F:  Total number of shapes in a sheet can't exceed the max (20 in this case)  NextFit Visual Output:  Sheet 1  ======================================================================  | H: 10 W: 30 || H: 10 W: 30 || H: 10 W: 30 || H: 10 W: 30 | (plus 6 more)  | H: 10 W: 30 || H: 10 W: 30 || H: 10 W: 30 || H: 10 W: 30 | (plus 6 more)  ======================================================================  Sheet 2  ======================================================================  | H: 20 W: 20 |  ======================================================================  FirstFit Visual Output:  Sheet 1  ======================================================================  | H: 10 W: 30 || H: 10 W: 30 || H: 10 W: 30 || H: 10 W: 30 | (plus 6 more)  | H: 10 W: 30 || H: 10 W: 30 || H: 10 W: 30 || H: 10 W: 30 | (plus 6 more)  ======================================================================  Sheet 2  ======================================================================  | H: 20 W: 20 |  ======================================================================  Using hand traces and comparing the results:  NextFit Passed = true  FirstFit Passed = true  ~~~~~~~~~~~~~~~~~~~~~~ | |

In addition to testing the rules I also ran the random generator and then completed the algorithm by hand, comparing it with the visual output I created as well as looking into the variables to check that shelves were showing the correct number of shapes and so on. In the CorrectnessTest class there are two traces I have shown – one for each algorithm:

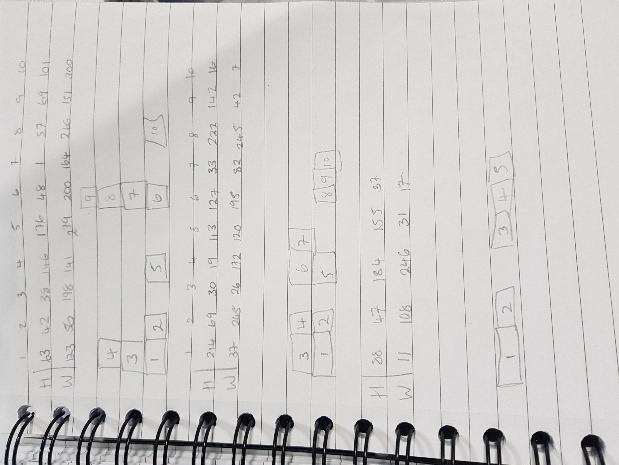
|  |  |  |
| --- | --- | --- |
| exampleNFTrace() | | |
| Input |  | |
| Expected Outcome | A close up of text on a black background  Description automatically generated | Sheets = 3  Shelves On Sheet1 = 1  Shelves On Sheet2 = 3  Shelves On Sheet3 = 2 |
| Actual Outcome | EXAMPLE TRACE:  NextFit Visual Output:  Sheet 1  ======================================================================  | H: 222 W: 161 || H: 137 W: 63 |  ======================================================================  Sheet 2  ======================================================================  | H: 88 W: 67 || H: 25 W: 187 |  | H: 20 W: 173 |  | H: 132 W: 99 || H: 100 W: 161 |  ======================================================================  Sheet 3  ======================================================================  | H: 27 W: 118 |  | H: 162 W: 217 || H: 90 W: 59 |  ======================================================================  Using hand traces and comparing the results:  NextFit Passed = true  ~~~~~~~~~~~~~~~~~~~~~~ | |

|  |  |  |
| --- | --- | --- |
| exampleFFTrace() | | |
| Input |  | |
| Expected Outcome | A close up of text on a black background  Description automatically generated | Sheets = 3  Shelves On Sheet1 = 1  Shelves On Sheet2 = 2  Shelves On Sheet3 = 1 |
| Actual Outcome | EXAMPLE TRACE:  FirstFit Visual Output:  Sheet 1  ======================================================================  | H: 222 W: 161 || H: 137 W: 63 || H: 173 W: 20 || H: 187 W: 25 || H: 118 W: 27 |  ======================================================================  Sheet 2  ======================================================================  | H: 88 W: 67 || H: 59 W: 90 |  | H: 132 W: 99 || H: 100 W: 161 |  ======================================================================  Sheet 3  ======================================================================  | H: 162 W: 217 |  ======================================================================  Using hand traces and comparing the results:  FirstFit Passed = true  ~~~~~~~~~~~~~~~~~~~~~~ | |

A close up of text on a white surface

Description automatically generatedA close up of text on a white background

Description automatically generatedA close up of a piece of paper

Description automatically generatedI also did lots more hand traces on paper as shown through the images below, making sure I had tried a trace where a shape had to be rotated to fit on the shelf, had to be rotated to fit on a new shelf or where a shape forced the creation of a new sheet and so on. I did these for both Next Fit and First Algorithm during the development process as well as afterwards in testing.

A close up of a piece of paper

Description automatically generated

Task 5

As shown in the results, the First Fit method uses a significantly lower number of sheets (for larger file sizes First Fit uses roughly half the amount of sheets) compared to the Next Fit algorithm – however it does take a lot longer.

Task 6

TESTING Next Fit ALGORITHM ONLY

Sorting by Width

| Num Of Shapes | Num Of Tests | Type of Test | Avg Sheets | Avg Time (mls) |

============================================================================

| 10 | 5 | Unsorted | 5 | 0.339

| | | Sorted (Asc) | 5 | 0.514

| | | Sorted (Des) | 5 | 0.650

----------------------------------------------------------------------------

| 100 | 5 | Unsorted | 49 | 0.317

| | | Sorted (Asc) | 46 | 0.658

| | | Sorted (Des) | 46 | 0.906

----------------------------------------------------------------------------

| 1000 | 5 | Unsorted | 480 | 1.496

| | | Sorted (Asc) | 460 | 3.765

| | | Sorted (Des) | 451 | 4.829

----------------------------------------------------------------------------

| 10000 | 5 | Unsorted | 4670 | 6.227

| | | Sorted (Asc) | 4450 | 34.241

| | | Sorted (Des) | 4377 | 41.213

----------------------------------------------------------------------------

| 20000 | 5 | Unsorted | 9391 | 6.694

| | | Sorted (Asc) | 8957 | 19.362

| | | Sorted (Des) | 8806 | 29.810

----------------------------------------------------------------------------

| 30000 | 5 | Unsorted | 14067 | 7.862

| | | Sorted (Asc) | 13435 | 26.126

| | | Sorted (Des) | 13224 | 36.482

----------------------------------------------------------------------------

| 40000 | 5 | Unsorted | 18751 | 8.285

| | | Sorted (Asc) | 17886 | 34.560

| | | Sorted (Des) | 17586 | 46.879

----------------------------------------------------------------------------

| 50000 | 5 | Unsorted | 23421 | 9.146

| | | Sorted (Asc) | 22361 | 32.413

| | | Sorted (Des) | 22002 | 44.276

----------------------------------------------------------------------------

Looking at the table above which shows the results for the next fit algorithm with lists sorted by width, it shows that sorting the list first results in the algorithm being much slower – however in all cases after the first sorting the list first reduces the amount of sheets needed – with descending order needing less sheets than ascending. Despite this the difference is quite close when you take into account the size of the list but the time difference is huge. The first test with the list being only 10 shapes, I believe that it showed no difference simply because the simply size was too small to have much variation – the second test only slightly differed as well.

TESTING Next Fit ALGORITHM ONLY

Sorting by Height

| Num Of Shapes | Num Of Tests | Type of Test | Avg Sheets | Avg Time (mls) |

============================================================================

| 10 | 5 | Unsorted | 5 | 0.004

| | | Sorted (Asc) | 5 | 0.042

| | | Sorted (Des) | 5 | 0.044

----------------------------------------------------------------------------

| 100 | 5 | Unsorted | 48 | 0.021

| | | Sorted (Asc) | 49 | 0.308

| | | Sorted (Des) | 44 | 0.332

----------------------------------------------------------------------------

| 1000 | 5 | Unsorted | 471 | 0.215

| | | Sorted (Asc) | 444 | 2.280

| | | Sorted (Des) | 434 | 2.485

----------------------------------------------------------------------------

| 10000 | 5 | Unsorted | 4715 | 1.525

| | | Sorted (Asc) | 4303 | 12.038

| | | Sorted (Des) | 4279 | 13.743

----------------------------------------------------------------------------

| 20000 | 5 | Unsorted | 9380 | 2.051

| | | Sorted (Asc) | 8584 | 8.505

| | | Sorted (Des) | 8551 | 11.059

----------------------------------------------------------------------------

| 30000 | 5 | Unsorted | 14090 | 2.817

| | | Sorted (Asc) | 12846 | 12.720

| | | Sorted (Des) | 12802 | 17.055

----------------------------------------------------------------------------

| 40000 | 5 | Unsorted | 18708 | 3.938

| | | Sorted (Asc) | 17110 | 20.863

| | | Sorted (Des) | 17048 | 29.812

----------------------------------------------------------------------------

| 50000 | 5 | Unsorted | 23454 | 6.502

| | | Sorted (Asc) | 21416 | 25.214

| | | Sorted (Des) | 21340 | 33.965

----------------------------------------------------------------------------

Sorting by height reduces the number of sheets drastically. In the largest file it removes over 2000 sheets compared to the amount it would need if the list wasn’t sorted beforehand. However it is a lot slower to complete the next fit algorithm when the list is already sorted. Looking at the difference in sheets between the ascending list and descending list there is little difference but the time required is very different, therefore it doesn’t appear smart to use the descending list as it wouldn’t reduce waste from the sheets by much but would take longer to complete.

TESTING Next Fit ALGORITHM ONLY

Sorting by Height \* Width

| Num Of Shapes | Num Of Tests | Type of Test | Avg Sheets | Avg Time (mls) |

============================================================================

| 10 | 5 | Unsorted | 5 | 0.005

| | | Sorted (Asc) | 5 | 0.053

| | | Sorted (Des) | 4 | 0.056

----------------------------------------------------------------------------

| 100 | 5 | Unsorted | 47 | 0.026

| | | Sorted (Asc) | 48 | 0.378

| | | Sorted (Des) | 46 | 0.400

----------------------------------------------------------------------------

| 1000 | 5 | Unsorted | 478 | 0.194

| | | Sorted (Asc) | 471 | 2.070

| | | Sorted (Des) | 462 | 2.535

----------------------------------------------------------------------------

| 10000 | 5 | Unsorted | 4660 | 1.894

| | | Sorted (Asc) | 4535 | 10.222

| | | Sorted (Des) | 4467 | 12.276

----------------------------------------------------------------------------

| 20000 | 5 | Unsorted | 9340 | 3.457

| | | Sorted (Asc) | 9083 | 13.441

| | | Sorted (Des) | 8926 | 17.995

----------------------------------------------------------------------------

| 30000 | 5 | Unsorted | 14022 | 3.854

| | | Sorted (Asc) | 13516 | 19.165

| | | Sorted (Des) | 13281 | 24.201

----------------------------------------------------------------------------

| 40000 | 5 | Unsorted | 18720 | 4.342

| | | Sorted (Asc) | 18016 | 23.957

| | | Sorted (Des) | 17692 | 30.749

----------------------------------------------------------------------------

| 50000 | 5 | Unsorted | 23438 | 4.989

| | | Sorted (Asc) | 22494 | 26.971

| | | Sorted (Des) | 22118 | 34.683

Looking at the tables from the next fit algorithm it is clear that sorting them in ascending order by their area only has a tiny effect on the amount of sheets used – however sorting them in descending order makes quite a large difference, especially in the larger files as the manufacturer will save over 2000 sheets on a batch of 50000 shapes. Unfortunately, the time it takes to complete the algorithm on a descending list is much longer and in all cases, it is much quicker to not sort the list at all.