Question 4a

For low comparison cost (first elements all different – randomly ordered to help determine average case)

Part A, n = 1500

0.008356181 seconds for insertion sort

0.00606884 seconds for merge sort

Converted to seconds from nanoseconds. We can see insertion sort is significantly more.

Part A, n = 500

0.001506426 seconds for insertion sort

0.002804877 seconds for merge sort

We can see insertion sort is significantly less.

We can binary search between these two n, 500 and 1500, to find an approximate answer.

See full results from binary search in the appendix.

The end result for n where merge sort becomes better than insertion sort with low comparison cost is around 950.

0.0028775837 seconds for insertion sort, n = 951

0.0028528939 seconds for merge sort, n = 951

Question 4b

For high comparison cost (only last elements differ – last elements differ and are randomly ordered to determine average case)

0.00133111718 seconds for insertion sort, n = 300

7.0503212E-4 seconds for merge sort, n = 300

We can see merge sort is significantly faster at this value

1.247742E-5 seconds for insertion sort, n = 10

2.980438E-5 seconds for merge sort, n = 10

We can see merge sort is significantly slower at this value

We can binary search between these two n, 10 and 300, to find an approximate answer.

See full results from binary search in the appendix.

The end result n for where merge sort becomes better than insertion sort with high comparison cost is around 140

2.72343E-4 seconds for insertion sort, n = 142

2.5093564E-4 seconds for merge sort, n = 142

Question 4c

For this question I will search for the answer modelling the curve dependent on t roughly as a quadratic. I will search for the minimum between a given range. I initially test between 500 and 1500 as I expect my answer to be marginally less than my answer to 4a. This proves correct as the upper bound when searching within this range does not strictly decrease towards 500.

Full results from ternary search in appendix

Sample of results:

0.0032552734 seconds for hybrid sort, threshold = 500

0.0031874172 seconds for hybrid sort, threshold = 1000

0.0027820484 seconds for hybrid sort, threshold = 648

0.0024803858 seconds for hybrid sort, threshold = 727

Hence our best result threshold for low cost comparisons is around 730

Question 4d

For this question we can use a similar approach as 4c. The range I choose is 0 and 300. The search does not strictly increase towards and end near 300 hence valid range.

Full results from ternary search in appendix.

Sample of results:

4.5873936E-4 seconds for hybrid sort, threshold = 100

4.2534142E-4 seconds for hybrid sort, threshold = 200

4.119571E-4 seconds for hybrid sort, threshold = 166

6.0594912E-4 seconds for hybrid sort, threshold = 234

4.0780244E-4 seconds for hybrid sort, threshold = 119

4.024309E-4 seconds for hybrid sort, threshold = 115

Hence our best result threshold for high cost comparisons is around 120.

Question 4e

Denoting variables n = length of initial array to be sorted, t = threshold in hybrid sort

We can see the complexity of the part of the algorithm which is insertion sort is t^2 * (n/t). This is as average and worst case insertion sort is t^2. We then perform this operation on n/t chunks. We can then simplify our overall result to t*n

We can see the complexity of part of the algorithm which is merge sort is nlog(n/t). This is as the depth a traditional merge sort goes down to is log(n). However our depth is reduced by the 'depth' insertion sort removes, which is log(t). This means depth our merge sort goes to is log(n)-log(t) = log(n/t). Array is length n hence n*log(n/t).

Hence overall:

$\Theta(n, t) = \Theta(n*t + n*\log(n/t))$

Appendix

Part A

Binary Search Results

Number of repeats = 1000

0.0036481437 seconds for insertion sort, n = 1000

0.0030607744 seconds for merge sort, n = 1000

0.0018964678 seconds for insertion sort, n = 750

0.0025088224 seconds for merge sort, n = 750

0.0025584621 seconds for insertion sort, n = 875

0.0026956145 seconds for merge sort, n = 875

0.002753735 seconds for insertion sort, n = 937

0.0029307284 seconds for merge sort, n = 937

0.0028814807 seconds for insertion sort, n = 968

0.0026811282 seconds for merge sort, n = 968

0.0029563443 seconds for insertion sort, n = 952

0.0029455277 seconds for merge sort, n = 952

0.0027655662 seconds for insertion sort, n = 944

0.0029396095 seconds for merge sort, n = 944

0.0028151336 seconds for insertion sort, n = 948

0.0029461272 seconds for merge sort, n = 948

0.0028129055 seconds for insertion sort, n = 950

0.0028877155 seconds for merge sort, n = 950

0.0028775837 seconds for insertion sort, n = 951

0.0028528939 seconds for merge sort, n = 951

Part B

Binary Search Results

Number of repeats = 5000

9.5409406E-4 seconds for insertion sort, n = 255

4.8845418E-4 seconds for merge sort, n = 255

2.1745042E-4 seconds for insertion sort, n = 132

2.3795544E-4 seconds for merge sort, n = 132

4.7235014E-4 seconds for insertion sort, n = 193

3.668519E-4 seconds for merge sort, n = 193

3.35933E-4 seconds for insertion sort, n = 162

2.9551806E-4 seconds for merge sort, n = 162

2.8166798E-4 seconds for insertion sort, n = 147

2.748294E-4 seconds for merge sort, n = 147

2.715004E-4 seconds for insertion sort, n = 139

2.972855E-4 seconds for merge sort, n = 139

2.7877232E-4 seconds for insertion sort, n = 143

2.4202748E-4 seconds for merge sort, n = 143

2.6210888E-4 seconds for insertion sort, n = 141

2.7574572E-4 seconds for merge sort, n = 141

2.72343E-4 seconds for insertion sort, n = 142

2.5093564E-4 seconds for merge sort, n = 142

Part C

Ternary Search Results

Num repeats = 1000

0.0032552734 seconds for hybrid sort, threshold = 500

0.0031874172 seconds for hybrid sort, threshold = 1000

0.0025203536 seconds for hybrid sort, threshold = 833

0.003422766 seconds for hybrid sort, threshold = 1167

0.0025748 seconds for hybrid sort, threshold = 722

0.0026965404 seconds for hybrid sort, threshold = 945

0.0027820484 seconds for hybrid sort, threshold = 648

0.0027780544 seconds for hybrid sort, threshold = 797

0.0025909198 seconds for hybrid sort, threshold = 747

0.0027684924 seconds for hybrid sort, threshold = 846

0.0027094306 seconds for hybrid sort, threshold = 714

0.0029056136 seconds for hybrid sort, threshold = 780

0.0027949614 seconds for hybrid sort, threshold = 692

0.0027413528 seconds for hybrid sort, threshold = 736

0.002742757 seconds for hybrid sort, threshold = 721

0.0026900754 seconds for hybrid sort, threshold = 751

0.0025869274 seconds for hybrid sort, threshold = 740

0.0026300884 seconds for hybrid sort, threshold = 761

0.0024890324 seconds for hybrid sort, threshold = 734

0.002775067 seconds for hybrid sort, threshold = 748

0.0025458728 seconds for hybrid sort, threshold = 730

0.002590966 seconds for hybrid sort, threshold = 739

0.0027474506 seconds for hybrid sort, threshold = 727

0.0027605468 seconds for hybrid sort, threshold = 733

0.0026521742 seconds for hybrid sort, threshold = 725

0.002493326 seconds for hybrid sort, threshold = 729

0.0025040852 seconds for hybrid sort, threshold = 727

0.0026748562 seconds for hybrid sort, threshold = 731

0.0024803858 seconds for hybrid sort, threshold = 727

0.002764643 seconds for hybrid sort, threshold = 729

0.0027981266 seconds for hybrid sort, threshold = 726

0.0026409996 seconds for hybrid sort, threshold = 728

0.0028949282 seconds for hybrid sort, threshold = 727

0.002841165 seconds for hybrid sort, threshold = 728

Part D

Ternary Search Results

4.5873936E-4 seconds for hybrid sort, threshold = 100

4.2534142E-4 seconds for hybrid sort, threshold = 200

4.119571E-4 seconds for hybrid sort, threshold = 166

6.0594912E-4 seconds for hybrid sort, threshold = 234

3.4133292E-4 seconds for hybrid sort, threshold = 144

3.5215624E-4 seconds for hybrid sort, threshold = 190

3.4938036E-4 seconds for hybrid sort, threshold = 130

3.8031504E-4 seconds for hybrid sort, threshold = 160

3.4488068E-4 seconds for hybrid sort, threshold = 120

3.5776142E-4 seconds for hybrid sort, threshold = 140

3.8869418E-4 seconds for hybrid sort, threshold = 113

3.5935774E-4 seconds for hybrid sort, threshold = 127

3.5500652E-4 seconds for hybrid sort, threshold = 122

3.605182E-4 seconds for hybrid sort, threshold = 131

3.352732E-4 seconds for hybrid sort, threshold = 119

3.6510732E-4 seconds for hybrid sort, threshold = 125

3.4410094E-4 seconds for hybrid sort, threshold = 117

3.5276736E-4 seconds for hybrid sort, threshold = 121

3.8320854E-4 seconds for hybrid sort, threshold = 115

4.0780244E-4 seconds for hybrid sort, threshold = 119

3.9946124E-4 seconds for hybrid sort, threshold = 115

4.0918342E-4 seconds for hybrid sort, threshold = 117

3.8162016E-4 seconds for hybrid sort, threshold = 114

3.9042644E-4 seconds for hybrid sort, threshold = 116

3.7863458E-4 seconds for hybrid sort, threshold = 114

4.024309E-4 seconds for hybrid sort, threshold = 115