DATA:
PercolationDFS
N= 50 T= 12 time: 0.1311
N= 100 T= 12 time: 0.5150
N= 200 T= 12 time: 6.6674
N= 50 T= 25 time: 0.0822
N= 100 T= 25 time: 1.0447
N= 200 T= 25 time: 14.7354
N= 50 T= 50 time: 0.1632
N= 100 T= 50 time: 2.1364
N= 200 T= 50 time: 29.3171

PercolationDFSFast
N= 50 T= 12 time: 0.0470
N= 100 T= 12 time: 0.1272
N= 200 T= 12 time: 0.1910
N= 50 T= 25 time: 0.0074
N= 100 T= 25 time: 0.0438
N= 200 T= 25 time: 0.1549
N= 50 T= 50 time: 0.0136
N= 100 T= 50 time: 0.0572
N= 200 T= 50 time: 0.2374

PercolationUF - QuickFind
N= 50 T= 12 time: 0.0531
N= 100 T= 12 time: 0.2627
N= 200 T= 12 time: 3.8404
N= 50 T= 25 time: 0.0416
N= 100 T= 25 time: 0.5247
N= 200 T= 25 time: 8.0671
N= 50 T= 50 time: 0.0787
N= 100 T= 50 time: 1.0253
N= 200 T= 50 time: 16.0216

PercolationUF - QuickUWPC
N= 50 T= 12 time: 0.0165
N= 100 T= 12 time: 0.0479
N= 200 T= 12 time: 0.0886
N= 50 T= 25 time: 0.0054
N= 100 T= 25 time: 0.0382
N= 200 T= 25 time: 0.1202
N= 50 T= 50 time: 0.0105
N= 100 T= 50 time: 0.0529
N= 200 T= 50 time: 0.2028

1. How does doubling the grid-size, N, affect the running time?
2. How does doubling the number of experiments, T, performed affect the running time?
3. Try to provide a formula for the running time in terms of N and T, use big-Oh.
4. Estimate the largest grid-size you can run in a day for 100 trials (assume time is the only limit here, not memory).
5. Give estimate for how much memory is used in terms of N, the grid-size. Provide your estimate in bytes and use four bytes for an int, one byte for a boolean, and eight bytes for a double. For example, an array of N integers uses 4N bytes in this model, there's no overhead for the array other than storing the integer values.

PercolationDFS:

1) It multiplies running time by 2^(4)=16. This makes sense as open is called on average Coefficient*N^2 times and for each open all N^2 boxes are checked to see if they are full.
2) It doubles the running time.
3) O((N^4)*T)
4)For N=200 T=50 running time is 29.3 seconds. For T=100 it would be 58.6 seconds. In a given day there are 24*60*60 seconds.
C*200^4*T = 58.6 seconds
C*N^4*T = 86400 seconds
(N/200)^4 = 1474.4
N/200 = 6.2
N = 1240, Thus largest grid size would be 1240*1240
5) Aproximately N*N*N*N*4 bytes.

PercolationDFSFast
1) It multiplies running time by 2^(2)=4 This is consisting with the fact that open is called for Coefficient*N^2 times until it percolates and for each open, dfs (O(1)) is called for some small number.
2) It doubles the running time.
3) Looking at part 1 and 2, big Oh seems like (N^2)*T, which is true as dfs will be called for small number of times for eaach open. However dfs is a recursive function and at worst case scenerio it will be called for coefficient*N^2 times meaning worst case scenerio big Oh is O((N^4)*T) but this is very unlucky in a real life situation, especially considering the 0.59 open average needed to percolate.
4) N=200 and T=100 will be about 0.47 seconds.
C*(200^2)*T = 0.47 seconds
C*(N^2)*T = 86400 seconds
N/200 = 429
N = 86000, thus approximately 86000*86000.
5) Appriximately N*N*4 bytes.

PercolationUF - QuickFind
1) It multiplies running time by 2^(4)=16. This makes sense as open is called for N^2 times and for each open quickfind has N calls of union with N complexity.
2)It doubles the running time.
3) Run time complexity of O((N^4)*T).
4)N=200, T=100 will be about 32 seconds.
C*(200^4)*T=32 seconds
C*(N^4)*T = 86400
(N/200) = 7.2
N = 1440, thus approximately 1440*1440.
5) Approximately N*N*N*N*4 bytes times.

PercolationUF - QuickUWPC
1)It multiplies running time by 2^(2) =4. This makes sense as union is called one time for each N^2 open command.
2) It doubles the running time.
3) Run time complexity, O((N^2)*T)
4)N=200, T=100 will be about 0.4 seconds.
C*(200^2)*T=0.4 seconds
C*(N^2)*T = 86400 seconds
N/200 = 465
N = 93000, thus approximately 93000*93000.
5) N*N*4 bytes.