|  |  |
| --- | --- |
| ONCE | REPEATED |
| int maxNum = 0; | arryBeg < arryEnd |
| int arryEnd = arry.length – 1; | int numCurr = arryBeg; |
| int arryBeg = 0; | i <= arryEnd; |
| int counter = 0; | arry[i] > numCurr |
| arryBeg = = arryEnd | numCurr = arry[i]; |
| int i = arryBeg; | maxNum = i; |
| i > arryEnd | i++; |
|  | counter == 0 || counter == 1 |
|  | swapArry(arry, maxNum, arryEnd); |
|  | int tempNum; |
|  | tempNum = arry[maxNum]; |
|  | arry[maxNum] = arry[swap]; |
|  | arry[swap] = tempNum; |
|  | counter++; |
|  | arryEnd--; |
|  | counter == 2 || counter == 3 |
|  | swapArry(arry, maxNum, arryBeg); |
|  | int tempNum; |
|  | tempNum = arry[maxNum]; |
|  | arry[maxNum] = arry[swap]; |
|  | arry[swap] = tempNum; |
|  | counter++; |
|  | arryBeg++; |
|  | counter == 4 |
|  | counter = 0; |

Time Complexity: 7+25n

Big O(n)

Best Case: The best-case scenario is when the array is already sorted this would cause the program to run the least amount of time possible

Worst Case / Average Case: The worst case and the average case are the same. It does not matter how big / small or mixed up the array is, it will take the program 7+25n amount of times to run anyways;