BLG 335E
HW1 REPORT

O. Kürşat Karayılan
150140011

a)

Bubblesort

	STATEMENT	STEPS	FREQUENCY		TOTAL STEPS	
			IF sorted = true	IF FALSE	IF TRUE	IF FALSE
1	int i = N; bool sorted = false;	2	1	1	2	2
2	while(i>1 and sorted == false)	2	1	N		
3	sorted = true	1	1	N-1		
4	for(int j = 1; j<i; j++)	1	N+1	N+1 * (N-1)		
5	if(arr[j]<arr[j-1])	1	N	N * (N-1)		
6	int temp = arr[j]; arr[j] = arr[j-1]; arr[j-1] = temp; sorted = false;	4	N	N * (N-1)		
7	i = i -1	1	1	N-1		
TOTAL			O(N)	O(n^2)		

```
void bubblesort(int arr[], int N){
    int i = N;
    bool sorted = false;
    while(i>1 and sorted == false){
        sorted = true;
        for(int j = 1; j<i; j++){
            if(arr[j]<arr[j-1]){
                int temp = arr[j];
                arr[j] = arr[j-1];
                arr[j-1] = temp;
                sorted = false;
            }
        }
        i = i-1;
    }
}
```

Bubble sort works with O(n^2) time complexity. In my implementation it is same. If the list is already sorted, then it works with O(n).

Merge

	STATEMENT	STEPS	FREQUENCY		TOTAL STEPS	
			IF TRUE	IF FALSE	IF TRUE	IF FALSE
1	int i, j, k; int temp[r+1]; i = p; k = p; j = q + 1;	7	1	1	1	1
2	while(i<=q && j<=r)	1	n/2	n/2	n/2	n/2
3	if(arr[i]<arr[j])	1	n/2	n/2	n/2	n/2
4	temp[k] = arr[i]; k++; i++;	3	n/2	0	3n/2	0
	else		0	n/2	0	n/2
	temp[k] = arr[j]; k++; j++;	3	0	n/2	0	3n/2
	while(i<=q)	1	n/2	n/2	n/2	n/2
	temp[k] = arr[i]; k++; i++;	3	n/2	n/2	n/2	n/2
	while(j<=r)	1	n/2	n/2	n/2	n/2
	temp[k] = arr[j]; k++; j++;	3	n/2	n/2	n/2	n/2
	for(i=p;i<k;i++)	1	n+1	n+1	n+1	n+1
	arr[i] = temp[i]	1	n	n	n	n
TOTAL					O(n)	Ω(n)

```
void merge(int arr[], int p, int r, int q){
    int i, j, k;
    int temp[r+1];
    i = p;
    k = p;
    j = q + 1;
    while(i<=q && j<=r){
        if(arr[i]<arr[j]){
            temp[k] = arr[i];
            k++;
            i++;
        }
        else{
            temp[k] = arr[j];
            k++;
            j++;
        }
    }
    while(i<=q){
        temp[k] = arr[i];
        k++;
        i++;
    }
    while(j<=r){
        temp[k] = arr[j];
        k++;
        j++;
    }
    for(i=p;i<k;i++){
        arr[i] = temp[i];
    }
}
```

Merge function works with O(n). Whether the list sorted or not, it is same.

Mergesort

	STATEMENT	STEPS	FREQUENCY		TOTAL STEPS	
			IF TRUE	IF FALSE	IF TRUE	IF FALSE
1	if(p<r)	1	1	1	1	1
2	int q = (p+r)/2	1	1	0	1	1
3	mergesort(arr, p, q)	x	1	0	x	0
4	mergesort(arr, q+1, r)	x	1	0	x	0
5	merge(arr, p, r, q)	O(n)	1	0	O(n)	
TOTAL					2x + O(n) + 2	2
					nlogn	nlogn

```
void mergesort(int arr[], int p, int r){
    if(p<r){
        int q = (p+r)/2;
        mergesort(arr, p, q);
        mergesort(arr, q+1, r);
        merge(arr, p, r, q);
    }
}
```

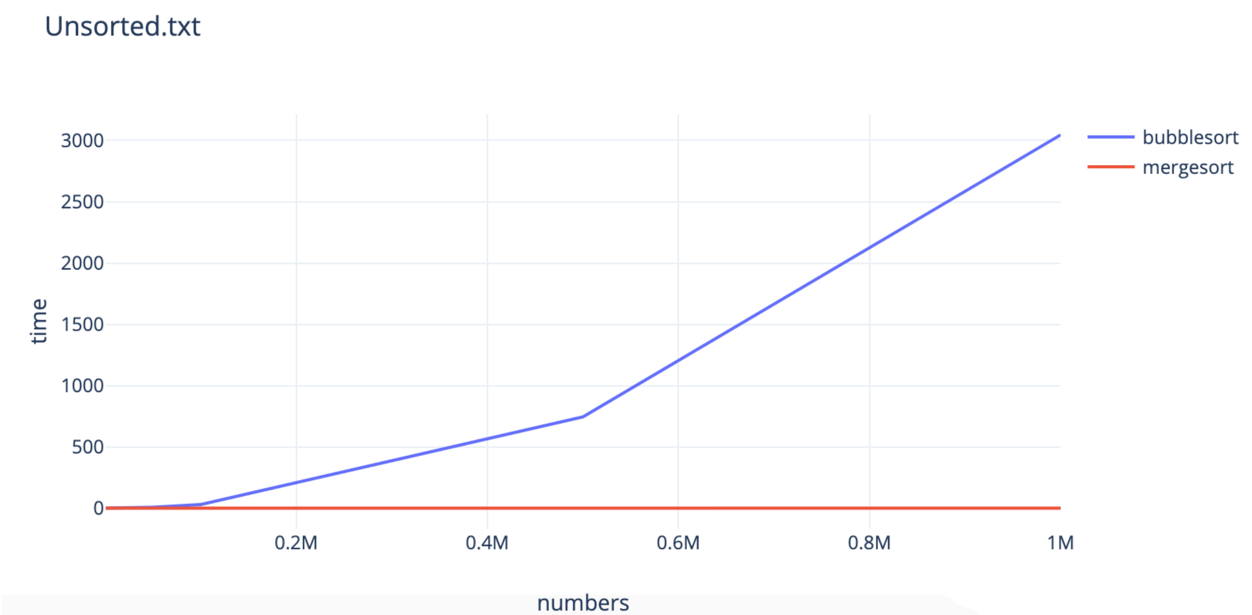
Mergesort’s time complexity is nlogn. I got same result from the equation $T(n)=2T(n/2)+O(n)$.

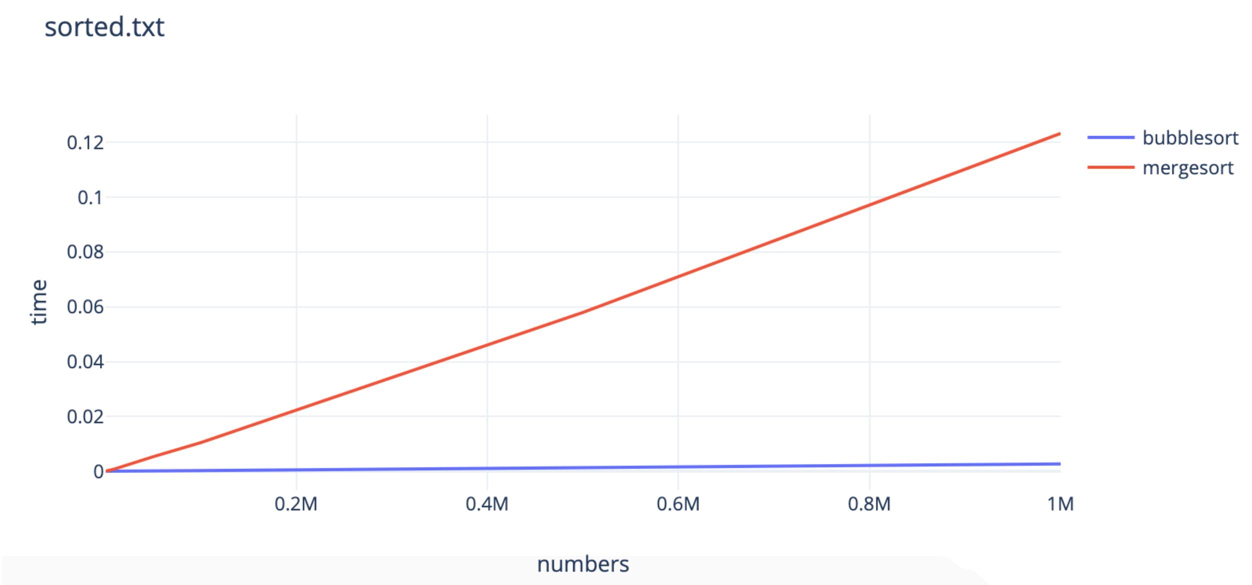
b)

	sorted		unsorted	
	buble	merge	buble	merge
1000	4e-06	8.3e-05	0.002139	0.000125
10000	3.1e-05	0.000948	0.253224	0.002162
100000	0.000261	0.010517	29.84	0.01874
1000000	0.002618	0.120992	3035.39	0.224181

Bubblesort works good if the list already sorted. Because it checks whole list just for once after the second(new one) implementation of the algorithm. But if the list unsorted, time increases significantly. It sorts 1mil number in 50mins. Mergesort works good in all cases as we can see above picture.

c)





As you can see from the plots mergesort works almost like a constant time compared to bubblesort for unsorted list. But bubblesort works slightly better for sorted list, because time amounts already in very small scale. We can choose mergesort for unsorterd list and bubblesort for sorted list.

d)

	STATEMENT	STEPS	FREQUENCY	TOTAL STEPS
1	r <- 0	1	1	
2	for i <- 1 to n do	1	n+1	
3	for j <- i+1 to n do	1	n*n	
4	for k <- 1 to j do	1	n*n-1*n	
5	r <- r+1;	1	1	
6	return r	1		
TOTAL			O(n^3)	

It’s time complexity O(n^3)

We can write the function as following equation:

$$\text{mystery(n)} = \sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{k=1}^{j} 1$$

$$= \sum_{k=1}^{j} 1 = \text{j}$$

$$= \sum_{j=i+1}^{n} j$$

$$= \sum_{j=1}^{n-i} (j + i)$$

$$= (n - i)(n - i + 1)/2 \quad + \quad (n - i)i$$

$$= \sum_{i=1}^{n} (n - i)(n - i + 1)/2 \quad + \quad (n - i)i$$

$$= \sum_{i=1}^{n} n^{2} - n(2i - 1) \; - \; (i^{2} + i)$$

$$= n^{3} \; - n(2n(n + 1)/2 - n) \; - \; (\tfrac{n(n+1)(2n+1)}{6} + n(n + 1)/2)$$