

ALGORITHM LAB

COUSE CODE: CSE214

BUBBLE SORT, LINEAR SORT, INSERTION SORT, SELECTION SORT MPLEMENTATION &

ANALYSIS OF TIME COMPLEXITY.

*M Shahriar Ishtiaque*

*191-15-12938*

*Section: 0 14*

# Bubble Sort

#include<stdio.h>

int main()

{

int x1[]= {2,3,5,7,11,13,17,19,23,29}; int x[]={5,2,8,14,10,9}; int t,i,j; for(i=0;i<6;i++)

{

for(j=0;j<6-i-1;j++)

{

if(x[j]>x[j+1])

{ t=x[j]; x[j]=x[j+1]; x[j+1]=t;

}

} }

for(i=0;i<6;i++)

{

printf("%d ",x[i]);

}

}

We have to do n-1 comparisons in 1st iteration and n-2 in 2nd one or n-3 in 3rd and so on..

So,

(n-1) + (n-2) + (n-3) + ..... + 3 + 2 + 1

Sum = n(n-1)/2

Hence,0(n^2)

# Linear Search

#include<stdio.h>

int main()

{

int x[]= {23,2,11,5,19,3,7,17,23,29}; int i,data=5; for(i=0; i<10; i++)

{

if(data==x[i])printf("%d found at %d",data,i);

}

}

Time complexity:

In best case the time complexity would be O (1)

The worst-case would-be O(n) because there could be case where we would iterate through the full array but didn't find the data or the data is at the last.

# Insertion Sort

#include<stdio.h> int main ()

{

int x1[]= {2,3,5,7,11,13,17,19,23,29}; int x[]={5,2,8,14,10,9}; int t,i,j; for(i=0;i<6;i++)

{ for(j=0;j<6-i-1;j++)

{ if(x[j]>x[j+1])

{ t=x[j]; x[j]=x[j+1]; x[j+1]=t;

}

} } for(i=0;i<6;i++)

{ printf("%d ",x[i]);

}

}

1st iteration => | 4 | 3 | 2 | 1 | No. of comparisons = 1 | No. of movements = 1

2nd iteration => | 3 | 4 | 2 | 1 | No. of comparisons = 2 | No. of movements = 2

3rd iteration => | 2 | 3 | 4 | 1 | No. of comparisons = 3 | No. of movements = 3

4th iteration => | 1 | 2 | 3 | 4 | No. of comparisons = 4 | No. of movements = 4

T(n) = 2 + 4 + 6 + 8 + ---------- + 2(n-1)

T(n) = 2 \* ( 1 + 2 + 3 + 4 + -------- + (n-1))

T(n) = 2 \* (n(n-1))/2

T(n) = O(n^2)

# Selection Sort

#include<stdio.h> int main()

{

int x[]={5,4,3,2,1}; int i,j,min,t; for(i=0;i<5;i++) { min=i; for(j=i+1;j<5;j++)

{

if(x[min]>x[j])

{ min=j; } if(min!=i) { t=x[i]; x[i]=x[min]; x[min]=t;

}

}

} for(i=0;i<5;i++) {

printf("%d ",x[i]);

}

}

iteration 1 -> 1 to n iteration 2 -> 2 to n iteration 3 -> 3 to n iteration 4 -> 4 to n

So we can say, n+n-1+n-2+..+1

n(n-1)/2 if we apply big oh notion O(n^2-n) or O(n^2 Hence 0(n^2) is the time complexity.