

LAPORAN PRAKTIKUM ANALISIS ALGORITMA



Disusun oleh:

Fadlan Mulya Priatna

140810180041

Kelas A

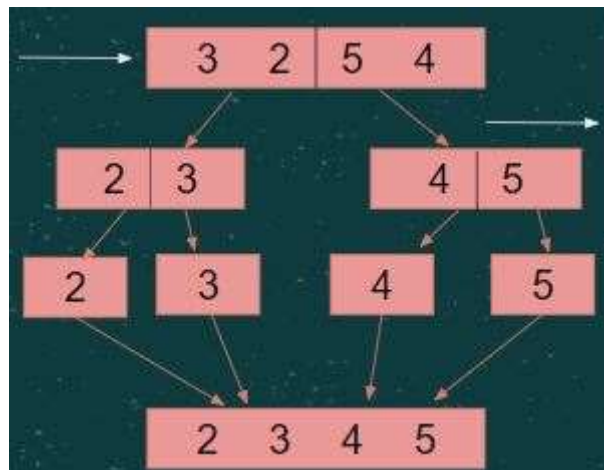
Program Studi S-1 Teknik Informatika
Departemen Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Padjadjaran

A. Paradigma Divide dan Conquer

Divide and Conquer adalah teknik algoritmik dengan cara memecah input menjadi beberapa bagian, memecahkan masalah di setiap bagian secara rekursif, dan menggabungkan solusi untuk subproblem ini menjadi solusi keseluruhan. Algoritma yang menggunakan *divide and conquer* adalah **algoritma merge-sort**.

B. Merge Sort

Misalkan ada array sebagai berikut. Tujuan kita adalah untuk mengurutkan dari terkecil - terbesar.



Proses **divide** pada saat membagi array menjadi 2 bagian $D(n)$.

Proses **conquer** pada saat array yang dibagi dua diurutkan $C(n)$.

$$\text{Sehingga, } T(n) = aT\left(\frac{n}{b}\right) + D(n) + C(n)$$

a = banyaknya cabang di awal proses *divide* $\rightarrow 2$

b = pada proses *divide*, array dibagi menjadi berapa bagian? $\rightarrow 2$

$D(n)$ = proses *divide* ; $C(n)$ = proses *conquer*

C. Metode Penyelesaian Requarensi

1) Metode Subtitusi

$$T(n) = aT\left(\frac{n}{b}\right) + cn$$

- Menebak sebuah solusi
- Membuktikan dengan cara induksi matematika

2) Recursion Tree

$$T(n) = aT\left(\frac{n}{b}\right) + cn$$

- Membuat recursion tree dari soal
- Melihat pola bilangan pada setiap level di recursion tree untuk mencapai $T(1)$

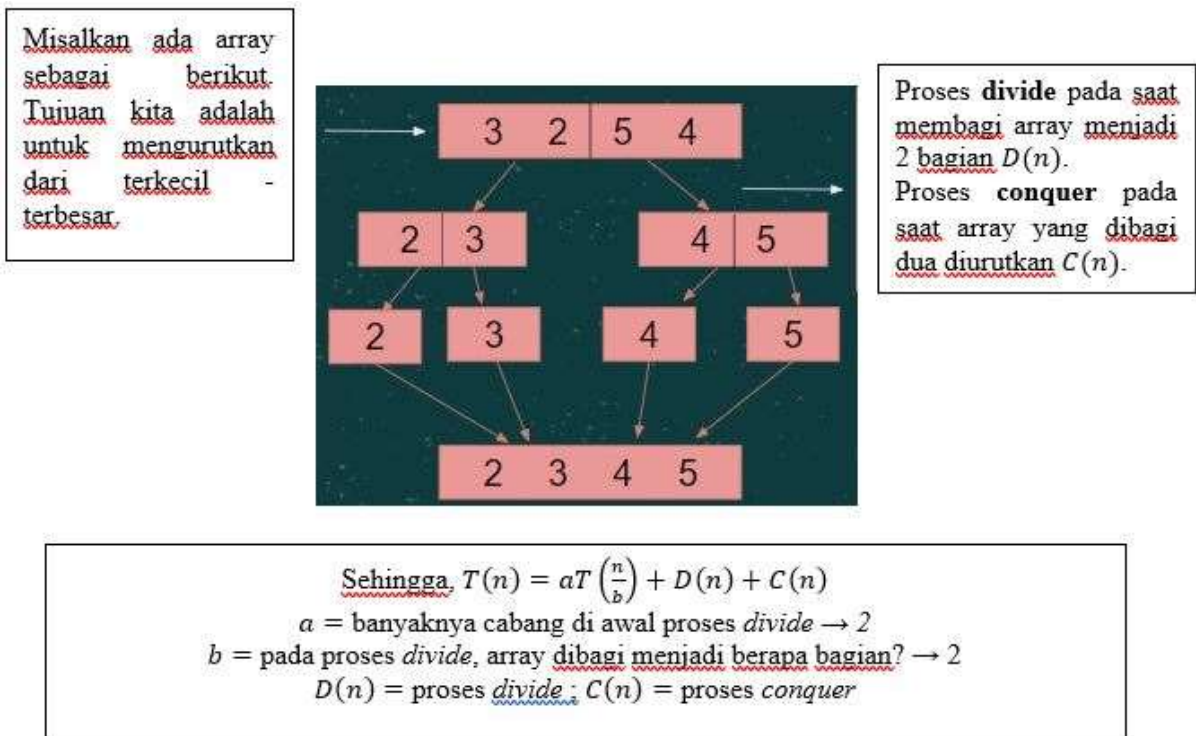
3) Metode Master

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

- Urutan pengerjaan cek contoh soal

D. Contoh Soal

1) Substitusi



$$T(n) = 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + f(n)$$

- $O(\text{tebakan}) = O(n \log n) \rightarrow$ karena mirip dengan algoritma heap sort pengerjaannya. Sehingga : $f(n) = n \log n$
- $T(n) \leq c(f(n))$
 $T(n) \leq c(n \log n)$
 $n = 2$, karena $\lfloor n/2 \rfloor = \lfloor 2/2 \rfloor = 1 \rightarrow$ kalau $n = 1$, $\lfloor n/2 \rfloor = \lfloor 1/2 \rfloor = 0$
 sehingga tidak memenuhi
 $T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) \leq c\left(\left\lfloor \frac{n}{2} \right\rfloor\right) \log \left\lfloor \frac{n}{2} \right\rfloor \rightarrow$ Substitusikan ke rekurensi
 $T(n) \leq c(n \log n) + cn$
 $T(n) \leq 2(c\left(\left\lfloor \frac{n}{2} \right\rfloor\right) \log \left(\left\lfloor \frac{n}{2} \right\rfloor\right)) + cn \rightarrow$ kenapa 2 didepan? Karena di persamaan awal $2T$

$$T(n) \leq cn \log \frac{n}{2} + cn$$

$$T(n) = cn \log n - cn \log 2 + cn \rightarrow \text{sifat algoritma}$$

$$T(n) = cn \log n - cn + cn$$

$T(n) = cn \log n \rightarrow -cn + cn$ diabaikan karena nilainya kecil sehingga tidak berpengaruh besar. Sehingga terbukti bahwa Big-Omerge sort adalah $O(n \log n)$

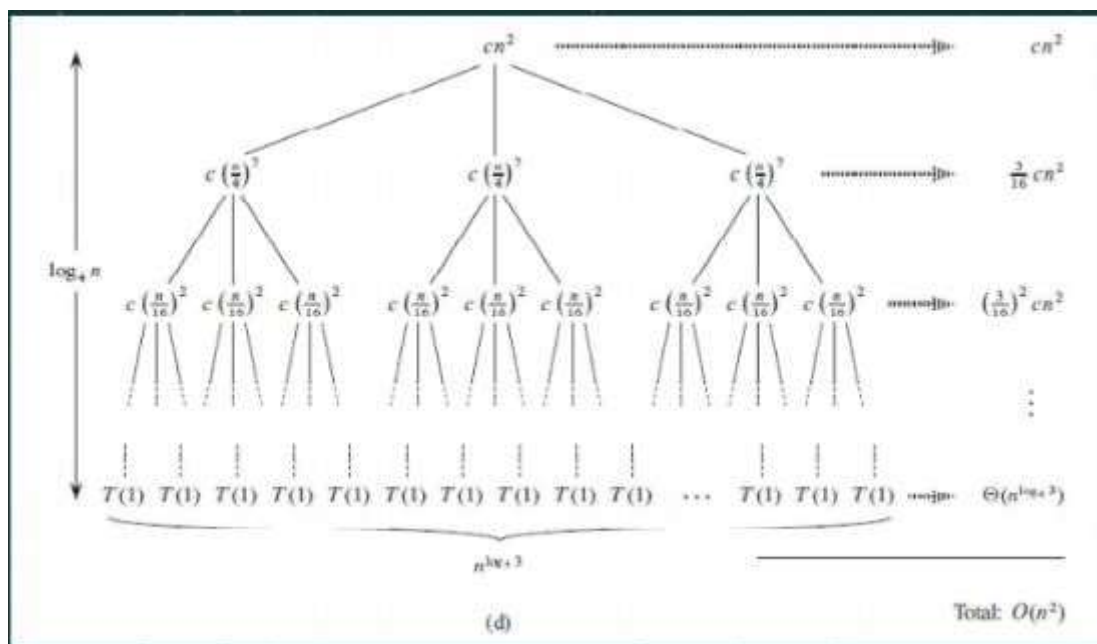
2) Recursion Tree

Temukan upper bound dari rekurensi berikut:

$$T(n) = 3T\left(\left\lfloor \frac{n}{4} \right\rfloor\right) + \Theta(n^2)$$

Jawab

- Membuat recursion tree sesuai fungsi
- Menghitung jumlah seluruh nilai di setiap kedalaman dengan sigma



Cost untuk keseluruhan tree, yaitu:

$$\begin{aligned} T(n) &= cn^2 + \frac{3}{16}cn^2 + \left(\frac{3}{16}\right)^2 cn^2 + \dots + \left(\frac{3}{16}\right)^{\log_4 n - 1} cn^2 + \Theta(n^{\log_4 3}) \\ &= \sum_{i=0}^{\log_4 n - 1} \left(\frac{3}{16}\right)^i cn^2 + \Theta(n^{\log_4 3}) \\ &< \sum_{i=0}^{\infty} \left(\frac{3}{16}\right)^i cn^2 + \Theta(n^{\log_4 3}) \end{aligned}$$

$$\begin{aligned}
&= \frac{1}{1 - \left(\frac{3}{16}\right)} cn^2 + \Theta(n^{\log_4 3}) \\
&= \frac{16}{13} cn^2 + \Theta(n^{\log_4 3}) \\
&= O(n^2)
\end{aligned}$$

3) Master

Tentukan Big- Θ dari rekurensi berikut:

$$T(n) = 9T\left(\frac{n}{3}\right) + n$$

Jawab

$$a = 9; b = 3; f(n) = n$$

$$n^{(\log_b a)} = n^{(\log_3 9)}$$

$$= n^2$$

$$f(n) = n = \Theta(n^{(\log_3 9 - \varepsilon)})$$

$$= \Theta(n^{(\log_3 9 - 1)}) \rightarrow \text{untuk } \varepsilon = 1$$

$$= \Theta(n^{(\log_3 8)})$$

$$= \Theta(n^{(1,89)})$$

$$= \Theta(n^2)$$

E. Bagian Analisis di Modul Praktikum

1. Merge Sort

Setelah Anda mengetahui Algoritma Merge-Sort mengadopsi paradigma divide & conquer, lakukan hal berikut:

- Buat program Merge-Sort dengan bahasa C++
- Kompleksitas waktu algoritma merge sort adalah $O(n \lg n)$. Cari tahu kecepatan computer Anda dalam memproses program. Hitung berapa running time yang dibutuhkan apabila input untuk merge sort-nya adalah 20?

Jawaban:

- /*

Nama Program	: Merge-Sort
Nama	: Fadlan Mulya Priatna
NPM	: 140810180041
Tanggal Pembuatan	: 23 Maret 2020
Deskripsi Program	: Koding program berisi merge sort dalam bahasa C++

```

*/

#include <iostream>
using namespace std;

int a[50];

void merge(int low,int mid,int high){
    int h,i,j,b[50],k;
    h=low;
    i=low;
    j=mid+1;
    while((h<=mid)&&(j<=high)){
        if(a[h]<=a[j]){
            b[i]=a[h]; h++;
        }
        else{
            b[i]=a[j]; j++;
        }
        i++;
    }

    if(h>mid){
        for(k=j;k<=high;k++){
            b[i]=a[k]; i++;
        }
    }
    else{
        for(k=h;k<=mid;k++){
            b[i]=a[k]; i++;
        }
    }
    for(k=low;k<=high;k++)
        a[k]=b[k];
}

void merge_sort(int low,int high){
    int mid;
    if(low<high){
        mid=(low+high)/2;
        merge_sort(low,mid);
        merge_sort(mid+1,high);
        merge(low,mid,high);
    }
}

int main(){
    int num,i;

    cout<<"Masukkan jumlah elemen yang akan diurutkan: ";cin >> num;

```

```

        cout<<endl;

        for(i=1;i<=num;i++){
            cout<<"Elemen "<<i<<" :";cin>>a[i] ;
        }

        merge_sort(1,num);

        cout << "\n-----" << endl;
        cout<<"Merge Sort :"<<endl;
        cout<<endl;

        for(i=1;i<=num;i++){
            cout<<a[i]<<" ";
            cout<<endl<<endl<<endl<<endl;
        }
    }
}

```

b. $O \rightarrow T(20 \log_{10} 20) = 26$

2. Selection Sort

Selection sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma selection sort, lakukan langkah-langkah berikut:

- Pelajari cara kerja algoritma selection sort
- Tentukan $T(n)$ dari rekurensi (pengulangan) selection sort berdasarkan penentuan rekurensi divide & conquer:

$$T(n) = \begin{cases} \Theta(1) & \text{if } n \leq c \\ aT\left(\frac{n}{b}\right) + D(n) + C(n) & \text{otherwise} \end{cases}$$

- Selesaikan persamaan rekurensi $T(n)$ dengan **metode recursion-tree** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ
- Lakukan implementasi koding program untuk algoritma selection sort dengan menggunakan bahasa C++

Jawaban:

```

for i ← n downto 2 do {pass sebanyak n-1 kali}
  imaks ← 1
  for j ← 2 to i do
    if  $x_j > x_{\text{imaks}}$  then
      imaks ← j
    endif
  endfor
  {pertukarkan  $x_{\text{imaks}}$  dengan  $x_i$ }
  temp ←  $x_i$ 
   $x_i$  ←  $x_{\text{imaks}}$ 
   $x_{\text{imaks}}$  ← temp
endfor

```

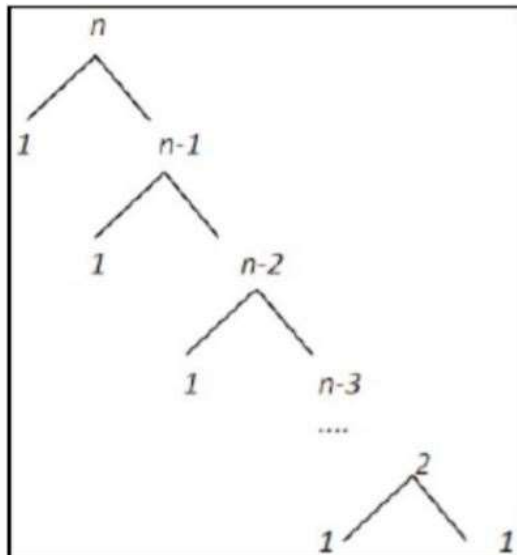
Subproblem = 1

Masalah setiap subproblem = $n-1$

Waktu proses pembagian = n

Waktu proses penggabungan = n

$$T(n) = \{\Theta(1)T(n-1) + \Theta(n)\}$$



$$T(n) = cn + cn - c + cn - 2c + \dots + 2c + cn$$

$$= c \left(\frac{(n-1)(n-2)}{2} \right) + cn$$

$$= c \left(\frac{n^2 - 3n + 2}{2} \right) + cn$$

$$= c \left(\left(\frac{n^2}{2} \right) - \left(\frac{3n}{2} \right) + 1 \right) + cn$$

$$= O(n^2)$$

$$T(n) = cn + cn - c + cn - 2c + \dots + 2c + cn$$

$$\begin{aligned}
&= c \left(\frac{(n-1)(n-2)}{2} \right) + cn \\
&= c \left(\frac{n^2 - 3n + 2}{2} \right) + cn \\
&= c \left(\left(\frac{n^2}{2} \right) - \left(\frac{3n}{2} \right) + 1 \right) + cn \\
&= \Omega(n^2)
\end{aligned}$$

$$T(n) = cn^2 = \Theta(n^2)$$

Kodingan:

```

/*
Nama Program      : Selection sort
Nama              : Fadlan Mulya Priatna
NPM              : 140810180041
Tanggal Pembuatan : 23 Maret 2020
Deskripsi Program : Koding program untuk algoritma selection sort dengan
menggunakan bahasa C++
*/
#include <iostream>
// #include <conio.h>

using namespace std;

int data[100], data2[100];
int n;

void tukar(int a, int b){
    int t;
    t = data[b];
    data[b] = data[a];
    data[a] = t;
}

void selection_sort(){
    int pos, i, j;
    for(i=1; i<=n-1; i++){
        pos = i;
        for(j = i+1; j<=n; j++){
            if(data[j] < data[pos]) pos = j;
        }
        if(pos != i) tukar(pos, i);
    }
}

int main(){
    cout<<"\nMasukkan jumlah elemen yang mau diurutkan: "; cin>>n;
    for(int i=1; i<=n; i++){

```

```

        cout<<"Masukkan elemen "<<i<<": ";
        cin>>data[i];
        data2[i]=data[i];
    }

    selection_sort();
    cout<<"Sorted Data: "<<endl;
    for(int i=1; i<=n; i++){
        cout<<" "<<data[i];
    }
}

```

3. Insertion Sort

Insertion sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma insertion sort, lakukan langkah-langkah berikut:

- Pelajari cara kerja algoritma insertion sort
- Tentukan $T(n)$ dari rekurensi (pengulangan) insertion sort berdasarkan penentuan rekurensi divide & conquer:

$$T(n) = \begin{cases} \Theta(1) & \text{if } n \leq c \\ aT\left(\frac{n}{b}\right) + D(n) + C(n) & \text{otherwise} \end{cases}$$

- Selesaikan persamaan rekurensi $T(n)$ dengan **metode substitusi** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ
- Lakukan implementasi koding program untuk algoritma insertion sort dengan menggunakan bahasa C++

Jawaban:

Algoritma

```

for i ← 2 to n do
    insert ← xi
    j ← i
    while (j < i) and (x[j-1] > insert) do
        x[j] ← x[j-1]
        j ← j-1
    endwhile
    x[j] = insert
endfor

```

Subproblem	= 1
Masalah setiap subproblem	= n-1
Waktu proses penggabungan	= n
Waktu proses pembagian	= n

$$T(n) = \{\Theta(1)T(n-1) + \Theta(n)\}$$

$$T(n) = cn + cn - c + cn - 2c + \dots + 2c + cn \Leftarrow 2cn^2 + cn^2$$

$$= c \left(\frac{(n-1)(n-2)}{2} \right) + cn \quad \Leftarrow 2cn^2 + cn^2$$

$$= c \left(\frac{n^2 - 3n + 2}{2} \right) + cn \quad \Leftarrow 2cn^2 + cn^2$$

$$= c \left(\left(\frac{n^2}{2} \right) - \left(\frac{3n}{2} \right) + 1 \right) + cn \quad \Leftarrow 2cn^2 + cn^2$$

$$= O(n^2)$$

$$T(n) = cn \Leftarrow cn$$

$$= \Omega(n)$$

$$T(n) = \frac{cn + cn^2}{n} = \Theta(n)$$

Kodingan:

```

/*
Nama Program      : Insertion sort
Nama              : Fadlan Mulya Priatna
NPM               : 140810180041
Tanggal Pembuatan : 23 Maret 2020
Deskripsi Program : Koding program untuk algoritma insertion sort dengan
menggunakan bahasa C++
*/
#include <iostream>

using namespace std;

int data[100],data2[100],n;

void insertion_sort(){
    int temp,i,j;
    for(i=1;i<=n;i++){
        temp = data[i];
        j = i -1;
        while(data[j]>temp && j>=0){
            data[j+1] = data[j];
            j--;
        }
        data[j+1] = temp;
    }
}

int main(){
    cout<<"Masukkan jumlah elemen yang mau diurutkan: "; cin>>n;

```

```

cout<<endl;

for(int i=1;i<=n;i++){
    cout<<"masukkan elemen "<<i<<": ";
    cin>>data[i];
    data2[i]=data[i];
}

insertion_sort();
cout<<"\nSorted Data: "<<endl;
for(int i=1; i<=n; i++){
    cout<<data[i]<<" ";
}
}

```

4. Bubble Sort

Bubble sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma bubble sort, lakukan langkah-langkah berikut:

- Pelajari cara kerja algoritma bubble sort
- Tentukan $T(n)$ dari rekurensi (pengulangan) insertion sort berdasarkan penentuan rekurensi divide & conquer:

$$T(n) = \begin{cases} \Theta(1) & \text{if } n \leq c \\ aT\left(\frac{n}{b}\right) + D(n) + C(n) & \text{otherwise} \end{cases}$$

- Selesaikan persamaan rekurensi $T(n)$ dengan **metode master** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ
- Lakukan implementasi koding program untuk algoritma bubble sort dengan menggunakan bahasa C++

Jawaban:

Algoritma

```

swapped = false
for i ← 1 to n-1 do
    if array[i-1] > array[i] then
        swap array[i-1] and array[i]
        swapped = true
    endif
endfor

```

Subproblem = 1

Masalah setiap subproblem = n-1

Waktu proses pembagian = n

Waktu proses penggabungan = n

$$T(n) = \{\Theta(1)T(n-1) + \Theta(n)\}$$

$$T(n) = cn + cn - c + cn - 2c + \dots + 2c + cn \Leftarrow 2cn^2 + cn^2$$

$$= c \left(\frac{(n-1)(n-2)}{2} \right) + cn \Leftarrow 2cn^2 + cn^2$$

$$= c \left(\frac{n^2 - 3n + 2}{2} \right) + cn \Leftarrow 2cn^2 + cn^2$$

$$= c \left(\left(\frac{n^2}{2} \right) - \left(\frac{3n}{2} \right) + 1 \right) + cn \Leftarrow 2cn^2 + cn^2$$

$$= O(n^2)$$

$$T(n) = cn + cn - c + cn - 2c + \dots + 2c + cn \Leftarrow 2cn^2 + cn^2$$

$$= c \left(\frac{(n-1)(n-2)}{2} \right) + cn \Leftarrow 2cn^2 + cn^2$$

$$= c \left(\frac{n^2 - 3n + 2}{2} \right) + cn \Leftarrow 2cn^2 + cn^2$$

$$= c \left(\left(\frac{n^2}{2} \right) - \left(\frac{3n}{2} \right) + 1 \right) + cn \Leftarrow 2cn^2 + cn^2$$

$$= \Omega(n^2)$$

$$T(n) = cn^2 + cn^2$$

$$= \Theta(n^2)$$

Kodingan:

```
/*
```

```
Nama Program      : Selection sort
```

```
Nama              : Fadlan Mulya Priatna
```

```
NPM               : 140810180041
```

```
Tanggal Pembuatan : 23 Maret 2020
```

```
Deskripsi Program : Koding program untuk algoritma selection sort dengan  
menggunakan bahasa C++
```

```
*/
```

```
#include <iostream>
```

```
using namespace std;
```

```
int main(){
```

```
    int arr[100],n,temp;
```

```
    cout<<" Masukkan jumlah elemen yang mau diurutkan: ";cin>>n;
```

```
    for(int i=0;i<n;++i){
```

```

        cout<<"Masukkan elemen "<<i+1<<" : ";cin>>arr[i];
    }

    for(int i=1;i<n;i++){
        for(int j=0;j<(n-1);j++){
            if(arr[j]>arr[j+1]){
                temp=arr[j];
                arr[j]=arr[j+1];
                arr[j+1]=temp;
            }
        }
    }

    cout<<"\nSorted Data: "<<endl;
    for(int i=0;i<n;i++){
        cout<<" "<<arr[i];
    }
}

```