

**STRUKTUR DATA**  
**INSERTION SORT DAN SELECTION SORT**

**(Dosen: Heti Mulyani, M.Kom)**

**LAPORAN PRAKTIKUM**

Untuk memenuhi tugas mata kuliah Struktur Data mengenai Pemrograman Insertion  
Sort dan Selection Sort



di susun oleh:

**Nopi Rahmawati (201904005)**

**Vira Virginia (201904021)**

**Luthfiyah Sakinah (201904024)**

**Ayu Siti Rohmah (201904016)**

**Adila Alaina Risqi (201904027)**

**PRODI TEKNOLOGI REKAYASA PERANGKAT LUNAK**  
**POLITEKNIK ENJINERING INDORAMA**  
**PURWAKARTA**  
**2019/2020**

## DAFTAR ISI

DAFTAR ISI.....	i
DAFTAR GAMBAR .....	ii
BAB I LANDASAN TEORI .....	1
1.1    Sorting (Pengurutan) .....	1
1.2    Insertion Sort .....	1
1.3    Selection Sort .....	2
BAB II IMPLEMENTASI .....	3
2.1    Insertion Sort .....	3
2.2    Selection Sort .....	16
BAB III PENUTUP .....	28
3.1    Kesimpulan.....	28
3.2    Saran .....	28
DAFTAR PUSTAKA .....	29

## DAFTAR GAMBAR

Gambar 2.1 File Header Insertion Sort .....	8
Gambar 2.2 Ascending dan Descending Angka Insertion Sort.....	10
Gambar 2.3 Insertion Sort Output Angka .....	11
Gambar 2.4 <i>Insertion Sort</i> Angka Pilihan.....	11
Gambar 2.5 Ascending Kata Insertion Sort .....	12
Gambar 2.6 Descending Kata dan Insertion Sort Output.....	13
Gambar 2.7 <i>Insertion Sort</i> Kata Pilihan.....	14
Gambar 2.8 Insertion Sort int main.....	15
Gambar 2.9 File Header Selection Sort.....	21
Gambar 2.10 Selection Sort Output dan Ascending Angka.....	22
Gambar 2.11 Descending Angka Selection Sort.....	23
Gambar 2.12 <i>Selection Sort</i> Angka Pilihan dan <i>Output</i> Kata .....	24
Gambar 2.13 Ascending Kata Selection Sort.....	25
Gambar 2.14 Descending Kata Selection Sort.....	26
Gambar 2.15 <i>Selection Sort</i> Kata Pilihan.....	27
Gambar 2.16 Selection Sort int main .....	27

# **BAB I**

## **LANDASAN TEORI**

### ***1.1 Sorting (Pengurutan)***

Pengurutan (*sorting*) adalah proses mengatur sekumpulan objek menurut urutan atau susunan tertentu. Urutan objek tersebut dapat menaik (*ascending*), yaitu urutan objek yang disusun mulai dari Nilai terkecil hingga terbesar atau menurun (*descending*), yaitu urutan objek yang disusun mulai dari Nilai terbesar hingga terkecil.

Pengurutan naik (*ascending*) mengurutkan data dari nilai yang terkecil atau yang terendah ke nilai yang lebih besar/tinggi.

Pengurutan turun (*descending*) adalah kebalikan dari pengurutan ascending dimana data akan diurutkan dari yang terbesar ke yang terkecil.

### ***1.2 Insertion Sort***

Insertion sort adalah sebuah algoritma pengurutan yang membandingkan dua elemen data pertama, mengurutkannya, kemudian mengecek elemen data berikutnya satu persatu dan membandingkannya dengan elemen data yang telah diurutkan.

Adapun kelebihan dan kekurangan yang dimiliki Insertion Sort yaitu sebagai berikut ini:

1. Kelebihan
  - a) Sederhana dalam penerapannya.
  - b) Mangkus dalam data yang kecil.
  - c) Jika list sudah terurut atau sebagian terurut maka Insertion Sort akan lebih cepat dibandingkan dengan Quicksort.
  - d) Mangkus dalam data yang sebagian sudah terurut.
  - e) Lebih mangkus dibanding Bubble Sort dan Selection Sort.
  - f) Loop dalam pada Inserion Sort sangat cepat, sehingga membuatnya salah satu algoritma pengurutan tercepat pada jumlah elemen yang sedikit.
  - g) Stabil.

## 2. Kekurangan

- a) Banyaknya operasi yang diperlukan dalam mencari posisi yang tepat untuk elemen larik.
- b) Untuk larik yang jumlahnya besar ini tidak praktis.
- c) Jika list terurut terbalik sehingga setiap eksekusi dari perintah harus memindai dan mengganti seluruh bagian sebelum menyisipkan elemen berikutnya.
- d) Membutuhkan waktu  $O(n^2)$  pada data yang tidak terurut, sehingga tidak cocok dalam pengurutan elemen dalam jumlah besar.

### ***1.3 Selection Sort***

*Selection sort* merupakan sebuah teknik pengurutan dengan cara mencari nilai tertinggi / terendah di dalam array kemudian menempatkan nilai tersebut di tempat semestinya. Algoritma ini dapat mengurutkan data dari besar ke kecil (*Ascending*) dan kecil ke besar (*Descending*). Algoritma ini tidak cocok untuk set data dengan jumlah besar karena kompleksitas dari algoritma ini adalah  $O(n^2)$  di mana  $n$  adalah jumlah item.

#### 1. Kelebihan

- a. Algoritma ini sangat rapat dan mudah untuk diimplementasikan.
- b. Operasi pertukarannya hanya dilakukan sekali saja.
- c. Waktu pengurutan dapat lebih ditekan.
- d. Mudah menggabungkannya kembali.
- e. Kompleksitas selection sort relatif lebih kecil.

#### 2. Kekurangan

- a. Membutuhkan method tambahan.
- b. Sulit untuk membagi masalah.

## BAB II

### IMPLEMENTASI

#### ***2.1 Insertion Sort***

##### *Source Code Ascending dan Descending Insertion Sort*

```
#include <iostream>
#include <string.h>

using namespace std;

int data[50], data2[50];
char kata[50][40], tmp[40];
int n,x;
//angka
void insertion_sortasc()
{
    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;
    }
}

void insertion_sortsc()
{
    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;
}
}
void insertion_output()
{
    for(int i=1; i<=n; i++)
    {
        cout<<"Data ["<<i<<" = ";
        cout<<data[i]<<endl;
    }
    cout << endl;
}

void insertion()
{
    cout << "Masukkan jumlah data: ";
    cin >> n;
    cout << endl;
    for(int i=1; i<=n; i++)
    {
        cout << "Masukkan data ke-" << i << ": ";
        cin >> data[i];
        data2[i] = data[i];
    }
    cout << "\nUrutkan Secara";
    cout << "\n1. Ascending \n2. Descending \n";
    cout << "Masukkan jenis yang dipilih: ";
    cin >> x;
    switch(x)
    {
    case 1:
    {
        insertion_sortasc();
        break;
    }
    case 2:

```

```

    {
        insertion_sortsc();
    }
}
cout << "\nData Setelah diurutkan" << endl;
cout << "-----" << endl;
insertion_output();
}
//kata
void insertion_kataasc()
{
    int i, j, nn, k, l;
    for (i=1; i<=n; i++)
    {
        cout<<"Kata ke-"<<i<<" = ";
        cin>>kata[i];
        if (i>1)
        {
            for (j=1; j<=(i-1); j++)
            {
                nn=(strcmp(kata[i], kata[j]));
                if (n<=0)
                {
                    strcpy (tmp, kata[i]);
                    for (k=(i-1); k>=j; k--)
                    {
                        l=(k+1);
                        strcpy (kata[l], kata[k]);
                    }
                    strcpy (kata[j], tmp);
                }
            }
        }
    }
}

void insertion_katasc()
{
    int i, j, nn, k, l;

```



```

for (i=1; i<=n; i++)
{
    cout<<"Kata ke-"<<i<<" = ";
    cin>>kata[i];
    if (i>1)
    {
        for (j=1; j<=(i-1); j++)
        {
            n=(strcmp(kata[i], kata[j]));
            if (n>=0)
            {
                strcpy (tmp, kata[i]);
                for (k=(i-1); k>=j; k--)
                {
                    l=(k+1);
                    strcpy (kata[l], kata[k]);
                }
                strcpy (kata[j], tmp);
            }
        }
    }
}

void output()
{
    cout<<"\nHasil pengurutan : \n";
    for (int i=1; i<=n; i++)
    {
        cout<<"Kata ke-"<<i;
        cout<<" = ";
        cout<<kata[i]<<endl;
    }
}

void insertionkata()
{
    cout << "Masukkan jumlah data: ";
    cin >> n;
    cout << endl;
}

```

```

for(int i=1; i<=n; i++)
{
    cout << "Masukkan data ke-" << i << ": ";
    cin >> data[i];
    data2[i] = data[i];
}
cout << "\nUrutkan Secara";
cout << "\n1. Ascending \n2. Descending \n";
cout << "Masukkan jenis yang dipilih: ";
cin >> x;
switch(x)
{
case 1:
{
    insertion_kataasc();
    break;
}
case 2:
{
    insertion_katasc();
}
}
cout << "\nData Setelah diurutkan" << endl;
cout << "-----" << endl;
output();
}

int main()
{
    cout << "\ninsertion";
    cout << "\n1. insertion angka \n2.insertion kata \n";
    cout << "Masukkan jenis yang dipilih: ";
    cin >> x;
    switch(x)
    {
    case 1:
    {
        insertion();
        break;
    }
    }
}

```

```

    }
    case 2:
    {
        insertionkata();
    }
}

```

Source Code diatas adalah Source Code Insertion Sort menggunakan pilihan angka dan kata. Insertion sort kata menggunakan string. Interaction Sort diatas memiliki pilihan Ascending dan Descending. Dibawah ini adalah penjelasan bagian-bagian dari Source Code.

```

1  #include <iostream>
2  #include <string.h>
3
4  using namespace std;

```

Gambar 2.1 File Header Insertion Sort

#Include<Iostream.h> digunakan untuk menampilkan perintah:

- Cin merupakan fungsi masukan(digunakan untuk menyimpan data dalam suatu variabel). Bentuk umum: cin>>var x;
- Cout merupakan fungsi keluaran(digunakan untuk menampilkan data ataupun tulisan). Bentuk umum: cout<<"tulisan"; atau cout<<var x;
- Endl digunakan untuk pindah baris/ enter.
- Ends merupakan suatu fungsi manipulator yang digunakan untuk menambah karakter null ( nilai ASCII NOL ) ke deretan suatu karakter. Fungsi ini akan berguna untuk mengirim sejumlah karakter ke file di disk atau modem dan mangakhirinya dengan karakter NULL.

`#Include<string.h>` merupakan file header yang berfungsi untuk melakukan manipulasi string. Fungsi-fungsi yang ada di `string.h` antara lain sebagai berikut :

- a. `strcpy()` : fungsi ini digunakan untuk menyalin suatu string ke variabel tujuan. Bentuk umum penulisannya adalah `strcpy(variabeltujuan, string);` .
- b. `strlen()` : fungsi ini digunakan untuk menghitung jumlah karakter yang ada dalam suatu string. Bentuk umum penulisannya adalah `strlen(string);` .
- c. `strcmp()` : fungsi ini digunakan untuk membandingkan 2 buah string. Bentuk umum penulisannya adalah `strcmp(string1,string2);` .
- d. `strrev()` : fungsi ini digunakan untuk membalikan urutan suatu string. Bentuk umum penulisannya adalah `strrev (string);` .
- e. `strlwr()` : fungsi ini digunakan untuk mengubah semua huruf menjadi huruf kecil. Bentuk umum penulisannya adalah `strlwr(sterng);` .
- f. `strupr()` : fungsi ini digunakan untuk mengubah semua huruf menjadi huruf kapital. Bentuk umum penulisannya adalah `strupr(string);` .
- g. `strcat()` : fungsi ini digunakan untuk menggabungkan 2 buah string, untuk menggunakan fungsi ini juga harus menambahkan file header `ctype.h`. Bentuk umum penulisannya adalah `strcat(variabeltujuan, string);` .

```

6      int data[50], data2[50];
7      char kata[50][40], tmp[40];
8      int n,x;
9      //angka
10     void insertion_sortasc()
11     {
12         int temp, i, j;
13         for(i=1; i<=n; i++)
14         {
15             temp = data[i];
16             j = i -1;
17             while(data[j]>temp && j>=0)
18             {
19                 data[j+1] = data[j];
20                 j--;
21             }
22             data[j+1] = temp;
23         }
24     }
25
26     void insertion_sortsc()
27     {
28         int temp, i, j;
29         for(i=1; i<=n; i++)
30         {
31             temp = data[i];
32             j = i -1;
33             while(data[j]<temp && j>0)
34             {
35                 data[j+1] = data[j];
36                 j--;
37             }
38             data[j+1] = temp;
39         }
40     }

```

Gambar 2.2 Ascending dan Descending Angka Insertion Sort

Pada Gambar 2.2 Ascending dan Descending Angka Source Code yang membedakan antara *Ascending* dan *Descending* adalah penggunaan tanda > lebih dari dan tanda < kurang dari. Tanda tersebut merupakan data untuk membedakan *Ascending* menggunakan tanda > dan *Descending* menggunakan tanda <. Untuk membedakannya terletak pada bagian *while* untuk Ascending adalah *while (data [j]<temp && j>0* sedangkan untuk Descending yaitu *while (data [j]<temp && j>0)* .

```

40 }
41 void insertion_output()
42 {
43     for(int i=1; i<=n; i++)
44     {
45         cout<<"Data ["<<i<<" = ";
46         cout<<data[i]<<endl;
47     }
48     cout << endl;
49 }
50

```

Gambar 2.3 Insertion Sort Output Angka

Pada Gambar 2.3 *Insertion Sort Output Angka* perintah diatas digunakan untuk menampilkan hasil pengurutan dari program yang telah diinputkan angka sesuai yang diinputkan.

```

50
51 void insertion()
52 {
53     cout << "Masukkan jumlah data: ";
54     cin >> n;
55     cout << endl;
56     for(int i=1; i<=n; i++)
57     {
58         cout << "Masukkan data ke-" << i << ": ";
59         cin >> data[i];
60         data2[i] = data[i];
61     }
62     cout << "\nUrutkan Secara";
63     cout << "\n1. Ascending \n2. Descending \n";
64     cout << "Masukkan jenis yang dipilih: ";
65     cin >> x;
66     switch(x)
67     {
68     case 1:
69     {
70         insertion_sortasc();
71         break;
72     }
73     case 2:
74     {
75         insertion_sortsc();
76     }
77     }
78     cout << "\nData Setelah diurutkan" << endl;
79     cout << "-----" << endl;
80     insertion_output();
81 }

```

Gambar 2.4 Insertion Sort Angka Pilihan

Pada Gambar 2.4 *Insertion Sort* Angka Pilihan digunakan untuk memilih program yang akan digunakan, yaitu memilih antara *Ascending* dan *Descending*. Setelah memilih maka program akan berjalan sesuai pilihan yaitu *Ascending* atau *Descending*.

```

82 // void
83 void insertion_kataasc()
84 {
85     int i, j, nn, k, l;
86     for (i=1; i<=n; i++)
87     {
88         cout<<"Kata ke-"<<i<<" = ";
89         cin>>kata[i];
90         if (i>1)
91         {
92             for (j=1; j<=(i-1); j++)
93             {
94                 nn=(strcmp(kata[i], kata[j]));
95                 if (nn<=0)
96                 {
97                     strcpy (tmp, kata[i]);
98                     for (k=(i-1); k>=j; k--)
99                     {
100                         l=(k+1);
101                         strcpy (kata[l], kata[k]);
102                     }
103                     strcpy (kata[j], tmp);
104                 }
105             }
106         }
107     }
108 }
109

```

Gambar 2.5 Ascending Kata Insertion Sort

Pada Gambar 2.5 *Ascending Kata Source Code Ascending* penggunaan tanda > lebih dari. Tanda tersebut merupakan data untuk membedakan *Ascending* dengan *Descending*. Untuk membedakannya terletak pada bagian *if* untuk *Ascending* adalah *if (i>1)*. Fungsi *strcmp()* digunakan untuk mengetahui atau membandingkan apakah dua string yang ditinjau itu sama atau tidak. Apabila sama, nilai balikan dari *strcmp()* samadengan 0. Jika tidak sama, maka nilai balikannya samadengan 1 sedangkan *strcpy()* berfungsi untuk menyalin String.

```

109
110 void insertion_kataasc()
111 {
112     int i, j, nn, k, l;
113     for (i=1; i<=n; i++)
114     {
115         cout<<"Kata ke-"<<i<<" = ";
116         cin>>kata[i];
117         if (i>1)
118         {
119             for (j=1; j<=(i-1); j++)
120             {
121                 n=(strcmp(kata[i], kata[j]));
122                 if (n>=0)
123                 {
124                     strcpy (tmp, kata[i]);
125                     for (k=(i-1); k>=j; k--)
126                     {
127                         l=(k+1);
128                         strcpy (kata[l], kata[k]);
129                     }
130                     strcpy (kata[j], tmp);
131                 }
132             }
133         }
134     }
135 }
136 void output()
137 {
138     cout<<"\nHasil pengurutan : \n";
139     for (int i=1; i<=n; i++)
140     {
141         cout<<"Kata ke-"<<i;
142         cout<<" = ";
143         cout<<kata[i]<<endl;
144     }
145 }

```

Gambar 2.6 Descending Kata dan Insertion Sort Output

Pada Gambar 2.6 *Descending Kata dan Insertion Sort Output*. *Source Code Descending* penggunaan tanda < kurang dari. Tanda tersebut merupakan data untuk membedakan *Descending* dengan *Ascending*. Untuk membedakannya terletak pada bagian *if* untuk *Descending* adalah *if (i>1)*. Fungsi *strcmp()* digunakan untuk mengetahui atau membandingkan apakah dua string yang ditinjau itu sama atau tidak. Apabila sama, nilai balikan dari *strcmp()* samadengan 0. Jika tidak sama, maka nilai balikannya samadengan 1 sedangkan *strcpy()* berfungsi untuk menyalin String.

Perintah pada bagian *void output* diatas digunakan untuk menampilkan hasil pengurutan dari program yang telah diinputkan angka sesuai yang diinputkan.



```

146
147 void insertionkata()
148 {
149     cout << "Masukkan jumlah data: ";
150     cin >> n;
151     cout << endl;
152     for(int i=1; i<=n; i++)
153     {
154         cout << "Masukkan data ke-" << i << ": ";
155         cin >> data[i];
156         data2[i] = data[i];
157     }
158     cout << "\nUrutkan Secara";
159     cout << "\n1. Ascending \n2. Descending \n";
160     cout << "Masukkan jenis yang dipilih: ";
161     cin >> x;
162     switch(x)
163     {
164     case 1:
165     {
166         insertion_kataasc();
167         break;
168     }
169     case 2:
170     {
171         insertion_katasc();
172     }
173     }
174     cout << "\nData Setelah diurutkan" << endl;
175     cout << "-----" << endl;
176     output();
177 }
178

```

Gambar 2.7 *Insertion Sort* Kata Pilihan

Pada Gambar 2.7 *Insertion Sort* Angka Pilihan digunakan untuk memilih program yang akan digunakan, yaitu memilih antara *Ascending* dan *Descending*. Setelah memilih maka program akan berjalan sesuai pilihan yaitu *Ascending* atau *Descending*.

```

179  int main()
180  {
181      cout << "\ninsertion";
182      cout << "\n1. insertion angka \n2.insertion kata \n";
183      cout << "Masukkan jenis yang dipilih: ";
184      cin >> x;
185      switch(x)
186      {
187          case 1:
188          {
189              insertion();
190              break;
191          }
192          case 2:
193          {
194              insertionkata();
195          }
196      }
197  }
198  }
199

```

Gambar 2.8 Insertion Sort int main

Pada Gambar 2.8 *Insertion Sort int main* adalah program utama yang akan di eksekusi. Int main() artinya main program mengembalikan nilai int secara default, int main() akan mengembalikan nilai 0, dan fungsi main() tidak memiliki bagan deklarasi lokal, dan hanya memiliki sebuah pernyataan yang dapat dieksekusi.

## 2.2 Selection Sort

### *Ascending dan Descending Selection Sort*

```
#include<iostream>
#include <string.h>

using namespace std;

int is,ns,js,ks,tmps;
int datas[50];
string nama[50],tmpss;

void outputselecangka()
{
    cout<<"\n setelah diurutkan akan menjadi : \n";
    for(is=0;is<ns;is++)
    {
        cout<<datas[is]<<" \n";
    }
}

void selectionangkaa()
{
    cout << "Masukkan jumlah data: ";
    cin >> ns;
    cout << endl;
    cout<<"mengurutkan nilai dari besar ke
kecil"<<endl<<endl;
    for(is=0;is<ns;is++)
    {
        cout<<"Masukkan nilai "<<is+1<<" :
";cin>>datas[is];
    }
    for(is=0;is<ns-1;is++)
    {
        ks=is;
        for(js=is+1;js<ns;js++)
        {
            if(datas[ks]>datas[js])
            {
                ks=js;
            }
        }
    }
}
```

```

        }
    }
    tmps=datas[ks];
    datas[ks]=datas[is];
    datas[is]=tmps;
}
outputselecangka();
}

void selectionangka()
{
    cout << "Masukkan jumlah data: ";
    cin >> ns;
    cout << endl;
    cout<<"mengurutkan nilai dari besar ke
kecil"<<endl<<endl;
    for(is=0;is<ns;is++)
    {
        cout<<"Masukkan nilai "<<is+1<<" :
";cin>>datas[is];
    }
    for(is=0;is<ns-1;is++)
    {
        ks=is;
        for(js=is+1;js<ns;js++)
        {
            if(datas[ks]<datas[js])
            {
                ks=js;
            }
        }
        tmps=datas[ks];
        datas[ks]=datas[is];
        datas[is]=tmps;
    }
    outputselecangka();
}

void angka()

```

```

{
    int x;
    cout << "selection angka \nUrutkan Secara";
    cout << "\n1. Ascending \n2. Descending \n";
    cout << "Masukkan jenis yang dipilih: ";
    cin >> x;
    switch(x)
    {
    case 1:
    {
        selectionangkaa();
        break;
    }
    case 2:
    {
        selectionangka();
    }
    }
}

void outputselect()
{
    cout<<"\n setelah diurutkan akan menjadi : \n";
    for(is=0; is<ns; is++)
    {
        cout<<nama[is]<<" \n";
    }
}

void selecta()
{
    cout<<"masukkan banyak data : ";
    cin>>ns;
    for (int is=0; is<ns; is++)
    {
        cout<<" Nama          : ";
        cin>>nama[is];
        cout<<endl;
    }
}

```

```

    }
    for (int is=0; is<ns; is++)
    {
        for (int js=is+1; js<ns; js++)
        {
            if (nama[is]< nama[js])
            {
                nama[is].swap (nama[js]);

            }
            tmpss=nama[is];
            nama[is]=nama[js];
            nama[js]=tmpss;
        }
    }
    outputselect();
}

void select()
{
    cout<<"masukkan banyak data : ";
    cin>>ns;
    for (int is=0; is<ns; is++)
    {
        cout<<" Nama          : ";
        cin>>nama[is];
        cout<<endl;
    }
    for (int is=0; is<ns; is++)
    {
        for (int js=is+1; js<ns; js++)
        {
            if (nama[is]> nama[js])
            {
                nama[is].swap (nama[js]);

            }
        }
    }
}

```

```

        tmpss=nama[is];
        nama[is]=nama[js];
        nama[js]=tmpss;

    }
}
outputselect();
}

void kata()
{
    int x;
    cout << "selection kata \nUrutkan Secara";
    cout << "\n1. Ascending \n2. Descending \n";
    cout << "Masukkan jenis yang dipilih: ";
    cin >> x;
    switch(x)
    {
    case 1:
    {
        selecta();
        break;
    }
    case 2:
    {
        select();
    }
    }
}

int main()
{
    int x;
    cout << "selection \nUrutkan Secara";
    cout << "\n1. selection angka \n2. selection kata
\n";
    cout << "Masukkan jenis yang dipilih: ";
    cin >> x;
    switch(x)
    {

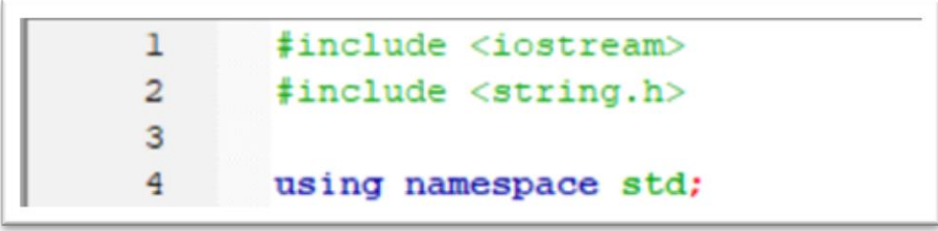
```

```

case 1:
{
    angka();
    break;
}
case 2:
{
    kata();
}
}

```

Source Code diatas adalah Source Code Selection Sort menggunakan pilihan angka dan kata. Selection Sort kata menggunakan string. Selection Sort diatas memiliki pilihan Ascending dan Descending. Dibawah ini adalah penjelasan bagian-bagian dari Source Code.



```

1  #include <iostream>
2  #include <string.h>
3
4  using namespace std;

```

Gambar 2.9 File Header Selection Sort

#Include<Iostream.h> digunakan untuk menampilkan perintah:

- Cin merupakan fungsi masukan(digunakan untuk menyimpan data dalam suatu variabel).
- Cout merupakan fungsi keluaran(digunakan untuk menampilkan data ataupun tulisan).
- Endl digunakan untuk pindah baris/ enter.
- Ends digunakan untuk menambah karakter null ( nilai ASCII NOL ) ke deretan suatu karakter.

#Include<string.h> merupakan file header yang berfungsi untuk melakukan manipulasi string.



```

6
7   int is,ns,js,ks,tmps;
8   int datas[50];
9   string nama[50],tmpss;
10
11 void outputselecangka()
12 {
13     cout<<"\n setelah diurutkan akan menjadi : \n";
14     for(is=0;is<ns;is++)
15     {
16         cout<<datas[is]<<" \n";
17     }
18 }
19 void selectionangkaa()
20 {
21     cout << "Masukkan jumlah data: ";
22     cin >> ns;
23     cout << endl;
24     cout<<"mengurutkan nilai dari besar ke kecil"<<endl<<endl;
25     for(is=0;is<ns;is++)
26     {
27         cout<<"Masukkan nilai "<<is+1<<" : ";cin>>datas[is];
28     }
29     for(is=0;is<ns-1;is++)
30     {
31         ks=is;
32         for(js=is+1;js<ns;js++)
33         {
34             if(datas[ks]>datas[js])
35             {
36                 ks=js;
37             }
38         }
39         tmps=datas[ks];
40         datas[ks]=datas[is];
41         datas[is]=tmps;
42     }
43     outputselecangka();
44 }
45

```

Gambar 2.10 Selection Sort Output dan Ascending Angka

Pada Gambar 2.10 *Selection Sort Output* dan *Ascending* Angka. Perintah pada bagian `void outputselecangka` diatas digunakan untuk menampilkan hasil pengurutan dari program yang telah diinputkan angka sesuai yang diinputkan.

*Source Code Ascending* penggunaan tanda `>` lebih dari. Tanda tersebut merupakan data untuk membedakan *Ascending* dengan *Descending*. Untuk membedakannya terletak pada bagian *if* untuk *Ascending* adalah `if (datas [ks]>datas[js])`.

```

45
46 void selectionangka()
47 {
48     cout << "Masukkan jumlah data: ";
49     cin >> ns;
50     cout << endl;
51     cout<<"mengurutkan nilai dari besar ke kecil"<<endl<<endl;
52     for(is=0;is<ns;is++)
53     {
54         cout<<"Masukkan nilai " <<is+1<<" : ";cin>>datas[is];
55     }
56     for(is=0;is<ns-1;is++)
57     {
58         ks=is;
59         for(js=is+1;js<ns;js++)
60         {
61             if(datas[ks]<datas[js])
62             {
63                 ks=js;
64             }
65         }
66         tmps=datas[ks];
67         datas[ks]=datas[is];
68         datas[is]=tmps;
69     }
70     outputseleangka();
71 }

```

Gambar 2.11 Descending Angka Selection Sort

Pada Gambar 2.11 *Descending Angka Selection Sort*. *Source Code Descending* penggunaan tanda < kurang dari. Tanda tersebut merupakan data untuk membedakan *Descending* dengan *Ascending*. Untuk membedakannya terletak pada bagian *if* untuk *Descending* adalah `if (datas [ks]<datas [js] )`.

```

72
73 void angka()
74 {
75     int x;
76     cout << "selection angka \nUrutkan Secara";
77     cout << "\n1. Ascending \n2. Descending \n";
78     cout << "Masukkan jenis yang dipilih: ";
79     cin >> x;
80     switch(x)
81     {
82     case 1:
83     {
84         selectionangkaa();
85         break;
86     }
87     case 2:
88     {
89         selectionangka();
90     }
91     }
92 }
93
94 void outputselect()
95 {
96     cout<<"\n setelah diurutkan akan menjadi : \n";
97     for(is=0; is<ns; is++)
98     {
99         cout<<nama[is]<<" \n";
100     }
101 }
102

```

Gambar 2.12 *Selection Sort* Angka Pilihan dan *Output* Kata

Pada Gambar 2.12 *Selection Sort* Angka Pilihan dan *Output* Kata. *Selection Sort* Pilihan digunakan untuk memilih program yang akan digunakan, yaitu memilih antara *Ascending* dan *Descending*. Setelah memilih maka program akan berjalan sesuai pilihan yaitu *Ascending* atau *Descending*.

Perintah pada bagian void *outputselec* diatas digunakan untuk menampilkan hasil pengurutan dari program yang telah diinputkan angka sesuai yang diinputkan.

```

102
103 void selecta()
104 {
105     cout<<"masukkan banyak data : ";
106     cin>>ns;
107     for (int is=0; is<ns; is++)
108     {
109         cout<<" Nama          : ";
110         cin>>nama[is];
111         cout<<endl;
112     }
113     for (int is=0; is<ns; is++)
114     {
115         for (int js=is+1; js<ns; js++)
116         {
117             if (nama[is]< nama[js])
118             {
119                 nama[is].swap (nama[js]);
120
121             }
122             tmpss=nama[is];
123             nama[is]=nama[js];
124             nama[js]=tmpss;
125         }
126     }
127     outputselect();
128 }
129

```

Gambar 2.13 Ascending Kata Selection Sort

*Source Code Ascending* penggunaan tanda < lebih dari. Tanda tersebut merupakan data untuk membedakan *Ascending* dengan *Descending*. Untuk membedakannya terletak pada bagian *if* untuk *Ascending* adalah `if (nama [ks]< nama[js]`.

```

130
131 void select()
132 {
133     cout<<"masukkan banyak data : ";
134     cin>>ns;
135     for (int is=0; is<ns; is++)
136     {
137         cout<<" Nama          : ";
138         cin>>nama[is];
139         cout<<endl;
140     }
141     for (int is=0; is<ns; is++)
142     {
143         for (int js=is+1; js<ns; js++)
144         {
145             if (nama[is]> nama[js])
146             {
147                 nama[is].swap (nama[js]);
148             }
149             tmpss=nama[is];
150             nama[is]=nama[js];
151             nama[js]=tmpss;
152         }
153     }
154     outputselect();
155 }
156
157
158

```

Gambar 2.14 Descending Kata Selection Sort

Pada Gambar 2.14 *Descending Kata Selection Sort*. *Source Code Descending* penggunaan tanda > kurang dari. Tanda tersebut merupakan data untuk membedakan *Descending* dengan *Ascending*. Untuk membedakannya terletak pada bagian *if* untuk *Descending* adalah `if (nama [is]>nama[js])`.

```

158
159 void kata()
160 {
161     int x;
162     cout << "selection kata \nUrutkan Secara";
163     cout << "\n1. Ascending \n2. Descending \n";
164     cout << "Masukkan jenis yang dipilih: ";
165     cin >> x;
166     switch(x)
167     {
168     case 1:
169     {
170         selecta();
171         break;
172     }
173     case 2:
174     {
175         select();
176     }
177     }
178 }
179

```

Gambar 2.15 *Selection Sort* Kata Pilihan

Pada Gambar 2.15 *Selection Sort* Kata Pilihan dan *Output* Kata. *Selection Sort* Pilihan digunakan untuk memilih program yang akan digunakan, yaitu memilih antara *Ascending* dan *Descending*. Setelah memilih maka program akan berjalan sesuai pilihan yaitu *Ascending* atau *Descending*.

```

179
180 int main()
181 {
182     int x;
183     cout << "selection \nUrutkan Secara";
184     cout << "\n1. selection angka \n2. selection kata \n";
185     cout << "Masukkan jenis yang dipilih: ";
186     cin >> x;
187     switch(x)
188     {
189     case 1:
190     {
191         angka();
192         break;
193     }
194     case 2:
195     {
196         kata();
197     }
198     }
199 }
200
201

```

Gambar 2.16 *Selection Sort* int main

Pada Gambar 2.16 *Selection Sort* int main adalah program utama yang akan di eksekusi.

## **BAB III**

### **PENUTUP**

#### **3.1 Kesimpulan**

Pada program yang telah dibuat diatas dapat disimpulkan teknik insertion sort dan selection sort sama-sama menghasilkan angka yang sama, output yang sama juga nilai yang sama tetapi memiliki teknik yang berbeda pada while dan if, pada insertion sort yaitu while dan pada selection sort yaitu if, pada kedua teknik tersebut digunakan untuk mengurutkan angka.

Program yang menggunakan ascending dan descending untuk mengurutkan angka yang lebih besar atau yang lebih kecil pada proses pengurutan angka yang telah diinputkan, teknik ini juga jauh memakan waktu lebih cepat dari pada teknik bubble short.

#### **3.2 Saran**

1. Sebagaimana program yang telah dibuat agar lebih dipahami program ini lebih diberikan keterangan pada kodingannya.
2. Untuk lebih memahami dalam program ini juga mencantumkan mencantumkan berbagai pengertian dari beberapa sumber.
3. Jika masih ada kesalahan, dengan penuh kerendahan hati mengharapkan saran dan kritik demi perbaikan dan semoga laporan ini bermanfaat bagi semua pihak terutama mahasiswa/mahasiswi Politeknik Enjineri Indorama.

## DAFTAR PUSTAKA

- [1] Pintarkom. (2019, 28 November). *Pengertian dan Contih Program Sorting pada C++*. Diakses pada 14 Juni 2020 dari <https://pintarkom.com/sorting-pada-c-plus/>
- [2] Dimas Setiawan. (2020, 23 Januari). *Contoh Program Bubble Sort C++*. Diakses pada 14 juni 2020 dari <https://kelasprogrammer.com/contoh-program-bubble-sort-cpp/>
- [3] Suryadinata Rismawan. (2017, 30 Maret). *Insertion Sort*. Diakses pada 14 juni 2020 dari <https://student.blog.dinus.ac.id/suryadinata/2017/03/30/insertion-sort/>
- [4] Cahyonanda.blogspot.com. (2019, 09 Maret). *Kelebihan dan Kekurangan Insertion Sort dan Selection Sort*. Diakses pada 14 Juni 2020 dari <http://chyonanda.blogspot.com/2012/03/kelebihan-dan-kekurangan-insertion-sort.html>
- [5] Binus.ac.id. (2019, 02 Desember). *Selection Sort*. Diakses pada 14 Juni 2020 dari <https://socs.binus.ac.id/2019/12/26/selection>