Nama : Muhammad Fikri Ramadhana

NIM : 21091397033

Kelas : A

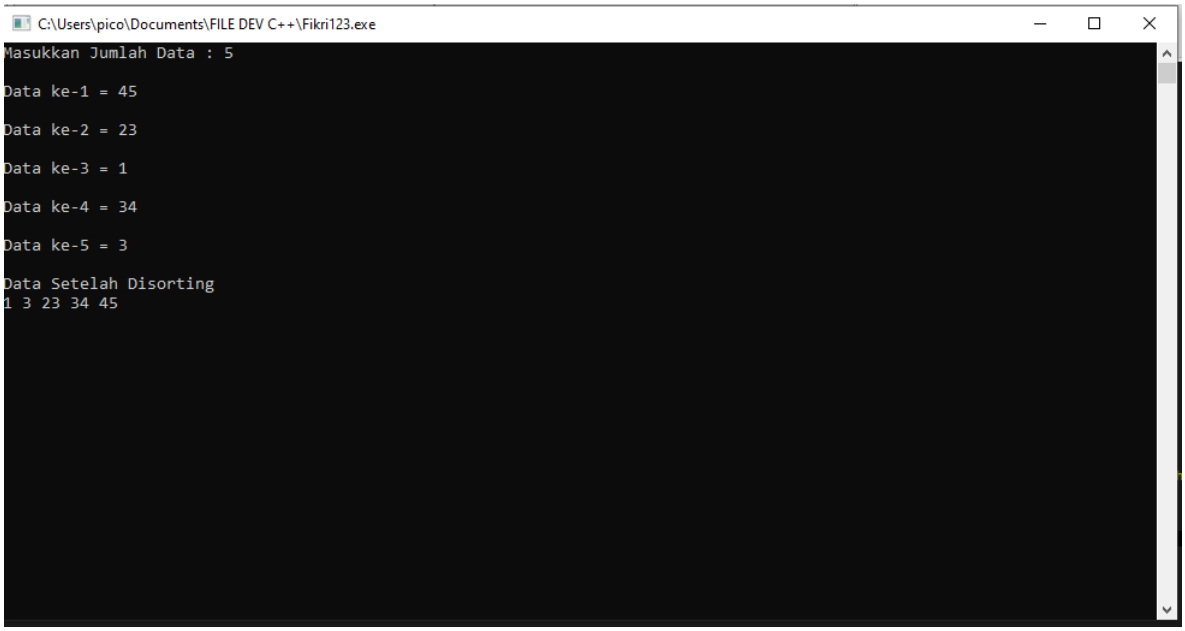
TUGAS STRUKTUR DATA INDIVIDU

A. Program Insertion Sort

Program ini menggunakan algoritma insertion sort yang berfungsi cukup efisien dalam mengurutkan sebuah data list.

```
//Pembuatan program insention sorting
#include <iostream>
#include <conio.h>
#include <stdlib.h>
using namespace std;

int main()
{
    //inialisasi variabel
    int kunci,a,x,array[100];

    //proses input jumlah data yang ingin dilakukan sorting
    cout<<"Masukkan Jumlah Data : "; cin>>x;
    cout<<endl;

    //proses input elemen data berdasarkan jumlah data yang dimasukkan
    for (int i=0;i<x;i++)
    {
        cout<<"Data ke-"<<i+1<<" = "; cin>>array[i];
        cout<<endl;
    }

    /* Proses data dengan operasi insertion sort dapat dilakukan
    dengan melakukan perbandingan elemen terbesar dengan elemen terkecil
    yang kemudian akan diurutkan dengan menyisipkan elemen sesuai dengan urutan */

    //proses pengecekan data awal sampai data ke-n
    for (int i=0;i<x-1;i++)
    {
        //proses perbandingan data sebelumnya dengan data selanjutnya sesuai urutan
        a=i;
        for (int j=1;j<x;j++)
        {
            kunci=array[j];
            i=j-1;
            //proses perbandingan data selanjutnya dengan data sebelumnya, jika data selanjutnya lebih kecil maka akan disisipkan pada data sebelumnya yang nilainya lebih besar
            while (array[i]>kunci&&i>=0)
            {
                array[i+1]=array[i];
                i--;
            }
            array[i+1]=kunci;
        }
    }

    //proses penampilan data yang telah dilakukan pengurutan dengan tipe selection sorting
    cout<<"Data Setelah Disorting "<<endl;
```

```
    int kunci,a,x,array[100];

    //proses input jumlah data yang ingin dilakukan sorting
    cout<<"Masukkan Jumlah Data : "; cin>>x;
    cout<<endl;

    //proses input elemen data berdasarkan jumlah data yang dimasukkan
    for (int i=0;i<x;i++)
    {
        cout<<"Data ke-"<<i+1<<" = "; cin>>array[i];
        cout<<endl;
    }

    /* Proses data dengan operasi insertion sort dapat dilakukan
    dengan melakukan perbandingan elemen terbesar dengan elemen terkecil
    yang kemudian akan diurutkan dengan menyisipkan elemen sesuai dengan urutan */

    //proses pengecekan data awal sampai data ke-n
    for (int i=0;i<x-1;i++)
    {
        //proses perbandingan data sebelumnya dengan data selanjutnya sesuai urutan
        a=i;
        for (int j=1;j<x;j++)
        {
            kunci=array[j];
            i=j-1;
            //proses perbandingan data selanjutnya dengan data sebelumnya, jika data selanjutnya lebih kecil maka akan disisipkan pada data sebelumnya yang nilainya lebih besar
            while (array[i]>kunci&&i>=0)
            {
                array[i+1]=array[i];
                i--;
            }
            array[i+1]=kunci;
        }
    }

    //proses penampilan data yang telah dilakukan pengurutan dengan tipe selection sorting
    cout<<"Data Setelah Disorting "<<endl;
    for (int i=0;i<x;i++)
    {
        cout<<array[i]<<" ";
    }

    getch();
    return 0;
}
```



Algoritma dengan tipe insertion sort menggunakan cara dengan mengambil elemen data pada list secara satu-satu yang kemudian akan dimasukkan ke dalam data list sesuai urutan posisi yang sebelumnya telah dilakukan pembandingan terhadap elemen saat itu dan elemen sebelumnya. Pada program pengurutan data dengan menggunakan insertion sort diatas dimulai dengan melakukan iniliasisasi variabel yang kemudian dilanjutkan pada proses input jumlah data yang ingin disorting. Setelah itu proses akan berlanjut pada tahapan proses memasukkan elemen data yang akan masuk pada jumlah data yang akan dilakukan sorting nantinya, kemudian proses berlanjut pada operasi insertion sort dengan tahapan dilakukannya proses pengecekan list data setelah itu akan dilanjutkan pada proses yang akan melakukan perbandingan data sebelumnya dengan data selanjutnya sesuai urutan. Langkah selanjutnya berupa pengurutan dan perpindahan data berdasarkan perbandingan yang telah dilakukan terhadap data sebelumnya dan data selanjutnya dengan fungsi jika data selanjutnya lebih kecil maka akan disisipkan pada data sebelumnya. Langkah terakhir ditutup dengan proses penampilan data yang telah dilakukan pengurutan data dengan tipe insertion sort.

Berdasarkan algoritma yang telah dibuat, kompleksitas intersention sort memiliki *best-case* jika sorting data list dapat terurutkan dengan benar dan *worst-case* jika data telat terurut namun urutannya terbalik. Sehingga pada kasus *worst-case* terdapat proses dimana setiap elemen data i lebih kecil dari elemen data 0 dan seterusnya sampai data ke i-1, kemudian setiap elemen data akan digeser atau dipindahkan setiap 1 langkah, hal tersebut dapat dinotasikan sebagai persamaan berikut :

$$T(n) = 1 + 2 + .. + n - 1 = \sum_{i=1}^{n-1} i = \frac{n(n-1)}{2} = O(n^2)$$

Sehingga kompleksitas dari insertion sort dapat dinotasikan dalam bentuk Big-O nya berupa $O(n^2)$, meskipun Big-O dari insertion sort dan selection sort hampir sama, namun tipe data insertion sort lebih cepat 40% dari selection sort.

Perhitungan Big-O

N	1	5	10
$O(n^2)$	1	25	100

Kelebihan Insertion Sort

- Memiliki penerapan yang lebih sederhana
- Mangkus dalam pemrosesan data yang kecil
- Looping yang dilakukan dalam insertion sort terjadi sangat cepat sehingga menjadikan algoritma sorting list tercepat jika jumlah elemen data atau array sedikit
- Lebih stabil dalam proses eksekusi pengurutannya

Kekurangan Insertion Sort

- Operasi yang terlalu banyak dalam mencari posisi yang tepat untuk elemen array
- Jika array berjumlah banyak maka algoritma ini tidak praktis dalam melakukan prosesnya
- Jika data list terurut dengan posisi terbalik maka program eksekusinya diharuskan memindai dan mengganti setiap bagian sebelum melakukan penyisipan elemen
- Tidak cocok digunakan dalam proses pengurutan elemen dengan data yang besar dikarenakan notasi Big-O nya $O(n^2)$.