Praktikum 6

Insertion sort dan Selection sort

Nama: Dio Stania Adinata

NRP: 5223600024

latihan 1

Input:

#include <iostream>

#include <vector>

#include <algorithm>

#include <cstdlib>

#include <ctime>

const int MAX = 10;

// Fungsi untuk melakukan penukaran

void Swap(int& a, int& b)

{

    int temp = a;

    a = b;

    b = temp;

}

//Fungsi untuk melakukan Insertion sort

void InsertionSort(std::vector<int>& data)

{

    for(int i = 1; i < MAX; i++)

    {

        int key = data[i];

        int j = i - 1;

        while(j >= 0 && data[j] > key)

        {

            data[j + 1] = data[j];

            j = j - 1;

        }

        data[j + 1] = key;

        //melakukan print hasil dari insertion sort

        std::cout << "Iteration " << i << ":\t";

        for(int k = 0; k < MAX; k++)

        {

            std::cout << data[k] << " ";

        }

        std::cout << "\n";

    }

}

// Fungsi untuk melakukan binary sort

void BinaryInsertionSort(std::vector<int>& data)

{

    for(int i = 1; i < MAX; i++)

    {

        int key = data[i];

        int left = 0;

        int right = i - 1;

        while(left <= right)

        {

            int mid = left + (right - left) / 2;

            if(data[mid] > key)

                right = mid - 1;

            else

                left = mid + 1;

        }

        for(int j = i - 1; j >= left; j--)

        {

            data[j + 1] = data[j];

        }

        data[left] = key;

        //melakukan print hasil insertion sort

        std::cout << "Iteration " << i << ":\t";

        for(int k = 0; k < MAX; k++)

        {

            std::cout << data[k] << " ";

        }

        std::cout << "\n";

    }

}

// fungsi untuk melakukan selection sort

void SelectionSort(std::vector<int>& data)

{

    for(int i = 0; i < MAX - 1; i++)

    {

        int minIndex = i;

        for(int j = i + 1; j < MAX; j++)

        {

            if(data[j] < data[minIndex])

            {

                minIndex = j;

            }

        }

        Swap(data[minIndex], data[i]);

        // Print iteration

        std::cout << "Iteration " << i+1 << ":\t";

        for(int k = 0; k < MAX; k++)

        {

            std::cout << data[k] << " ";

        }

        std::cout << "\n";

    }

}

int main()

{

    std::srand(std::time(0)); //menginisiasi random generator

    std::vector<int> data(MAX);

    std::cout << "DATA BEFORE SORTING:\n";

    for(int i = 0; i < MAX; i++)

    {

        data[i] = std::rand() % 1000 + 1;

        std::cout << "Data[" << i << "]: " << data[i] << "\n";

    }

    std::cout << "\nINSERTION SORT PROCESS:\n";

    InsertionSort(data);

    std::cout << "\nBINARY INSERTION SORT PROCESS:\n";

    BinaryInsertionSort(data);

    std::cout << "\nSELECTION SORT PROCESS:\n";

    SelectionSort(data);

    // print sorted data

    std::cout << "\nDATA AFTER SORTING:\n";

    for(int i = 0; i < MAX; i++)

    {

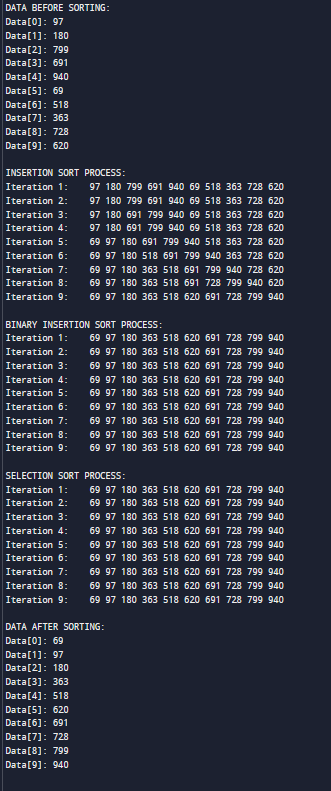
        std::cout << "Data[" << i << "]: " << data[i] << "\n";

    }

    return 0;

}

Output:



Latihan 2

Input:

#include <iostream>

#include <vector>

#include <cstdlib>

#include <ctime>

const int MAX = 10;

//Fungsi untuk melakukan penukaran

void Swap(int& a, int& b)

{

    int temp = a;

    a = b;

    b = temp;

}

// Fungsi untuk melakukan insertion sort dan melakukan pembandingan dan perhitungan dari perubahan yang ada

void InsertionSort(std::vector<int>& data, int& comparisons, int& shifts)

{

    for(int i = 1; i < MAX; i++)

    {

        int key = data[i];

        int j = i - 1;

        comparisons++;

        while(j >= 0 && data[j] > key)

        {

            data[j + 1] = data[j];

            j = j - 1;

            shifts++;

            comparisons++;

        }

        data[j + 1] = key;

    }

}

// Fungsi untuk melakukan binary sort dan melakukan pembandingan dan perhitungan dari perubahan yang ada

void BinaryInsertionSort(std::vector<int>& data, int& comparisons, int& shifts)

{

    for(int i = 1; i < MAX; i++)

    {

        int key = data[i];

        int left = 0;

        int right = i - 1;

        comparisons++;

        while(left <= right)

        {

            int mid = left + (right - left) / 2;

            if(data[mid] > key)

                right = mid - 1;

            else

                left = mid + 1;

            comparisons++;

        }

        for(int j = i - 1; j >= left; j--)

        {

            data[j + 1] = data[j];

            shifts++;

        }

        data[left] = key;

    }

}

// Fungsi untuk melakukan binary sort dan melakukan pembandingan dan perhitungan dari perubahan yang ada

void SelectionSort(std::vector<int>& data, int& comparisons, int& shifts)

{

    for(int i = 0; i < MAX - 1; i++)

    {

        int minIndex = i;

        for(int j = i + 1; j < MAX; j++)

        {

            comparisons++;

            if(data[j] < data[minIndex])

            {

                minIndex = j;

            }

        }

        Swap(data[minIndex], data[i]);

        shifts++;

    }

}

int main()

{

    std::srand(std::time(0)); //menginisiasi random data

    std::vector<int> data(MAX);

    std::cout << "DATA BEFORE SORTING:\n";

    for(int i = 0; i < MAX; i++)

    {

        data[i] = std::rand() % 1000 + 1;

        std::cout << "Data[" << i << "]: " << data[i] << "\n";

    }

    int comparisons\_insertion = 0, shifts\_insertion = 0;

    int comparisons\_binary = 0, shifts\_binary = 0;

    int comparisons\_selection = 0, shifts\_selection = 0;

    // Insertion Sort

    std::vector<int> data\_insertion = data;

    InsertionSort(data\_insertion, comparisons\_insertion, shifts\_insertion);

    // Binary Insertion Sort

    std::vector<int> data\_binary = data;

    BinaryInsertionSort(data\_binary, comparisons\_binary, shifts\_binary);

    // Selection Sort

    std::vector<int> data\_selection = data;

    SelectionSort(data\_selection, comparisons\_selection, shifts\_selection);

    // Print Results

    std::cout << "\nINSERTION SORT RESULTS:\n";

    std::cout << "Sorted Data:\n";

    for(int i = 0; i < MAX; i++)

    {

        std::cout << "Data[" << i << "]: " << data\_insertion[i] << "\n";

    }

    std::cout << "Comparisons: " << comparisons\_insertion << "\n";

    std::cout << "Shifts: " << shifts\_insertion << "\n";

    std::cout << "\nBINARY INSERTION SORT RESULTS:\n";

    std::cout << "Sorted Data:\n";

    for(int i = 0; i < MAX; i++)

    {

        std::cout << "Data[" << i << "]: " << data\_binary[i] << "\n";

    }

    std::cout << "Comparisons: " << comparisons\_binary << "\n";

    std::cout << "Shifts: " << shifts\_binary << "\n";

    std::cout << "\nSELECTION SORT RESULTS:\n";

    std::cout << "Sorted Data:\n";

    for(int i = 0; i < MAX; i++)

    {

        std::cout << "Data[" << i << "]: " << data\_selection[i] << "\n";

    }

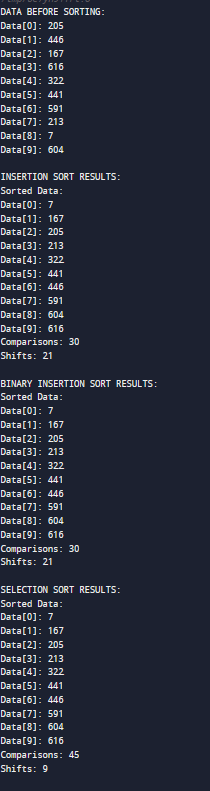
    std::cout << "Comparisons: " << comparisons\_selection << "\n";

    std::cout << "Shifts: " << shifts\_selection << "\n";

    return 0;

}

Output:



Latihan 3

Input:

#include <iostream>

#include <string>

#include <algorithm>

using namespace std;

struct Pegawai {

    int nip;

    string nama;

};

void tukar(Pegawai& a, Pegawai& b) {

    Pegawai temp = a;

    a = b;

    b = temp;

}

// Metode pengurutan: insertion sort

void insertionSort(Pegawai data[], int n, bool naik, string kriteria) {

    for (int i = 1; i < n; i++) {

        int j = i;

        Pegawai key = data[i];

        while (j > 0 && (naik ? (kriteria == "nip" ? data[j - 1].nip > key.nip : data[j - 1].nama > key.nama)

                                 : (kriteria == "nip" ? data[j - 1].nip < key.nip : data[j - 1].nama < key.nama))) {

            data[j] = data[j - 1];

            j--;

        }

        data[j] = key;

    }

}

int main() {

    const int MAX = 5;

    Pegawai data[MAX] = {

        {1, "Dio"},

        {5, "Nauval"},

        {3, "Rakha"},

        {2, "Jagur"},

        {4, "Rifath"}

    };

    int choice;

    bool naik;

    string kriteria;

    while (true) {

        cout << "Pilih metode pengurutan:\n";

        cout << "1. NIP naik\n";

        cout << "2. NIP turun\n";

        cout << "3. NAMA naik\n";

        cout << "4. NAMA turun\n";

        cout << "5. Keluar\n";

        cout << "Masukkan pilihan: ";

        cin >> choice;

        if (choice == 5) {

            break;

        }

        cout << "Masukkan pengurutan naik (true) atau turun (false): ";

        cin >> naik;

        if (choice == 1 || choice == 2) {

            kriteria = "nip";

        } else {

            kriteria = "nama";

        }

        insertionSort(data, MAX, naik, kriteria);

        cout << "Hasil pengurutan:\n";

        for (int i = 0; i < MAX; i++) {

            cout << "NIP: " << data[i].nip << ", NAMA: " << data[i].nama << endl;

        }

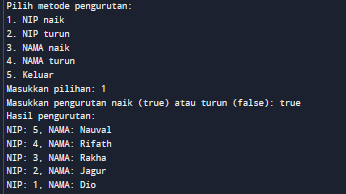
        cout << endl;

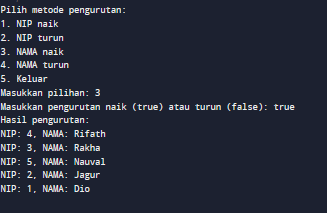
    }

    return 0;

}

Output:





Latihan 4

Melalui percobaan dan latihan, saya belajar dalam implementasi algoritma pengurutan data.

Saya mempelajari berbagai metode pengurutan seperti binary insertion sort, selection insertion sort, straight selection sort dan penerapannya pada data berdasarkan kriteria yang ditentukan.

Proses ini mengajarkan saya pentingnya efisiensi waktu, minimalisasi kompleksitas algoritma, dan penulisan kode yang mudah dipahami.