**LAPORAN PRAKTIKUM 4**

**Analisis algoritma**

****

**Sarah Navianti Dwi Sutisna**

**140810180021**

**Kelas A**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS PADJADJARAN**

**2020**

Studi Kasus

**Studi Kasus 1: MERGE SORT**

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

Hal berikut:

1. Buat program Merge-Sort dengan bahasa C++

/\*

Nama        : Sarah Navianti Dwi S

NPM         : 140810180021

Kelas       : A

Deskripsi   : Merge Short

 \*/

#include <iostream>

#include <chrono>

using namespace std;

using namespace std::chrono;

void mergeSort1 (int\* x, int rend, int ting, int mid){

    int i, j, k, temp[ting-rend+1];

    i = rend;

    k = 0;

    j = mid + 1;

    while (i <= mid && j <= ting){

        if (x[i] < x[j]){

            temp[k] = x[i];

            k++;

            i++;

        }

        else {

            temp[k] = x[j];

            k++;

            j++;

        }

    }

    while (i <= mid){

        temp[k] = x[i];

        k++;

        i++;

    }

    while (j <= ting){

        temp[k] = x[j];

        k++;

        j++;

    }

        for (i = rend; i <= ting; i++){

        x[i] = temp[i-rend];

    }

}

void mergeSort2(int \*x, int rend, int ting){

    int mid;

    if (rend < ting){

        mid=(rend+ting)/2;

        mergeSort2(x, rend, mid);

        mergeSort2(x, mid+1, ting);

        mergeSort1(x, rend, ting, mid);

    }

}

int main(){

    int jml, i;

    high\_resolution\_clock::time\_point t1 = high\_resolution\_clock::now();

    cout << "=============Merge Short==============="<<endl;

    cout <<"\nJumlah data yang ingin diurutkan : ";

    cin  >>jml;

    cout << "\n=======================================" << endl;

    int arr[jml];

    for(i = 0; i < jml; i++){

        cout<<"Masukkan elemen ke-"<<i+1<<": ";

        cin>>arr[i];

    }

    cout <<"=======================================" <<endl;

    mergeSort2(arr, 0, jml-1);

    cout<<"\nArray yang telah diurutkan: ";

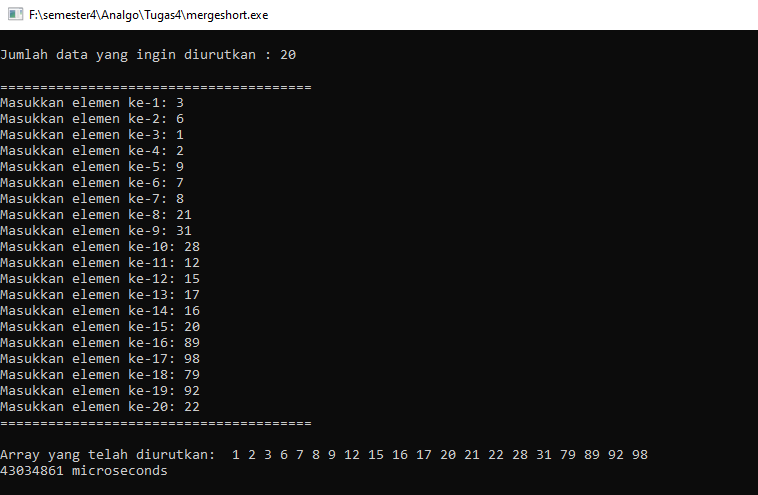
    for (i = 0; i < jml; i++) cout<<" "<<arr[i];

    high\_resolution\_clock::time\_point t2 = high\_resolution\_clock::now();

        auto duration = duration\_cast<microseconds>( t2 - t1 ).count();

        cout<<endl<<duration<<" microseconds"<<endl;

}

* Hasil Output jika dalam microseconds

1. Kompleksitas waktu algoritma merge sort adalah O(n lg n). Cari tahu kecepatan komputer Anda dalam memproses program. Hitung berapa running time yang dibutuhkan apabila input untuk merge sort-nya adalah 20?

Jawab :

Durasi yang dibutuhkan apabila input untuk merge sortnya 20 adalah : 43034861 = 43.0349 seconds jika dibulatkan.

Tetapi jika dalam rumusnya maka :

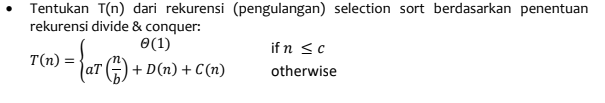
Big-O = Big-Ω = Big-θ = n \* log n

Maka : T(26

**Studi Kasus 2: SELECTION SORT**

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

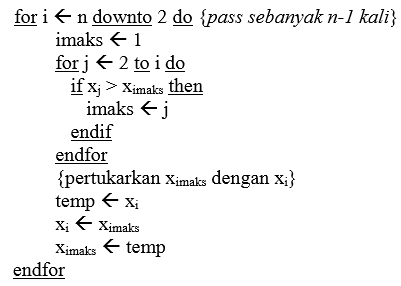
* Pelajari cara kerja algoritma selection sort





* 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++

Jawab :



Subproblem = 1

Masalah setiap subproblem = n-1

Waktu proses pembagian = n

Waktu proses penggabungan = n



T(n) = cn + cn-c +cn-2c + ..... + 2c +cn

= c((n-1)(n-2)/2) + cn

= c((n2-3n+2)/2) + cn

= c(n2/2)-(3n/2)+1 + cn

=O(n2)

T(n) = cn + cn-c +cn-2c + ..... + 2c +cn

= c((n-1)(n-2)/2) + cn

= c((n2-3n+2)/2) + cn

= c(n2/2)-(3n/2)+1 + cn

= Ω (n2)

T(n) = cn2

= Θ(n2)

/\*

Nama        : Sarah Navianti Dwi S

NPM         : 140810180021

Kelas       : A

Deskripsi   : Selection Sort

 \*/

#include <iostream>

#include <chrono>

using namespace std;

using namespace std::chrono;

void selectionSort (int arr[], int n){

    int i, j;

    for (i = 0; i < n; ++i){

        for (j = i+1; j < n; ++j){

            if (arr[i] > arr[j]){

                arr[i] = arr[i]+arr[j];

                arr[j] = arr[i]-arr[j];

                arr[i] = arr[i]-arr[j];

            }

        }

    }

}

int main(){

    int jml, i;

       // high\_resolution\_clock::time\_point t1 = high\_resolution\_clock::now();

        cout << "=============Selection Sort==============="<<endl;

        cout <<"\nJumlah elemen yang ingin diurutkan : ";

        cin  >>jml;

        cout << "=======================================" << endl;

    int arr[jml];

    for(i = 0; i < jml; i++){

        cout<<"Masukkan elemen ke-"<<i+1<<": ";

        cin>>arr[i];

    }

    selectionSort(arr, jml);

    cout<<"\nArray yang telah diurutkan: ";

    for (i = 0; i < jml; i++) cout<<" "<<arr[i];

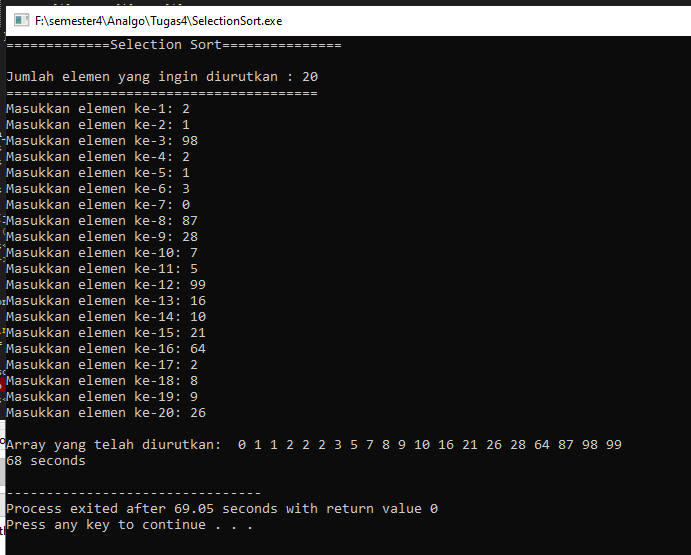
    //high\_resolution\_clock::time\_point t2 = high\_resolution\_clock::now();

      //auto duration = duration\_cast<seconds>( t2 - t1 ).count();

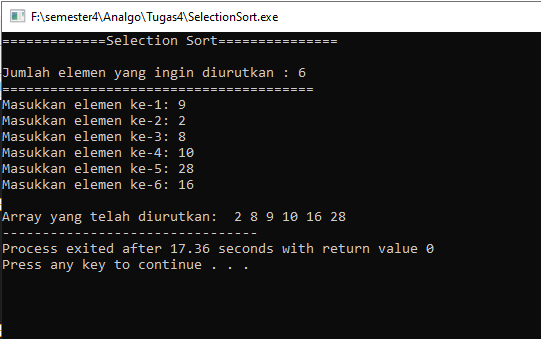
         //cout<<endl<<duration<<" seconds" <<endl;

}

* Hasil dari Selection Sort jika memakai kompleksitas waktu



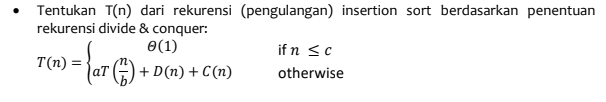
Jika tidak memakai kompleksitas waktu



**Studi Kasus 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

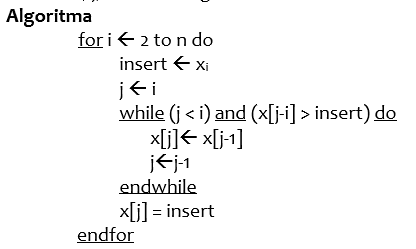




 Selesaikan persamaan rekurensi T(n) dengan **metode subtitusi** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ

* Lakukan implementasi koding program untuk algoritma insertion sort dengan menggunakan bahasa C++

Jawab :



Subproblem = 1

Masalah setiap subproblem = n-1

Waktu proses penggabungan = n

Waktu proses pembagian = n

T(n) = cn + cn-c +cn-2c + ..... + 2c +cn <= 2cn2 + cn2

= c((n-1)(n-2)/2) + cn<= 2cn2 + cn2

= c((n2-3n+2)/2) + cn<= 2cn2 + cn2

= c(n2/2)-c(3n/2)+c+cn <= 2cn2 + cn2

=O(n2)

T(n) = cn <= cn

= Ω (n)

T(n) = (cn + cn2)/n

= Θ(n)

/\*

Nama        : Sarah Navianti Dwi S

NPM         : 140810180021

Kelas       : A

Deskripsi   : Insertion Sort

 \*/

#include <iostream>

#include <conio.h>

using namespace std;

int dat[500],dat2[500],n;

void insertionSort(){

    int temp,i,j;

    for(i=1;i<=n;i++){

        temp = dat[i];

        j = i -1;

        while(dat[j]>temp && j>=0){

            dat[j+1] = dat[j];

            j--;

        }

        dat[j+1] = temp;

    }

}

int main(){

    cout << "=============Insertion Sort==============="<<endl;

    cout <<"\nJumlah elemen yang ingin diurutkan : ";

    cin  >>n;

    cout << "=======================================" << endl;

    for(int i=1;i<=n;i++)

    {

        cout<<"Masukkan data ke-"<<i<<" : ";

        cin>>dat[i];

        dat2[i]=dat[i];

    }

    cout << "\n=====================================" << endl;

    insertionSort();

    cout<<"\nData Setelah di Sort : "<<endl;

    for(int i=1; i<=n; i++)

    {

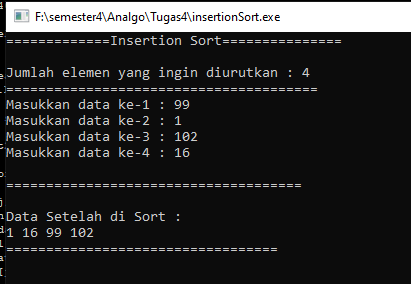
        cout<<dat[i]<<" ";

    }

    cout << "\n=================================="<<endl;

    getch();

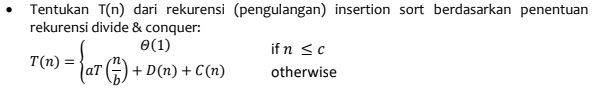
}

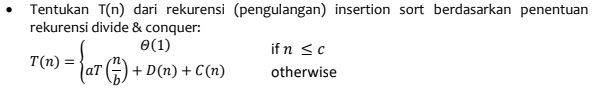


**Studi Kasus 4: BUBBLE SORT**

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





* 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++

jawab :

Subproblem = 1

Masalah setiap subproblem = n-1

Waktu proses pembagian = n

Waktu proses penggabungan = n

T(n) = cn + cn-c +cn-2c + ..... + 2c +c <= 2cn2 + cn2

= c((n-1)(n-2)/2) + c<= 2cn2 + cn2

= c((n2-3n+2)/2) + c<= 2cn2 + cn2

= c(n2/2)-c(3n/2)+2c <= 2cn2 + cn2

=O(n2)

T(n) = cn + cn-c +cn-2c + ..... + 2c +c <= 2cn2 + cn2

= c((n-1)(n-2)/2) + c<= 2cn2 + cn2

= c((n2-3n+2)/2) + c<= 2cn2 + cn2

= c(n2/2)-c(3n/2)+2c <= 2cn2 + cn2

= Ω (n2)

T(n) = cn2 + cn2

= Θ(n2)

/\*

Nama        : Sarah Navianti Dwi S

NPM         : 140810180021

Kelas       : A

Deskripsi   : Bubble Sort

 \*/

#include <iostream>

using namespace std;

void bubbleSort (int arr[], int n){

    int i, j;

    for (i = 0; i < n; ++i){

        for (j = 0; j < n-i-1; ++j){

            if (arr[j] > arr[j+1]){

                arr[j] = arr[j]+arr[j+1];

                arr[j+1] = arr[j]-arr[j + 1];

                arr[j] = arr[j]-arr[j + 1];

            }

        }

    }

}

int main(){

    int n, i;

    cout << "============= Bubble Sort ==============="<<endl;

    cout <<"\nJumlah elemen yang akan diinputkan : ";

    cin  >>n;

    cout << "=======================================" << endl;

    int arr[n];

    for(i = 0; i < n; i++){

        cout<<"Masukkan elemen ke-"<<i+1<<": ";

        cin>>arr[i];

    }

    bubbleSort(arr, n);

    cout<<"\nHasil Bubble Sort yang telah diurutkan: ";

    for (i = 0; i < n; i++){

            cout<<" "<<arr[i];

    }

}

