

Laporan Individu Selection Sort

By : Shandy Ilham Alamsyah | 21091397015

○ Kode C++ Selection Sort

- Function swap untuk algoritma penukaran indeks array

```
// function untuk menukar array
void swap(int *xp, int *yp) {
    int temp = *xp;                                // temp =
    tempat untuk menyimpan array yang akan dibandingkan
    *xp = *yp;
    *yp = temp;
}
```

- Function selectionSort yang berisi algoritma sorting

```
// function yang berisi algoritma selection sort
void selectionSort(int array[], int n) {
    int i, j, minimum;

    //proses seleksi satu per satu
    for (i = 0; i < n-1; i++) {
        //untuk menemukan nilai minimum dari array yang belum di
        sorting
        minimum = i;
        for (j = i+1; j < n; j++)
            if (array[j] < array[minimum])
                minimum = j;

        // untuk menukar tempat atau indeks jika ditemukan angka yang
        lebih kecil dalam array
        swap(&array[minimum], &array[i]);
    }
}
```

- Function untuk mencetak hasil sorting

```
// function print untuk mencetak hasil dari pengurutan array
void printArray(int array[], int size) {
    int i;
    for (i=0; i < size; i++)
        cout << array[i] << " ";
    cout << endl;
}
```

- Fungsi utama program selection sort

```
// function utama program, untuk menjalankan semua function diatas
int main() {
    int n;                                // variabel n
    = besar atau jumlah angka yang ingin dimasukkan
    int myArray[n];
    cout << "masukkan jumlah angka : ";
}
```

```

cin >> n;
cout << "masukkan " << n << " angka acak : " << endl;
// iterasi untuk memasukkan angka sesuai dengan jumlahnya
for (int i = 0; i < n; i++) {
    cin >> myArray[i];
}
cout << endl;

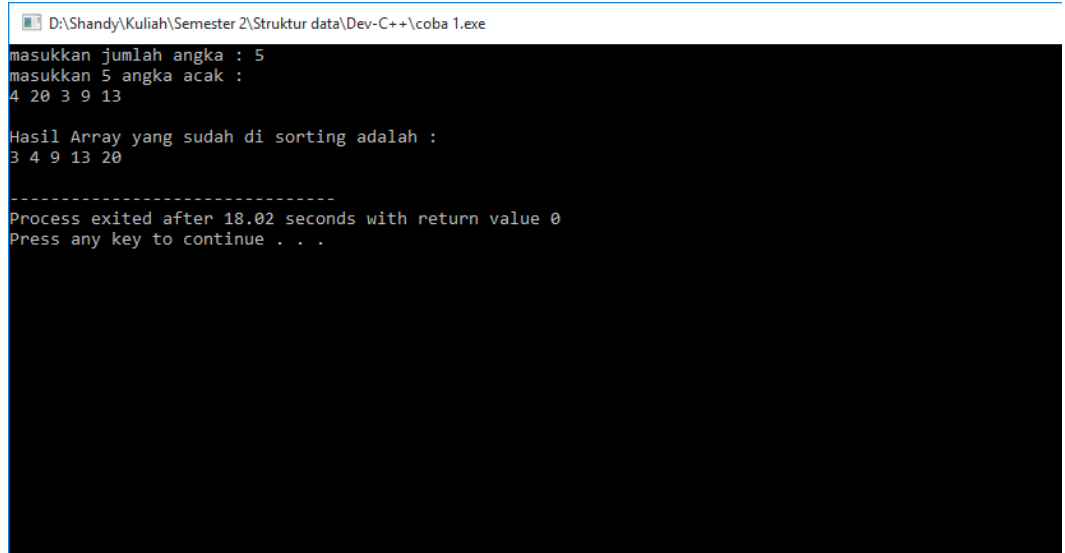
// proses mengurutkan array dimulai
selectionSort(myArray, n);

// menampilkan output dari function print
cout << "Hasil Array yang sudah di sorting adalah : " << endl;
printArray(myArray, n);

return 0;
}

```

- Hasil Program :



```

D:\Shandy\Kuliah\Semester 2\Struktur data\Dev-C++\coba 1.exe
masukkan jumlah angka : 5
masukkan 5 angka acak :
4 20 3 9 13

Hasil Array yang sudah di sorting adalah :
3 4 9 13 20

-----
Process exited after 18.02 seconds with return value 0
Press any key to continue . . .

```

○ Big O Selection Sort

- Konsep Selection Sort

Algoritma sorting sederhana yang lain adalah Selection Sort. Ide dasarnya adalah melakukan beberapa kali pass untuk melakukan penyeleksian elemen struktur data. Untuk sorting ascending (menaik), elemen yang paling kecil di antara elemen-elemen yang belum urut, disimpan indeksinya, kemudian dilakukan pertukaran nilai elemen dengan indeks yang disimpan tersebut dengan elemen yang paling depan yang belum urut. Sebaliknya, untuk, sorting descending (menurun), elemen yang paling besar yang disimpan indeksinya kemudian ditukar.

Cara kerja selection sort

Pass 1:

(70, 60, 30, 50, 40, 20) -> min 60
(70, 60, 30, 50, 40, 20) -> min 30
(70, 60, 30, 50, 40, 20) -> min 30
(70, 60, 30, 50, 40, 20) -> min 30
(70, 60, 30, 50, 40, 20) -> min 20
(20, 60, 30, 50, 40, 70) -> swap (70, 20)

Pass 2:

(20, 60, 30, 50, 40, 70) -> min 30
(20, 60, 30, 50, 40, 70) -> min 30
(20, 60, 30, 50, 40, 70) -> min 30
(20, 60, 30, 50, 40, 70) -> min 30
(20, 30, 60, 50, 40, 70) -> swap (60, 30)

Pass 3:

(20, 30, 60, 50, 40, 70) -> min 50
(20, 30, 60, 50, 40, 70) -> min 40
(20, 30, 60, 50, 40, 70) -> min 40
(20, 30, 40, 50, 60, 70) -> swap (60, 40)

Pass 4:

(20, 30, 40, 50, 60, 70) -> min 50
(20, 30, 40, 50, 60, 70) -> min 50
(20, 30, 40, 50, 60, 70)

Pass 5:

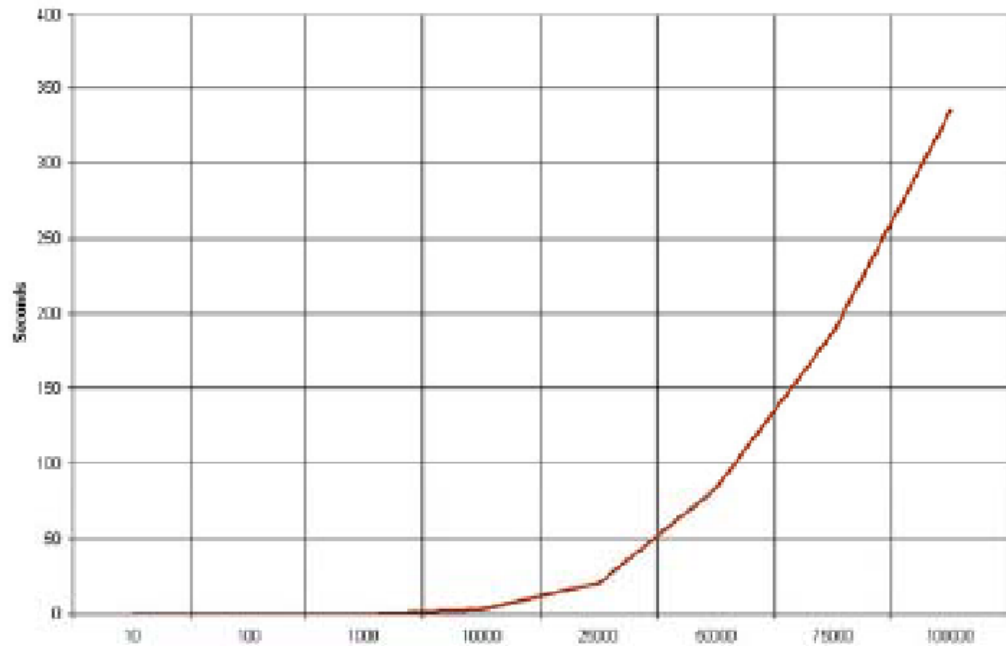
(20, 30, 40, 50, 60, 70) -> min 60
(20, 30, 40, 50, 60, 70)

- Kompleksitas Selection sort

Algoritma di dalam Selection Sort terdiri dari kalang Bersarang yang disebut pass berlangsung N-1 kali. Di dalam kalang kedua, dicari elemen dengan nilai terkecil. Jika didapat, indeks yang didapat ditimpakan ke variabel min. Lalu dilakukan proses penukaran. Begitu seterusnya untuk setiap Pass. Berdasarkan operasi perbandingan elemennya:

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

Perhitungan ini mirip dengan perhitungan kompleksitas bubble sort sebelumnya. Namun, melalui uji empiris, didapatkan bahwa Selection Sort lebih efisien daripada Bubble Sort. Grafiknya dapat dilihat berikut ini:



Peningkatan performansi yang diberikan 60% lebih baik daripada bubble sort. Namun, sorting model ini tergolong buruk dan lebih baik dihindari penggunaannya, terutama untuk penanganan tabel dengan lebih dari 1000 elemen. Karena masih ada algoritma lain yang implementasinya sama mudahnya, namun performansinya jauh lebih baik.

○ Kelebihan & Kekurangan Selection Sort

- Kelebihan
 - Algoritmanya mudah untuk diimplementasikan
 - Kompleksitas relatif lebih kecil
 - Waktu pengurutan singkat
 - Operasi pertukaran hanya dilakukan sekali
- Kekurangan
 - Sulit untuk membagi masalah