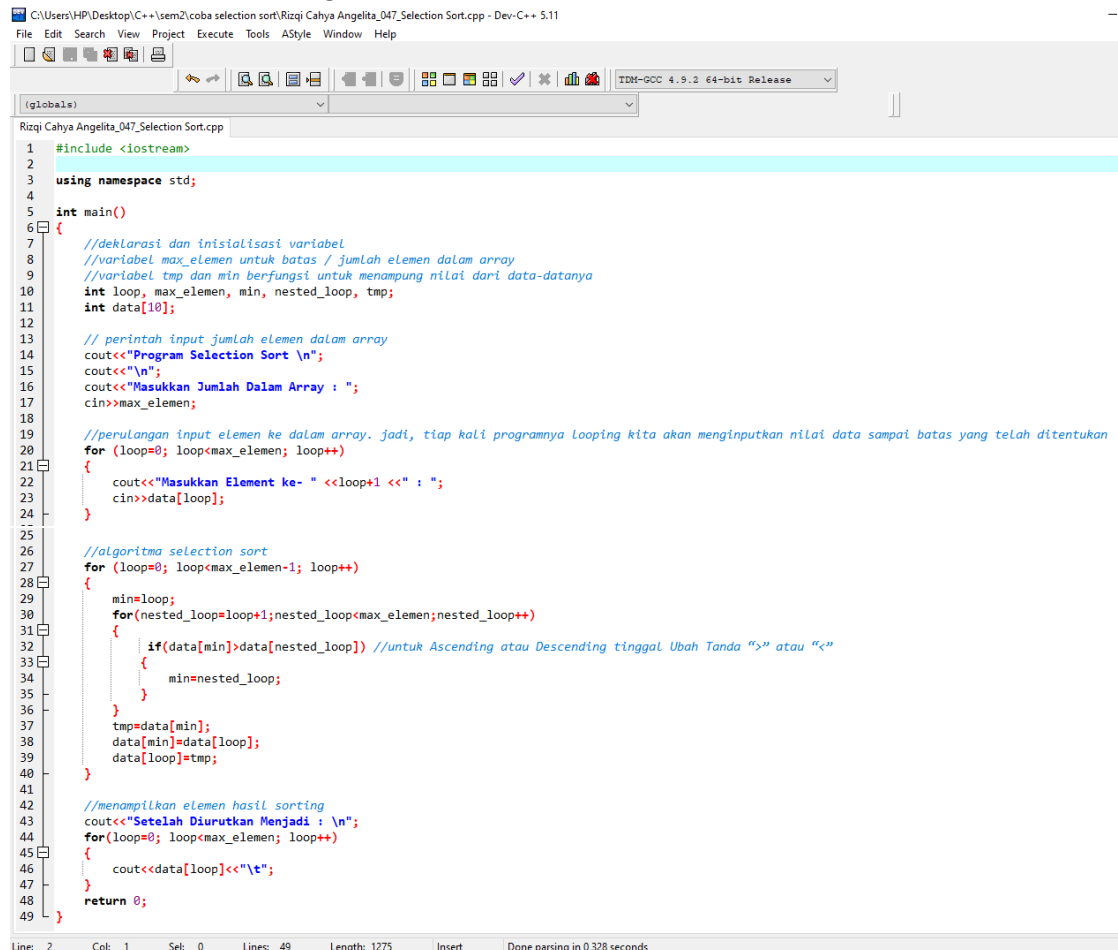


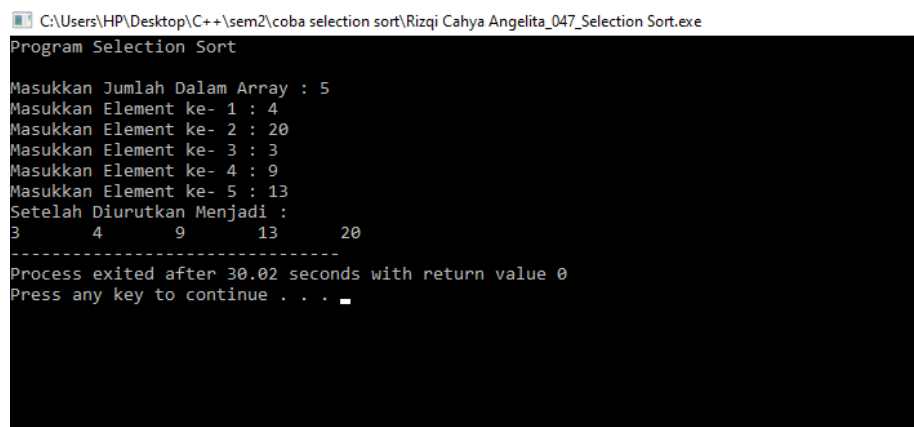
NAMA : RIZQI CAHYA ANGELITA  
NIM : 21091397047  
KELAS : 2021A  
PRODI : DIV MANAJEMEN INFORMATIKA

1. Laporan kodingan, screenshot kode tipe sorting yang anda buat, beri penjelasan, dan bukti berupa screenshot hasil run kodingan



```
1 #include <iostream>
2
3 using namespace std;
4
5 int main()
6 {
7     //deklarasi dan inisialisasi variabel
8     //variabel max_elemen untuk batas / jumlah elemen dalam array
9     //variabel tmp dan min berfungsi untuk menampung nilai dari data-datanya
10    int loop, max_elemen, min, nested_loop, tmp;
11    int data[10];
12
13    // perintah input jumlah elemen dalam array
14    cout<<"Program Selection Sort \n";
15    cout<<"\n";
16    cout<<"Masukkan Jumlah Dalam Array : ";
17    cin>>max_elemen;
18
19    //perulangan input elemen ke dalam array. jadi, tiap kali programnya looping kita akan menginputkan nilai data sampai batas yang telah ditentukan
20    for (loop=0; loop<max_elemen; loop++)
21    {
22        cout<<"Masukkan Element ke- " <<loop+1 <<" : ";
23        cin>>data[loop];
24    }
25
26    //algoritma selection sort
27    for (loop=0; loop<max_elemen-1; loop++)
28    {
29        min=loop;
30        for (nested_loop=loop+1; nested_loop<max_elemen; nested_loop++)
31        {
32            if (data[min]>data[nested_loop]) //untuk Ascending atau Descending tinggal Ubah Tanda ">" atau "<"
33            {
34                min=nested_loop;
35            }
36        }
37        tmp=data[min];
38        data[min]=data[loop];
39        data[loop]=tmp;
40    }
41
42    //menampilkan elemen hasil sorting
43    cout<<"Setelah Diurutkan Menjadi : \n";
44    for (loop=0; loop<max_elemen; loop++)
45    {
46        cout<<data[loop]<<"\t";
47    }
48    return 0;
49 }
```

Hasil run :



```
C:\Users\HP\Desktop\C++\sem2\coba selection sort\Rizqi Cahya Angelita_047_Selection Sort.exe
Program Selection Sort

Masukkan Jumlah Dalam Array : 5
Masukkan Element ke- 1 : 4
Masukkan Element ke- 2 : 20
Masukkan Element ke- 3 : 3
Masukkan Element ke- 4 : 9
Masukkan Element ke- 5 : 13
Setelah Diurutkan Menjadi :
3      4      9      13      20
-----
Process exited after 30.02 seconds with return value 0
Press any key to continue . . .
```

### Konsep Selection Sort:

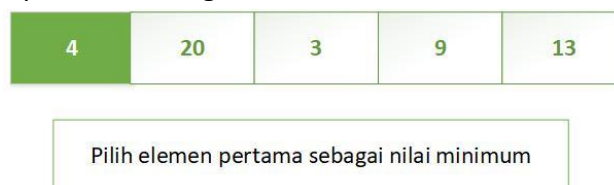
Algoritma Selection Sort pengurutannya dengan melakukan beberapa kali pass untuk melakukan penyeleksian elemen struktur data. Untuk sorting ascending (menaik), elemen yang paling kecil di antara elemen-elemen yang belum urut, disimpan indeksinya, kemudian dilakukan pertukaran nilai elemen dengan indeks 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. Selection Sort diakui karena kesederhanaan algoritmanya dan performanya lebih bagus daripada algoritma lain yang lebih rumit dalam situasi tertentu.

Algoritma ini bekerja sebagai berikut:

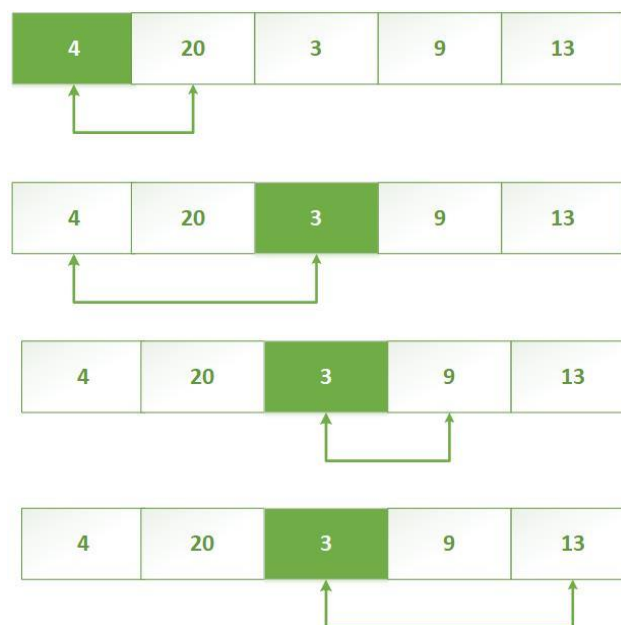
1. Mencari nilai minimum (jika ascending) atau maksimum (jika descending) dalam sebuah list
2. Menukarkan nilai ini dengan elemen pertama list
3. Mengulangi langkah di atas untuk sisa list dengan dimulai pada posisi kedua

### Penjelasan algoritma Selection Sort:

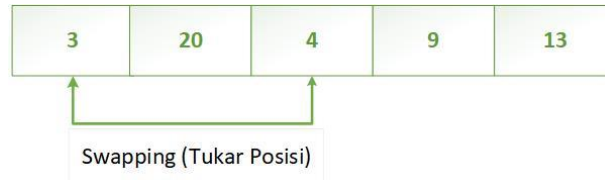
- 1.) Tetapkan elemen pertama sebagai nilai minimum



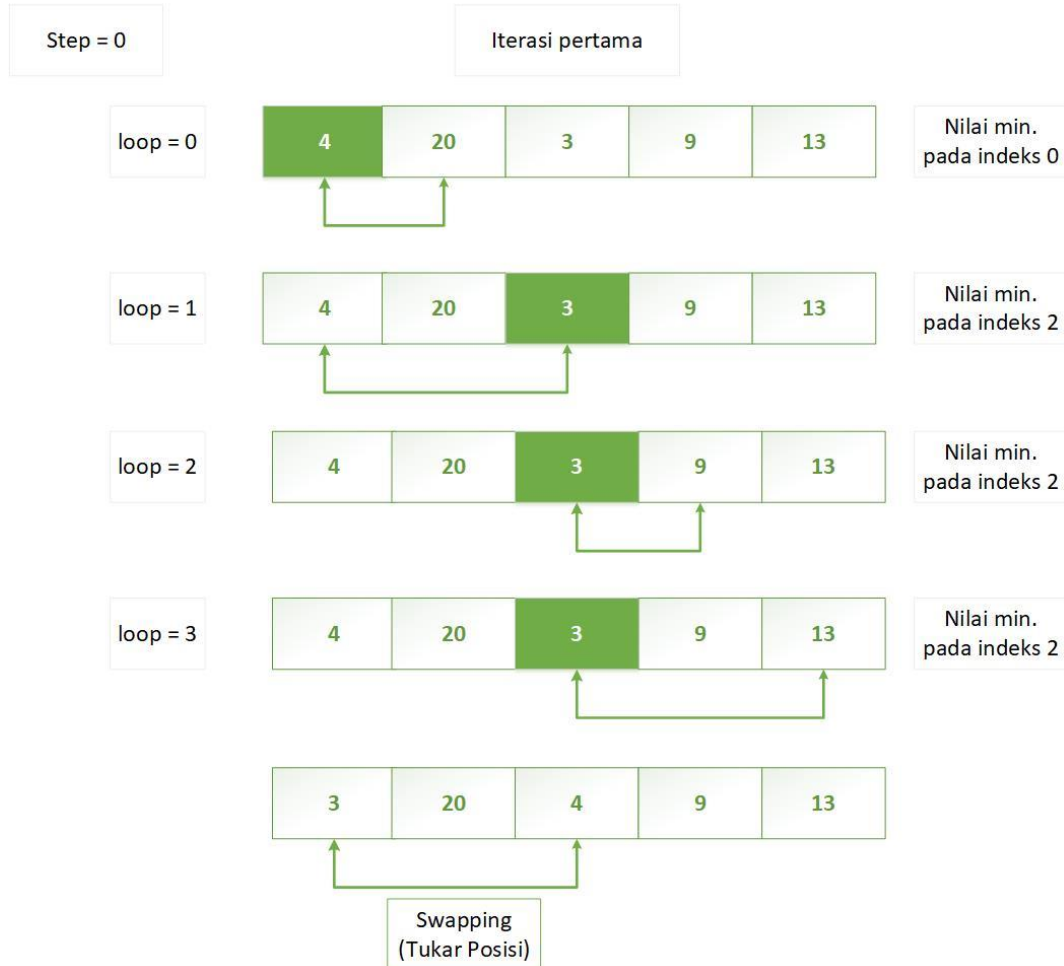
- 2.) Bandingkan nilai minimum dengan elemen kedua. Jika elemen kedua lebih kecil dari nilai minimum, tetapkan elemen kedua sebagai nilai minimum. Selanjutnya bandingkan nilai minimum dengan elemen ketiga. Jika elemen ketiga lebih kecil, maka tetapkan nilai minimum ke elemen ketiga jika tidak, maka jangan lakukan apapun. Proses berlangsung sampai elemen terakhir.



- 3.) Setelahnya setiap iterasi nilai minimum ditempatkan di depan daftar yang tidak disortir



4.) Setiap iterasi pengindeksan dimulai dari elemen pertama yang tidak disortir. Langkah 1 sampai 3 diulang sampai semua elemen ditempatkan pada posisi benar.



Step = 1

Iterasi kedua

loop = 0



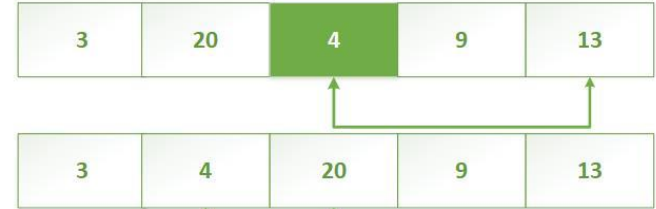
Nilai min.  
pada indeks 2

loop = 1



Nilai min.  
pada indeks 2

loop = 2



Nilai min.  
pada indeks 2

Swapping  
(Tukar Posisi)

Step = 2

Iterasi ketiga

loop = 0



Nilai min.  
pada indeks 3

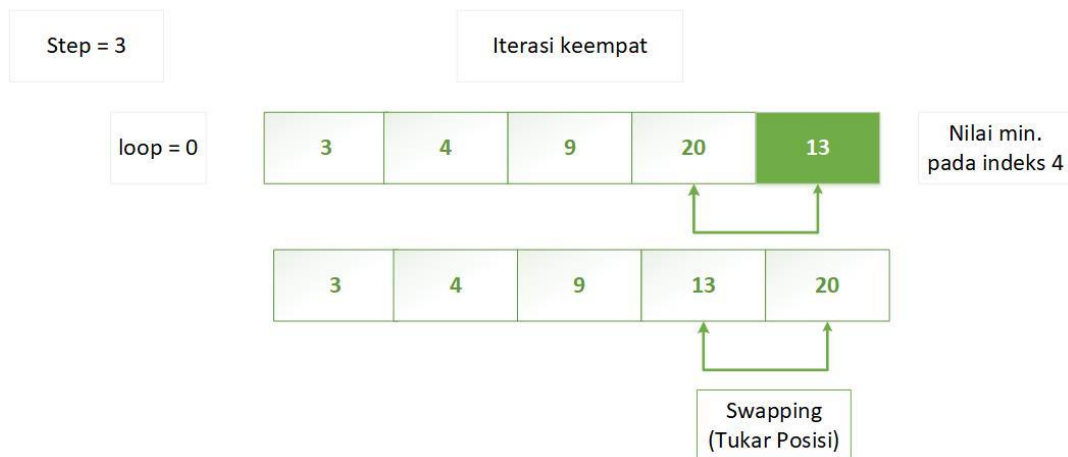
loop = 1



Nilai min.  
pada indeks 3



Swapping  
(Tukar Posisi)



2. Hitung jenis Big O nya, jelaskan kenapa kompleksitasnya adalah yang anda temukan.

Pada algoritma di dalam pengurutan selection sort ini terdiri dari kalang bersarang. Kalang tingkat pertama (disebut pass) berlangsung  $N-1$  kali. Di dalam kalang kedua, dicari elemen dengan nilai terkecil. Jika didapat, indeks yang didapat ditimpakan ke variabel min. kemudian dilakukan proses penukaran. Begitu seterusnya untuk setiap Pass. Pass sendiri ini akan makin berkurang hingga nilainya menjadi semakin kecil. Berdasarkan operasi perbandingan elemennya yaitu :

$$\begin{aligned} \text{Worst and Best case} &= (n-1) + (n-2) + \dots + 1 \\ &= \frac{n(n-1)}{2} = O(n^2) \end{aligned}$$

Banyaknya perbandingan yang harus dilakukan untuk siklus pertama adalah  $n$ , perbandingan yang harus dilakukan untuk siklus yang kedua  $n-1$ , dan seterusnya. Sehingga jumlah keseluruhan perbandingan adalah  $n(n+1)/2$  perbandingan.

```
int main()
{
    int loop, max_elemen, min, nested_loop, tmp; _____ 5
    int data[10]; _____ 1

    for (loop=0; loop<max_elemen-1; loop++) _____ 1+n+n-1 = 2n
    {
        min=loop; _____ n-1
        for (nested_loop=loop+1; nested_loop<max_elemen; nested_loop++) _____ (n-1)+(x-1)+x
        {
            if (data[min]>data[nested_loop]) _____ 3x
            {
                min=nested_loop; _____ x
            }
        }
        tmp=data[min]; _____ 2(n-1)
        data[min]=data[loop]; _____ 2(n-1)
        data[loop]=tmp; _____ 2(n-1)
    }
    return 0; _____ 1
}
```

Worst case terjadi jika array yang diterima dalam urutan menurun. Dalam hal itu kedua loop akan mengeksekusi jumlah maksimum kali dan blok if akan dimasukkan pada setiap instance. Sedangkan untuk Best Case terjadi ketika array sudah diurutkan.

n=5      loop

loop=0 → loop < 4

0,1,2,3

Eksekusi 4 kali

n-1 kali

n = 5          nested\_loop

loop	loop+1 → nested_loop < 5	eksekusi
0	1, 2, 3, 4	4 kali
1	2, 3, 4	3 kali
2	3, 4	2 kali
3	4	1 kali

Bahwa , nested\_loop dijalankan 1 kali + 2 kali + 3 kali .... + (n-1) , untuk mempermudah hitungan maka dimisalkan = x kali

$$\begin{aligned}T(n) &= 5 + 1 + 2n + n-1 + n-1 + x-1 + x + 3x + x + 2(n-1) + 2(n-1) + 2(n-1) + 1 \\&= 6x + 10n - 2\end{aligned}$$

$$x = 1 + 2 + 3 \dots + (n-1)$$

Dalam Arithmetic progression (AP), jika memiliki deret  $1 + 2 + 3 \dots + n = \frac{n(n+1)}{2}$

$$\text{bahwa } x = \frac{(n-1)(n-1+1)}{2} = \frac{n(n-1)}{2}$$

selanjutnya substitusikan

$$\begin{aligned}T(n) &= 6x + 10n - 2 \\&= 3n^2 - 3n + 10n - 2 \\&= 3n^2 + 7n - 2\end{aligned}$$

Berdasarkan implementasi, maka menemukan notasi Big O dengan mengambil istilah yang akan mempengaruhi kompleksitas waktu yang paling berkaitan dengan n yaitu  $3n^2$  karena akan mempengaruhi kompleksitas waktu terhadap n secara kuadrat. Sehingga (dengan mengabaikan konstanta) di dapat  $T(n) = O(n^2)$  atau selection sort berjalan di  $O(n^2)$ .

Untuk kompleksitas ruang  $O(1)$  karena variabel tambahan tmp digunakan.

3. Jelaskan kelebihan dan kekurangan sorting yang kalian buat dibandingkan yang dibuat oleh teman kalian.

Kelebihan :

- 1.) Algoritma ini sangat rapat dan mudah untuk diimplementasikan.
- 2.) Operasi pertukarannya hanya dilakukan sekali.
- 3.) Waktu pengurutan dapat lebih ditekan.
- 4.) Mudah menggabungkannya kembali.
- 5.) Kompleksitas selection sort relatif lebih kecil.

Kekurangan :

- 1.) Penanganan dengan menggunakan model algoritma ini perlu di hindari jika pengurutan nilai dengan data pada tabel lebih besar dari 1000 buah atau pengurutan tabel lebih dari beberapa ratus kali. Karena hal ini akan menyebabkan kompleksitas yang lebih tinggi dan kurang praktis
- 2.) Membutuhkan metode tambahan
- 3.) Sulit untuk mengatasi masalah