Praktikum 10

Algoritma Pengurutan (Insertion Sort dan Selection Sort)

A. TUJUAN PEMBELAJARAN

Setelah melakukan praktikum dalam bab ini, mahasiswa diharapkan mampu:

1. Memahami mengenai algoritma pengurutan *insertion sort* dan *selection sort.*
2. Mampu mengimplementasikan algoritma pengurutan *insertion sort* dan *selection sort* secara *ascending* dan *descending.*

B. DASAR TEORI

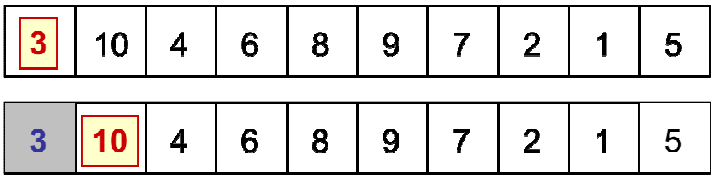
B.1 Algoritma *Insertion Sort*

Salah satu algoritma sorting yang paling sederhana adalah *insertion sort*. Algoritma *insertion sort* pada dasarnya memilah data yang akan urutkan menjadi 2 bagian, yang belum diurutkan dan yang sudah diurutkan. Elemen pertama diambil dari bagian array yang belum diurutkan dan kemudian diletakkan sesuai posisinya pada bagian lain dari array yang telah diurutkan. Langkah ini dilakukan secara berulang hingga tak ada lagi elemen tersisa pada bagian array yang belum diurutka.

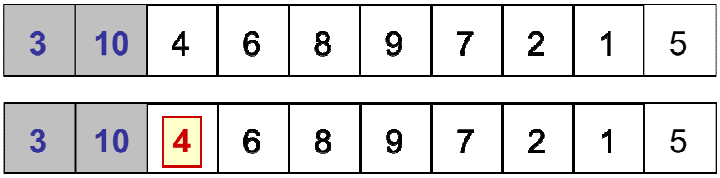
Metode insection sort merupakan metode yang mengurutkan bilangan-bilangan yang telah terbaca, dan berikutnya secara berulang akan menyisipkan bilangan-bilangan dalam array yang belum terbaca ke sisi kiri array yang telah terurut. Kita mengambil pada bilangan yang paling kiri. Bilangan tersebut dikatakan urut terhadap dirinya sendiri karena bilangan yang di bandingkan baru 1.

3	10	4	6	8	9	7	2	1	5

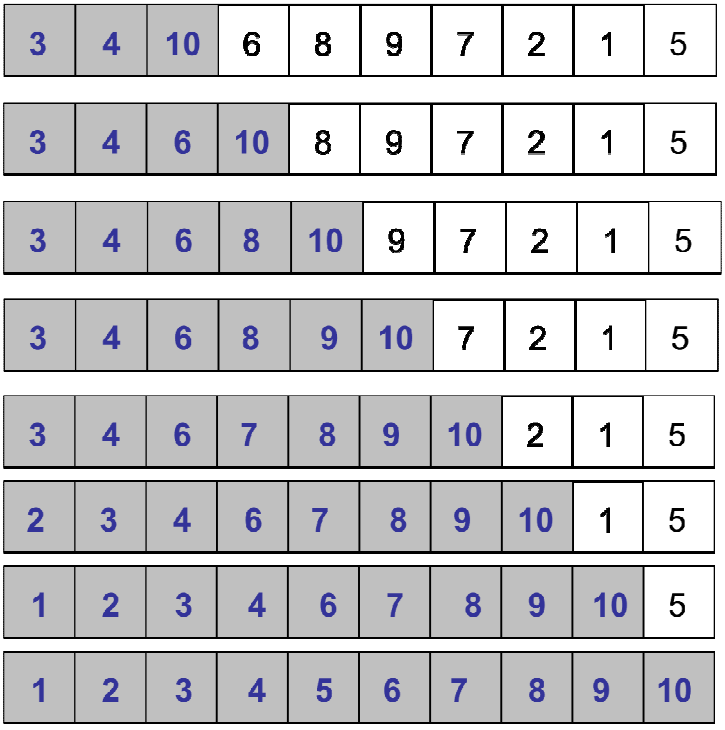
Cek bilangan ke 2 (10) apakah lebih kecil dari bilangan yang ke 1(3).Apabila lebih kecil maka ditukar. Tapi kali ini bilangan ke 1 lebih kecil dari bilangan ke 2 maka tidak ditukar.



Pada kotak warna abu2 sudah dalam keadaan terurut. Kemudian membandingkan lagi pada bilangan selanjutnya yaitu bilangan ke 3 (4). Bandingkan dengan bilangan yang ada di sebelah kirinya. Pada kasus ini bilangan ke 2 bergeser dan digantikan bilangan ke 3.



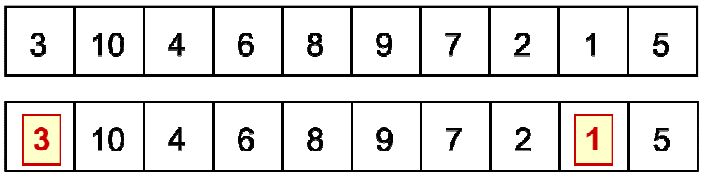
Lakukan langkah seperti di atas pada bilangan selanjutnya. 4 bilangan pertama sudah dalam keadaan terurut relatif. Ulangi proses tersebut sampai bilangan terakhir disisipkan.



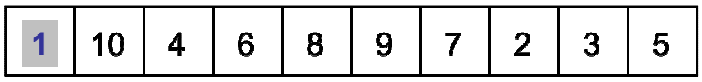
B.2 Algoritma *Selection Sort*

Ide utama dari algoritma *selection sort* adalah memilih elemen dengan nilai paling rendah dan menukar elemen yang terpilih dengan elemen ke-*i*. Nilai dari *i* dimulai dari 1 ke *n*, dimana *n* adalah jumlah total elemen dikurangi 1.

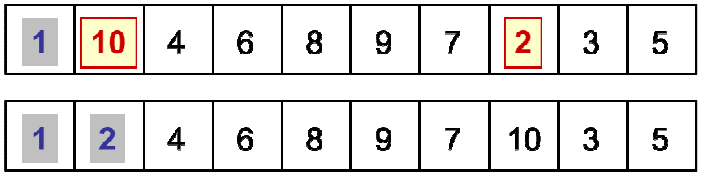
Cek seluruh array dan cari array yang mempunyai nilai terkecil → index 8 (1). Setelah ketemu tukar dengan array yang berada di pojok kiri (3).



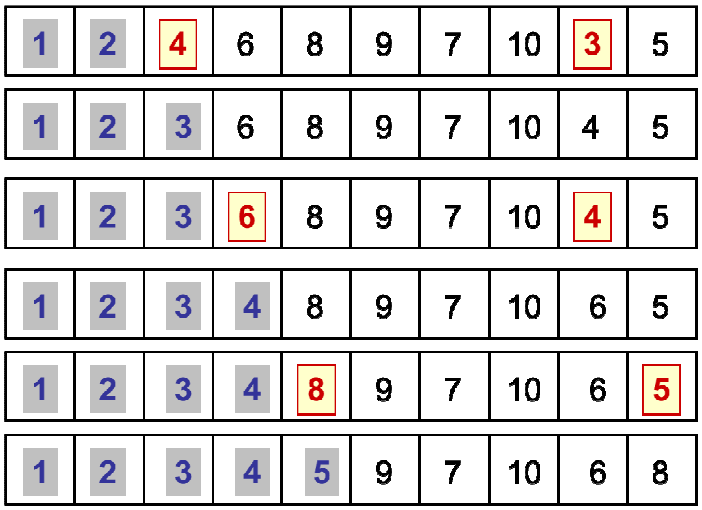
Setelah di tukar bagian yang berwarna abu-abu merupakan index yang telah terurutkan.

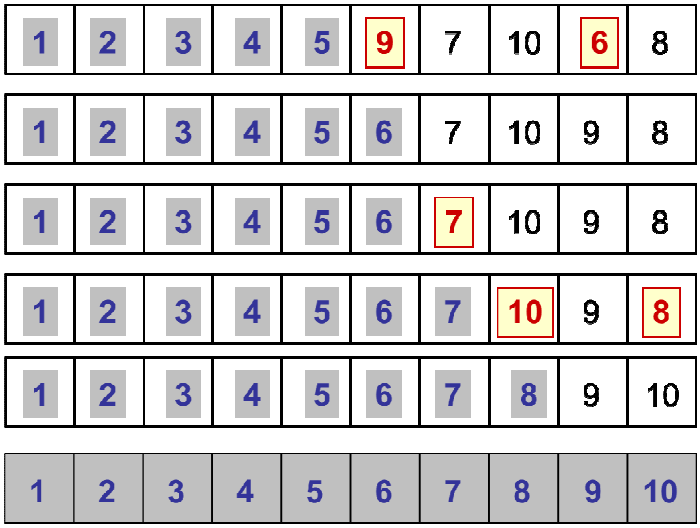


Kemudian cari bilangan terkecil selanjutnya (selain di kotak abu-abu) yaitu bilangan 2 dan tukar dengan sebelah array yang telah terurutkan.



Dua array sudah terurutkan. Kemudian ulangi langkah di atas dan lakukan langkah yang sama yaitu pilih terkecil dan tukar dengan sebelah array yang sudah terurutkan.





C. TUGAS PENDAHULUAN

Jawablah pertanyaan berikut ini :

1. Tuliskan algoritma pengurutan *insertion sort* secara *ascending*.
2. Tuliskan algoritma pengurutan *selection sort* secara *ascending*.

D. PERCOBAAN

Percobaan 1 : Mengacak data

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>

#define MAX 20

void main()
{
	int data_awal[MAX], data_urut[MAX];
	int i;

	printf("Sebelum pengurutan : \n");
	for(i=0; i<MAX; i++){
		srand(time(NULL) * (i+1));
		data_awal[i] = rand() % 100 + 1;
		printf("%d ", data_awal[i]);
	}
	printf("\n");
	for(i=0; i<MAX; i++)
		data_urut[i] = data_awal[i];
}
```

Percobaan 2 : *Insertion sort* secara *ascending*

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
```

```
#define MAX 20

void InsertionSort(int arr[])
{
      int i, j, key;
    for (i=1; i<MAX; i++) {
            key = arr[i];
            j = i-1;
            while(j>=0 && arr[j]>key){
                  arr[j+1] = arr[j];
                  j--;
            }
            arr[j+1] = key;
      }
}

void main()
{
      int data_awal[MAX], data_urut[MAX];
      int i;
      long k1, k2;

      printf("Sebelum pengurutan : \n");
      for(i=0; i<MAX; i++){
            srand(time(NULL) * (i+1));
            data_awal[i] = rand() % 100 + 1;
            printf("%d ", data_awal[i]);
      }
      printf("\nSetelah pengurutan : \n");
      for(i=0; i<MAX; i++)
            data_urut[i] = data_awal[i];

      time(&k1);
      InsertionSort(data_urut);
      time(&k2);
      for(i=0; i<MAX; i++)
            printf("%d ", data_urut[i]);
      printf("\nWaktu = %ld\n", k2-k1);
}
```

Percobaan 3 : *Selection sort* secara *ascending*

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>

#define MAX 20


void SelectionSort(int arr[])
{
      int i, j, temp, min, min_id;
      i = 0;
      while(i < MAX-1){
            min_id = i;
            min = arr[i];
            for(j=i+1; j<MAX; j++)
                  if(arr[j] < min){
```

```
                        min = arr[j];
                        min_id = j;
                    }
            temp = arr[min_id];
            arr[min_id] = arr[i];
            arr[i] = temp;
            i++;
      }
}

void main()
{
      int data_awal[MAX], data_urut[MAX];
      int i;
      long k1, k2;

      printf("Sebelum pengurutan : \n");
      for(i=0; i<MAX; i++){
            srand(time(NULL) * (i+1));
            data_awal[i] = rand() % 100 + 1;
            printf("%d ", data_awal[i]);
      }
      printf("\nSetelah pengurutan : \n");
      for(i=0; i<MAX; i++)
            data_urut[i] = data_awal[i];

      time(&k1);
      SelectionSort(data_urut);
      time(&k2);
      for(i=0; i<MAX; i++)
            printf("%d ", data_urut[i]);
      printf("\nWaktu = %ld\n", k2-k1);
}
```

E. LATIHAN

1. Dari percobaan 1 tambahkan fungsi untuk melakukan pengurutan *insertion sort* secara *descending*.
2. Dari percobaan 2 tambahkan fungsi untuk melakukan pengurutan *selection sort* secara *descending*.
3. Buatlah program untuk melakukan pengurutan *insertion sort* terurut dari kanan ke kiri secara *ascending* dan *descending*.
4. Buatlah program untuk melakukan pengurutan *selection sort* terurut dari kanan ke kiri secara *ascending* dan *descending*.
5. Buatlah struktur `Mahasiswa` dengan variable `nrp` dan `nama` yang memiliki tipe `String` dan `kelas` yang bertipe `int`. Buatlah fungsi pengurutan dengan *insertion sort* dan *selection sort* berdasarkan `nrp`.

```
struct Mahasiswa {
    char nrp[10];
    char nama[20];
    int kelas;
};
```

F. LAPORAN RESMI

1. Kerjakan hasil percobaan(D) dan latihan(E) di atas dan tambahkan analisa.
2. Tuliskan kesimpulan dari percobaan dan latihan yang telah anda lakukan.