

**12 LAPORAN AKHIR PRAKTIKUM**  
**MATA KULIAH**  
**MODUL 7**  
**JUDUL MODUL : SORTING DAN SEARCHING**



**Nama** : Kurnia Fajar Rahyudi Putra  
**No. BP** : 2211512013  
**Hari/Tanggal** : Senin/ 12 Juni 2023  
**Shift** : 1

**Dosen** : Dodon Yendri, M.Kom

**LABORATORIUM KOMPUTER DAN JARINGAN**  
**DEPARTEMEN TEKNIK KOMPUTER**  
**FAKULTAS TEKNOLOGI INFORMASI**  
**UNIVERSITAS ANDALAS**  
**PADANG**  
**2023**

# **BAB I**

## **PENDAHULUAN**

### 1.1 Tujuan

- 1.1.1 Memperkenalkan kepada mahasiswa metoda pengurutan (sorting)
- 1.1.2 Memperkenalkan kepada mahasiswa metoda pencarian (searching)
- 1.1.3 Mempraktekkan pemakaian metoda bubblesort dan selection sort
- 1.1.4 Mempraktekkan pemakaian metoda sequential search dan binary search

### 1.2 Landasan Teori

#### 1.2.1 Sorting (Pengurutan)

Sorting adalah suatu proses pengurutan data yang sebelumnya disusun secara acak atau tidak teratur menjadi urut dan teratur menurut suatu aturan tertentu. Sorting dapat dibedakan menjadi dua jenis yaitu ascending dan descending. Ascending adalah pengurutan data dari kecil ke besar, sedangkan descending adalah pengurutan data dari besar ke kecil. Nah, ada banyak program sorting seperti bubble sort, selection sort, insertion sort, merge sort, quick sort, dan lain sebagainya.

Contoh:

Data Acak : 5 6 8 1 3 25 10

Ascending : 1 3 5 6 8 10 25

Descending : 25 10 8 6 5 3 1

Dalam aktivitas sehari-hari, kita sering di hadapkan pada perlunya melakukan suatu sortir (pengurutan). Anak-anak sekolah dasar berbaris menurut tinggi badannya, petugas perpustakaan menyusun buku-buku sesuai dengan nomor bukunya, lembar absensi dicetak berdasarkan urutan abjad nama, dan sebagainya.

Pengurutan biasanya dilakukan untuk tujuan mempermudah pencarian, misalnya mencari nomor telepon di buku telepon, mengetahui ranking siswa di sekolah, dan sebagainya.

Pada algoritma sorting umumnya menggunakan fungsi tukar 2 buah data:

```

void tukar( int a, int b) {
    int tmp;
    tmp = data[a];
    data[a] = data[b];
    data[b] = tmp;
}

```

### 1.2.2 Buuble Sort

Bubble Sort merupakan metode sorting termudah. Diberi nama “Bubble” karena proses pengurutan secara berangsur-angsur bergerak/berpindah ke posisinya yang tepat, seperti gelembung yang keluar dari sebuah gelas bersoda. Bubble Sort **mengurutkan data dengan cara membandingkan elemen sekarang dengan elemen berikutnya**. Jika elemen sekarang lebih besar dari elemen berikutnya maka kedua elemen tersebut ditukar, jika pengurutan ascending. Jika elemen sekarang lebih kecil dari elemen berikutnya, maka kedua elemen tersebut ditukar, jika pengurutan descending.

Algoritma ini seolah-olah menggeser satu per satu elemen dari kanan ke kiri atau kiri ke kanan, tergantung jenis pengurutannya. **Ketika satu proses telah selesai, maka bubble sort akan mengulangi proses, demikian seterusnya.** Ketika satu proses telah selesai, maka bubble sort akan mengulangi proses, demikian seterusnya. Kapan berhentinya? Bubble sort berhenti jika seluruh array telah diperiksa dan tidak ada pertukaran lagi yang bisa dilakukan, serta tercapai perurutan yang telah diinginkan. Algoritma Bubble Sorting mudah dalam sintaks, tetapi lebih lambat dibandingkan dengan algoritma sorting yang lain.

Algoritma dari teknik ini adalah dengan melakukan proses perbandingan sebanyak  $n$  elemen dimulai dari  $n=1$  (selanjutnya disebut mulai=1). Bandingkan seluruh elemen diawali dari elemen sebelah kanan hingga ke- $n$ . bila elemen tersebut  $<$  dari mulai, maka lakukan pertukaran tempat. Lakukan pada elemen mulai +1 seperti proses sebelumnya hingga nilai mulai +1 =  $n$ .

Contoh data awal : 8, 7, 6, 5, 4

*Pass 1 :*

Proses 1, bandingkan 8 (elemen pertama) dengan 7 (elemen kedua), hasilnya 7, 8, 6, 5, 4.

Proses 2, bandingkan 7 (elemen pertama) dengan 6 (elemen kedua), hasilnya 6, 8, 7, 5, 4

Proses 3, bandingkan 6 (elemen pertama) dengan 5 (elemen kedua), hasilnya 5, 8, 7, 6, 4

Proses 4, bandingkan 5 (elemen pertama) dengan 4 (elemen kedua), hasilnya 4, 8, 7, 6, 5

*Pass 2* : Dimulai dengan elemen kedua, dibandingkan dengan elemen ketiga hingga elemen kelima, berikut hasil-hasilnya :

Proses 1, menghasilkan : 4, 7, 8, 6, 5

Proses 2, menghasilkan : 4, 6, 8, 7, 5

Proses 3, menghasilkan : 4, 7, 8, 7, 6 (urutan *pass 2* telah diperoleh).

*Pass 3* : Dimulai dengan elemen ketiga, dibandingkan dengan elemen keempat hingga elemen kelima, berikut hasil-hasilnya :

Proses 1, menghasilkan : 4, 5, 7, 8, 6

Proses 2, menghasilkan : 4, 5, 6, 8, 7 (urutan *pass 3* telah diperoleh).

*Pass 4* : Dimulai dengan elemen keempat, dibandingkan dengan elemen kelima, dan menghasilkan :

Proses 1, menghasilkan : 4, 5, 6, 7, 8, (urutan *pass 3* telah diperoleh).

#### Kelebihan Bubble Sort

- Metode Buble Sort merupakan metode yang paling simple
- Metode Buble Sort mudah dipahami algoritmanya.

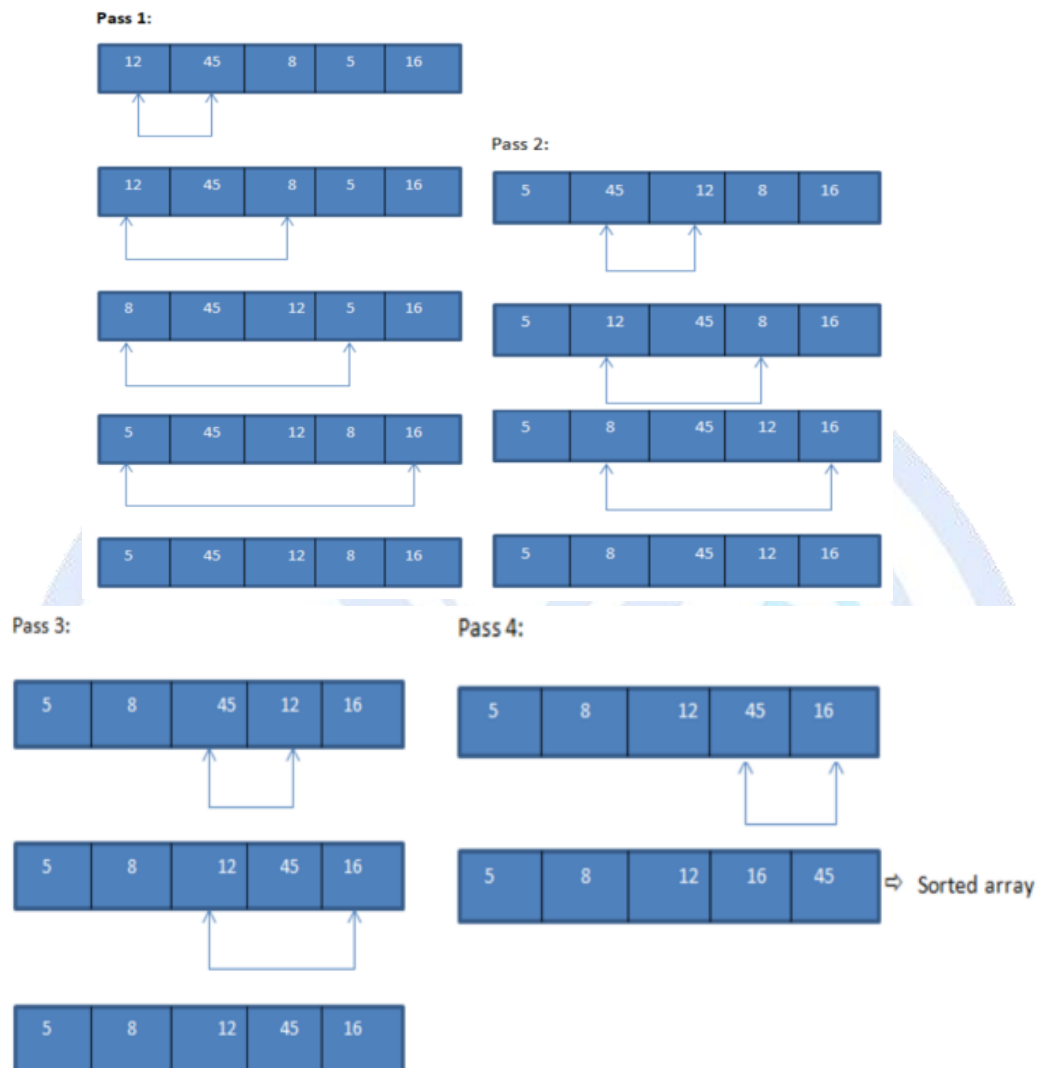
#### Kelemahan Bubble Sort

Meskipun simpel metode Bubble sort merupakan metode pengurutan yang paling tidak efisien. *Kelemahan bubble sort* adalah pada saat mengurutkan data yang sangat besar akan mengalami kelambatan luar biasa, atau dengan kata lain kinerja memburuk cukup signifikan ketika data yang diolah jika data cukup banyak. Kelemahan lain adalah jumlah pengulangan akan tetap sama jumlahnya walaupun data sesungguhnya sudah cukup terurut. Hal ini disebabkan setiap data dibandingkan dengan setiap data yang lain untuk menentukan posisinya.

#### 1.2.3 Selection Sort

Merupakan kombinasi antara sorting dan searching. Untuk setiap proses, akan dicari elemen-elemen yang belum diurutkan yang memiliki nilai terkecil atau terbesar akan dipertukarkan ke posisi yang tepat di dalam array. Misalnya untuk putaran pertama, akan dicari data dengan nilai terkecil dan data ini akan ditempatkan di indeks terkecil (data[0]), pada putaran kedua akan dicari data kedua terkecil, dan akan ditempatkan di indeks kedua (data[1]). Selama proses, perbandingan dan pengubahan hanya dilakukan

pada indeks pembandingan saja, pertukaran data secara fisik terjadi pada akhir proses.



#### 1.2.4 Searching

Searching merupakan proses fundamental dalam pengelolaan data. Proses pencarian adalah menemukan nilai (data) tertentu di dalam sekumpulan data yang bertipe sama (baik bertipe dasar atau bertipe bentukan).

**Sebagai contoh,** dikehendaki untuk mendapatkan mahasiswa dengan nomor urut 9834567. Hasilnya adalah rekaman yang berisi data mahasiswa tersebut; yang barangkali berisi nama, alamat, tanggal lahir, dan nama program studi. Dalam implementasi, algoritma bisa jadi memberikan nilai balik berupa sebuah rekaman yang diperoleh, tetapi bisa pula hanya memberikan pointer yang merujuk ke sebuah rekaman.

Pencarian dapat dilakukan terhadap data yang secara keseluruhan berada dalam memori komputer ataupun terhadap data yang berada dalam pencarian eksternal (harddisk). Pencarian yang dilakukan terhadap data yang berada dalam memori komputer dikenal dengan sebutan pencarian internal,

sedangkan pencarian yang dilakukan pada media penyimpanan eksternal disebut pencarian eksternal.

Metode Searching dapat dibedakan menjadi 2, yaitu :

1. Sequential Search
2. Binary Search

#### 1.2.5 Sequential Search

Konsep yang digunakan dalam metode ini adalah membandingkan data-data yang ada dalam kumpulan tersebut, mulai dari elemen pertama sampai elemen ditemukan, atau sampai elemen terakhir.

12	13	19	27	28
----	----	----	----	----

Misalkan, dari data diatas angka yang akan dicari adalah angka 19 dalam array A, maka proses yang akan terjadi pada proses pencarian adalah sebagai berikut:

- Pencarian dimulai pada index ke-0 yaitu angka 12, kemudian dicocokkan dengan angka yang akan dicari, jika tidak sama maka pencarian akan dilanjutkan ke index selanjutnya.
- Pada index ke-1, yaitu angka 13, juga bukan angka yang dicari, maka pencarian juga akan dilanjutkan pada index selanjutnya
- Pada index ke-2, yaitu angka 19, ternyata angka 19 merupakan angka yang dicari. Pencarian angka telah ditemukan, maka pencarian akan dihentikan dan keluar dari looping pencarian.

#### 1.2.6 Binary Search

Binary search adalah sebuah algoritma pencarian dengan cara membagi data menjadi dua bagian setiap kali terjadi proses pencarian untuk menemukan nilai tertentu dalam sebuah larik (array) linear. Sebuah pencarian biner mencari nilai tengah (median), melakukan sebuah perbandingan untuk menentukan apakah nilai yang dicari ada sebelum atau sesudahnya, kemudian mencari setengah sisanya dengan cara yang sama.

**Contoh Data:**

Misalnya data yang dicari 17

3	9	11	12	15	17	20	23	31	35
awal				tengah					akhir

Karena  $17 > 15$  (data tengah), maka: **awal = tengah + 1**

3	9	11	12	15	17	20	23	31	35
				awal	tengah				akhir

Karena  $17 < 23$  (data tengah), maka: **akhir = tengah - 1**

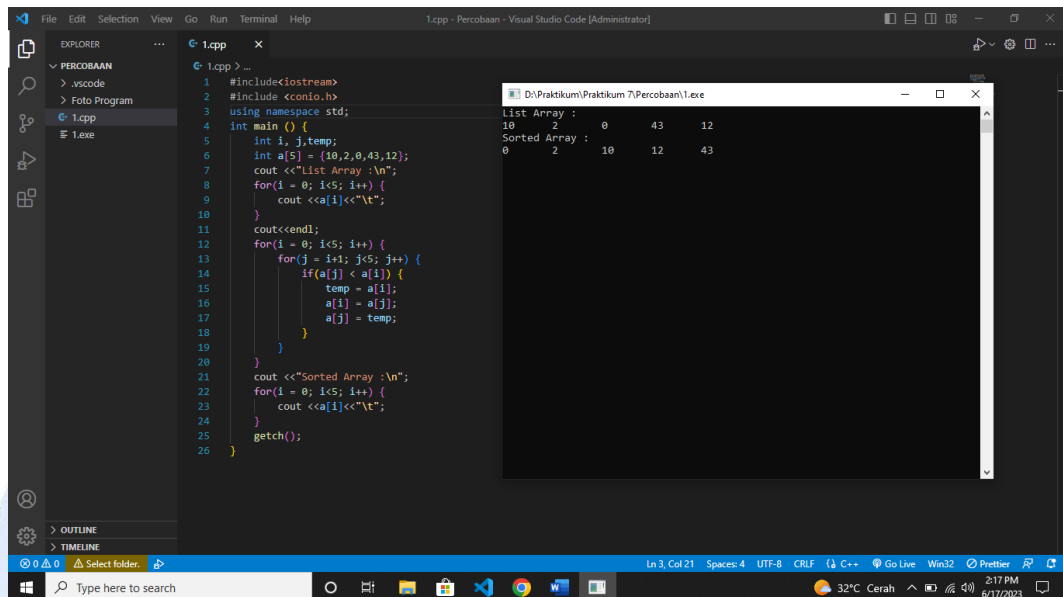
3	9	11	12	15	17	20	23	31	35
				awal=tengah	akhir				

Karena  $17 = 17$  (data tengah), maka **KETEMU!**



## BAB II PERCOBAAN

### 2.1 Percobaan 7.1



The screenshot shows the Visual Studio Code editor with a C++ file named `1.cpp`. The code implements a sorting algorithm on an array of 5 integers. The Explorer sidebar on the left shows the project structure with folders `.vscode` and `Foto Program`, and files `1.cpp` and `1.exe`. The code in `1.cpp` is as follows:

```
1 #include<iostream>
2 #include <conio.h>
3 using namespace std;
4 int main () {
5     int i, j, temp;
6     int a[5] = {10, 2, 0, 43, 12};
7     cout << "List Array : \n";
8     for(i = 0; i < 5; i++) {
9         cout << a[i] << " ";
10    }
11    cout << endl;
12    for(i = 0; i < 5; i++) {
13        for(j = i+1; j < 5; j++) {
14            if(a[j] < a[i]) {
15                temp = a[i];
16                a[i] = a[j];
17                a[j] = temp;
18            }
19        }
20    }
21    cout << "Sorted Array : \n";
22    for(i = 0; i < 5; i++) {
23        cout << a[i] << " ";
24    }
25    getch();
26 }
```

The output window on the right shows the execution results:

```
List Array :
10 2 0 43 12
Sorted Array :
0 2 10 12 43
```

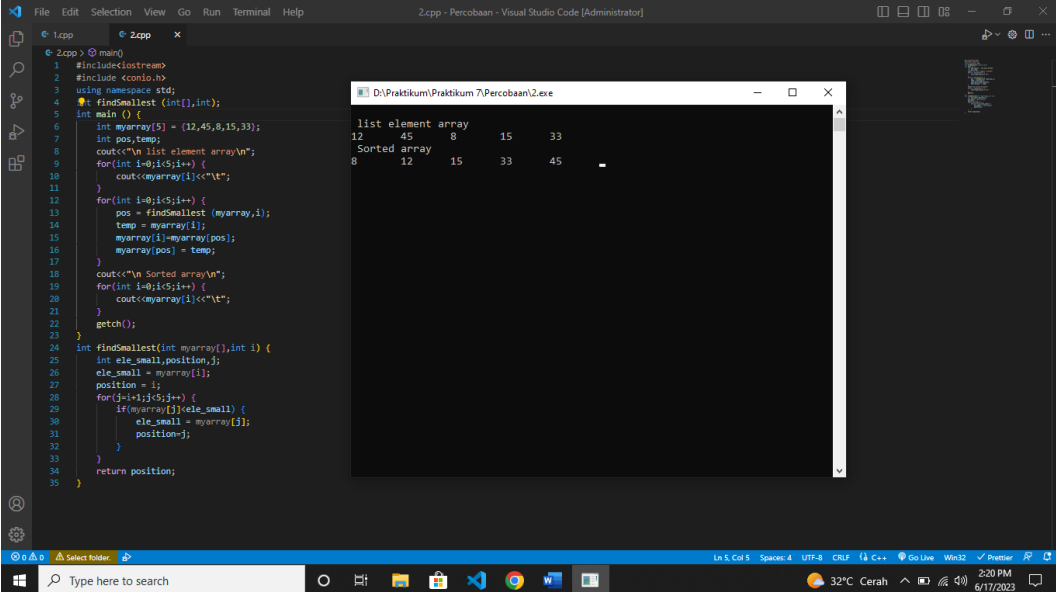
Program dimulai dengan mendefinisikan beberapa variabel yang diperlukan, seperti `i` dan `j` untuk perulangan, serta `temp` sebagai variabel penampung sementara saat pertukaran nilai dalam array. Kemudian, terdapat array `a` yang berisi 5 bilangan integer tak terurut.

Program mencetak daftar array awal dengan menggunakan perulangan `for` pertama. Nilai-nilai array dicetak satu per satu dengan menggunakan `cout`. Setelah itu, program melakukan pengurutan array dengan menggunakan nested loop `for`. Loop pertama (variabel `i`) digunakan untuk mengiterasi elemen-elemen array dari indeks 0 hingga 4.

Loop kedua (variabel `j`) digunakan untuk membandingkan elemen array dengan elemen-elemen setelahnya. Jika nilai pada indeks `j` lebih kecil dari nilai pada indeks `i`, maka dilakukan pertukaran nilai tersebut.

Pertukaran nilai dilakukan dengan menggunakan variabel `temp` sebagai variabel penampung. Pertukaran dilakukan dengan cara memindahkan nilai pada indeks `j` ke `temp`, lalu menggantinya dengan nilai pada indeks `i`, dan terakhir memasukkan nilai dari `temp` ke indeks `j`. Setelah selesai melakukan pengurutan, program mencetak daftar array yang telah diurutkan menggunakan perulangan `for` ketiga.

## 2.2 Percobaan 7.2



The screenshot shows a Visual Studio Code editor with a C++ file named `2.cpp`. The code implements a selection sort algorithm. It starts with a `main` function that initializes an array `myarray` with the values `{12, 45, 8, 15, 33}`. It then prints the initial array. A `for` loop iterates over the array, and for each element, it calls the `findSmallest` function to find the minimum element in the unsorted portion of the array. The elements are then swapped. Finally, the sorted array is printed. The `findSmallest` function is a helper function that takes the array and a starting index `i`, and returns the index of the smallest element from `i` onwards.

```
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4 int findSmallest (int[],int);
5 int main () {
6     int myarray[5] = {12,45,8,15,33};
7     int pos,temp;
8     cout<<"list element array\n";
9     for(int i=0;i<5;i++) {
10         cout<<myarray[i]<<"\t";
11     }
12     for(int i=0;i<5;i++) {
13         pos = findSmallest (myarray,i);
14         temp = myarray[i];
15         myarray[i]=myarray[pos];
16         myarray[pos] = temp;
17     }
18     cout<<"\n Sorted array\n";
19     for(int i=0;i<5;i++) {
20         cout<<myarray[i]<<"\t";
21     }
22     getch();
23 }
24 int findSmallest(int myarray[],int i) {
25     int ele_small,position,j;
26     ele_small = myarray[i];
27     position = i;
28     for(j=i+1;j<5;j++) {
29         if(myarray[j]<ele_small) {
30             ele_small = myarray[j];
31             position=j;
32         }
33     }
34     return position;
35 }
```

The output window shows the following text:

```
list element array
12    45    8    15    33
Sorted array
8    12    15    33    45
```

Program dimulai dengan mendefinisikan fungsi `findSmallest`, yang akan mencari elemen terkecil dalam array yang belum diurutkan.

Di dalam fungsi `main`, terdapat array `myarray` yang berisi 5 bilangan integer tak terurut. Program mencetak daftar array awal dengan menggunakan perulangan `for` pertama. Nilai-nilai array dicetak satu per satu dengan menggunakan `cout`. Kemudian, program melakukan pengurutan array dengan menggunakan nested loop `for`. Loop pertama (variabel `i`) digunakan untuk mengiterasi elemen-elemen array dari indeks 0 hingga 4.

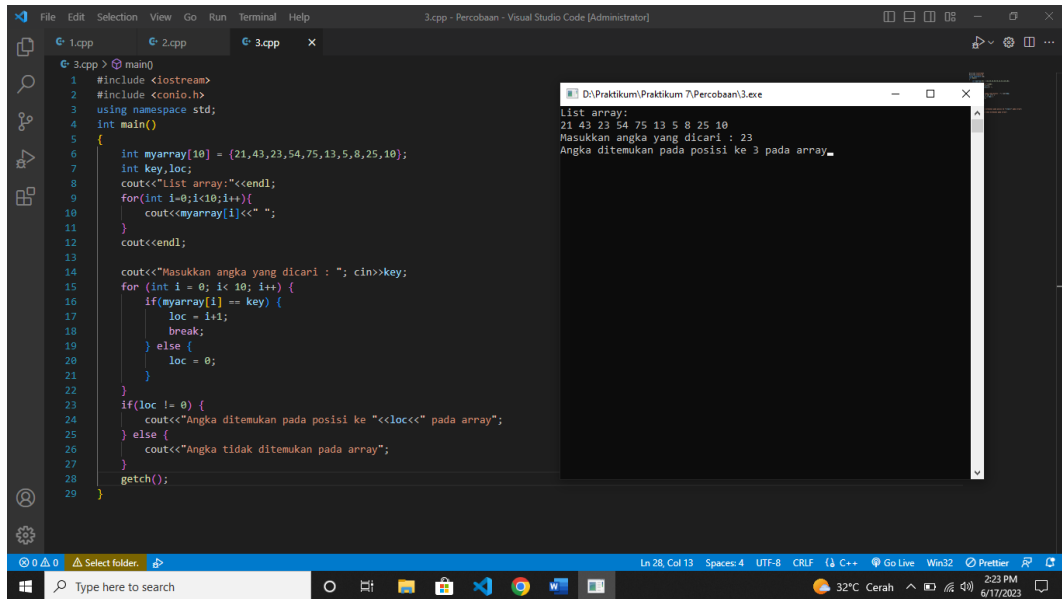
Di dalam loop pertama, program memanggil fungsi `findSmallest` untuk mencari elemen terkecil dalam array yang belum diurutkan, dimulai dari indeks `i`. Fungsi `findSmallest` akan mengembalikan posisi (indeks) elemen terkecil tersebut.

Setelah mendapatkan posisi elemen terkecil, program menukar nilai elemen pada indeks `i` dengan elemen pada posisi terkecil tersebut. Hal ini dilakukan dengan menggunakan variabel `temp` sebagai variabel penampung sementara.

Program melanjutkan loop pertama untuk mengulangi proses pengurutan, hingga semua elemen dalam array terurut. Setelah selesai melakukan pengurutan, program mencetak daftar array yang telah diurutkan menggunakan perulangan `for` kedua.

## 2.3 Percobaan 7.3





The screenshot shows the Visual Studio Code editor with a C++ file named 3.cpp. The code defines an array of 10 integers, prints it, prompts the user for a search key, and then searches for the key. The output window shows the array [21, 43, 23, 54, 75, 13, 5, 8, 25, 10], the input key 23, and the result: 'Angka ditemukan pada posisi ke 3 pada array'.

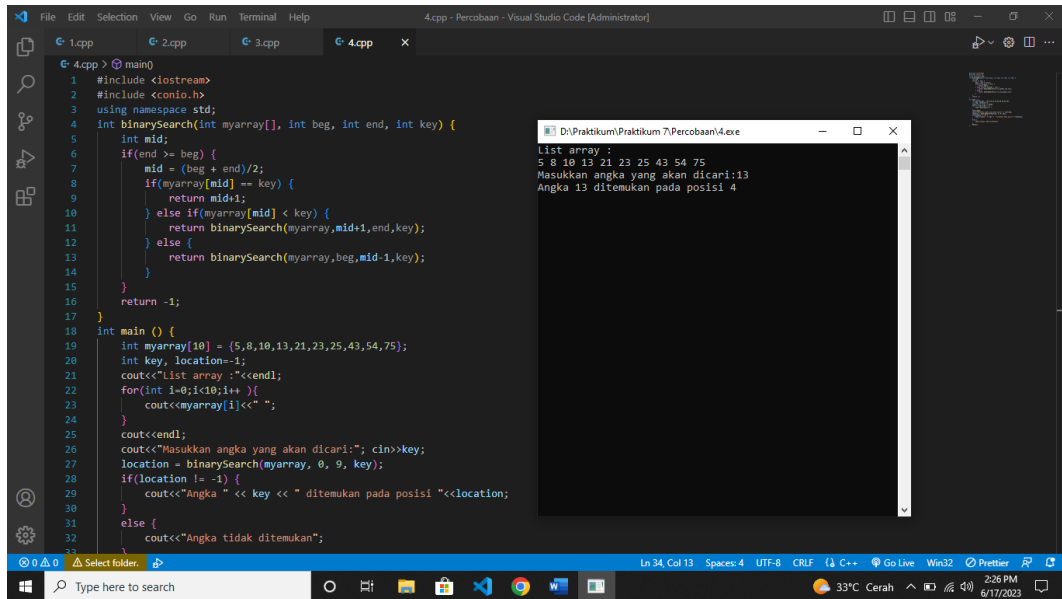
```
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4 int main()
5 {
6     int myarray[10] = {21,43,23,54,75,13,5,8,25,10};
7     int key,loc;
8     cout<<"List array:"<<endl;
9     for(int i=0;i<10;i++){
10         cout<<myarray[i]<<" ";
11     }
12     cout<<endl;
13
14     cout<<"Masukkan angka yang dicari : "; cin>>key;
15     for (int i = 0; i < 10; i++) {
16         if(myarray[i] == key) {
17             loc = i+1;
18             break;
19         } else {
20             loc = 0;
21         }
22     }
23     if(loc != 0) {
24         cout<<"Angka ditemukan pada posisi ke "<<loc<<" pada array";
25     } else {
26         cout<<"Angka tidak ditemukan pada array";
27     }
28     getch();
29 }
```

Output:

```
List array:
21 43 23 54 75 13 5 8 25 10
Masukkan angka yang dicari : 23
Angka ditemukan pada posisi ke 3 pada array
```

Program dimulai dengan mendefinisikan array myarray yang berisi 10 bilangan integer. Program mencetak daftar array awal dengan menggunakan perulangan for pertama. Nilai-nilai array dicetak satu per satu dengan menggunakan cout. Selanjutnya, program meminta pengguna untuk memasukkan angka yang ingin dicari dengan menggunakan cin dan disimpan dalam variabel key. Program melakukan pencarian menggunakan perulangan for kedua. Setiap elemen dalam array diperiksa apakah sama dengan angka yang dicari (key). Jika ditemukan, program menyimpan posisi elemen tersebut dalam variabel loc (dengan menambahkan 1 karena indeks array dimulai dari 0) dan keluar dari loop menggunakan break. Jika angka ditemukan, program mencetak pesan yang menyatakan angka tersebut ditemukan pada posisi ke loc dalam array. Jika angka tidak ditemukan, variabel loc tetap bernilai 0, dan program mencetak pesan yang menyatakan angka tidak ditemukan dalam array.

## 2.4 Percobaan 7.4



```
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4 int binarySearch(int myarray[], int beg, int end, int key) {
5     int mid;
6     if(end >= beg) {
7         mid = (beg + end)/2;
8         if(myarray[mid] == key) {
9             return mid+1;
10        } else if(myarray[mid] < key) {
11            return binarySearch(myarray, mid+1, end, key);
12        } else {
13            return binarySearch(myarray, beg, mid-1, key);
14        }
15    }
16    return -1;
17 }
18 int main () {
19     int myarray[10] = {5,8,10,13,21,23,25,43,54,75};
20     int key, location=-1;
21     cout<<"List array :<<endl;
22     for(int i=0; i<10; i++){
23         cout<<myarray[i]<<" ";
24     }
25     cout<<endl;
26     cout<<"Masukkan angka yang akan dicari: "; cin>>key;
27     location = binarySearch(myarray, 0, 9, key);
28     if(location != -1) {
29         cout<<"Angka " << key << " ditemukan pada posisi "<<location;
30     }
31     else {
32         cout<<"Angka tidak ditemukan";
33     }
```

```
List array :
5 8 10 13 21 23 25 43 54 75
Masukkan angka yang akan dicari:13
Angka 13 ditemukan pada posisi 4
```

Program dimulai dengan mendefinisikan fungsi `binarySearch`, yang merupakan implementasi rekursif dari algoritma binary search. Fungsi ini menerima array `myarray`, indeks awal (`beg`), indeks akhir (`end`), dan angka yang dicari (`key`) sebagai argumen. Di dalam fungsi `binarySearch`, jika indeks akhir (`end`) masih lebih besar atau sama dengan indeks awal (`beg`), program melakukan langkah-langkah berikut:

- Menghitung nilai tengah (`mid`) dengan cara menjumlahkan indeks awal dan indeks akhir, lalu membaginya dengan 2.
- Jika elemen tengah (`myarray[mid]`) sama dengan angka yang dicari (`key`), maka angka tersebut ditemukan dan program mengembalikan nilai `mid+1`.
- Jika elemen tengah lebih kecil dari angka yang dicari, maka panggil rekursif fungsi `binarySearch` dengan indeks awal yang baru (`mid+1`) dan indeks akhir yang sama.
- Jika elemen tengah lebih besar dari angka yang dicari, maka panggil rekursif fungsi `binarySearch` dengan indeks awal yang sama dan indeks akhir yang baru (`mid-1`). Fungsi `binarySearch` mengembalikan -1 jika angka tidak ditemukan dalam array.

Di dalam fungsi `main`, terdapat array `myarray` yang berisi 10 bilangan integer yang sudah terurut. Program mencetak daftar array awal dengan menggunakan perulangan `for`. Nilai-nilai array dicetak satu per satu dengan menggunakan `cout`.

Program meminta pengguna untuk memasukkan angka yang ingin dicari dengan menggunakan cin dan menyimpannya dalam variabel key. Program memanggil fungsi binarySearch dengan mempassing array myarray, indeks awal (0), indeks akhir (9), dan angka yang dicari (key). Variabel location akan memperoleh hasil yang dikembalikan oleh fungsi binarySearch. Jika hasilnya bukan -1, maka angka ditemukan dalam array dan program mencetak pesan yang menyatakan angka tersebut ditemukan pada posisi location. Jika hasilnya -1, maka angka tidak ditemukan dalam array dan program mencetak pesan yang menyatakan angka tidak ditemukan.



## BAB III

### TUGAS PRAKTIKUM

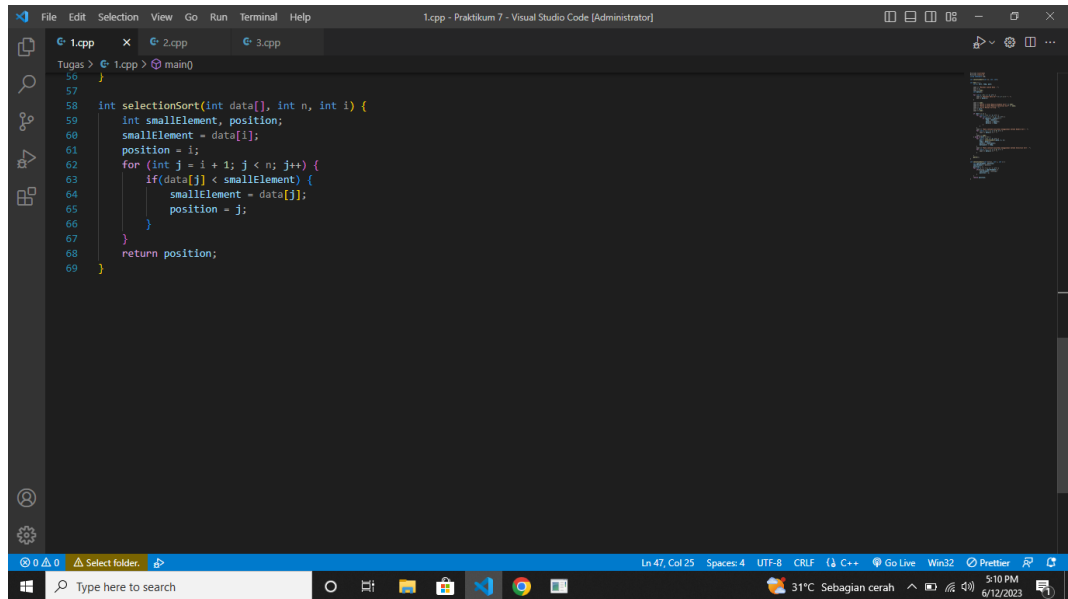
#### 3.1 Tugas 1

```
1.cpp - Praktikum 7 - Visual Studio Code [Administrator]
File Edit Selection View Go Run Terminal Help
1.cpp x 2.cpp 3.cpp
Tugas > 1.cpp > main()
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4
5 int selectionSort(int [], int, int);
6
7 int main () {
8     int n, Sort, temp, post;
9
10    cout << "Masukan Jumlah Data : ";
11    cin >> n;
12    cout << endl;
13    int data[n];
14
15    for (int i = 0; i < n; i++) {
16        cout << "Masukan Data ke-" << i + 1 << " : ";
17        cin >> data[i];
18    }
19
20    cout << endl;
21    cout << "Ketik 1 untuk Memilih Bubble Sort" << endl;
22    cout << "Ketik 2 untuk Memilih Selection Sort" << endl;
23    cout << "Pilih Metode Sorting : ";
24    cin >> Sort;
25    cout << endl;
26
27    if (Sort == 1) {
28        for (int i = 0; i < n; i++) {
29            for (int j = i + 1; j < n; j++) {
30                if (data[j] < data[i]) {
31                    temp = data[i];
32                    data[i] = data[j];
33                    data[j] = temp;
34                }
35            }
36        }
37        cout << "Data setelah diurutkan menggunakan metode Bubble Sort : ";
38        for (int i = n - 1; i >= 0; i--) {
39            cout << data[i] << " ";
40        }
41        cout << endl;
42    } else if (Sort == 2) {
43        for (int i = 0; i < n; i++) {
44            post = selectionSort(data, n, i);
45            temp = data[i];
46            data[i] = data[post];
47            data[post] = temp;
48        }
49        cout << "Data setelah diurutkan menggunakan metode Selection Sort : ";
50        for (int i = n - 1; i >= 0; i--) {
51            cout << data[i] << " ";
52        }
53    }
54    getch();
55 }
56
57 int selectionSort(int data[], int n, int i) {
58     int smallestElement, position;
59     smallestElement = data[i];
60     position = i;
61     for (int j = i + 1; j < n; j++) {
62         if (data[j] < smallestElement) {
63             smallestElement = data[j];
64             position = j;
65         }
66     }
67     temp = data[i];
68     data[i] = data[position];
69     data[position] = temp;
70     return position;
71 }
```

Masukan Data ke-1 : 12  
Masukan Data ke-2 : 5  
Masukan Data ke-3 : 23  
Masukan Data ke-4 : 1  
Masukan Data ke-5 : 4

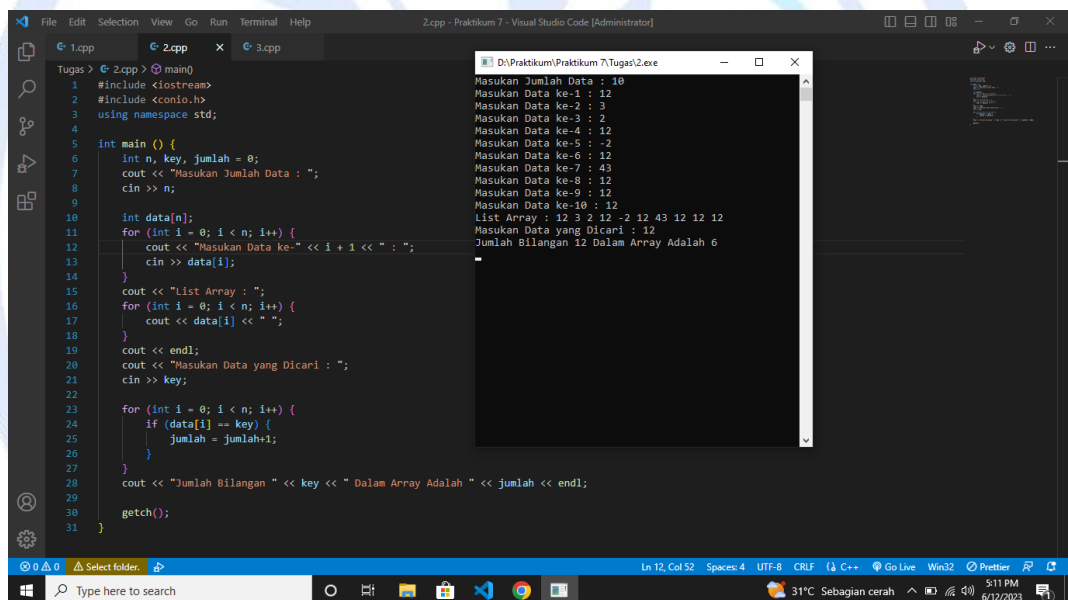
Ketik 1 untuk Memilih Bubble Sort  
Ketik 2 untuk Memilih Selection Sort  
Pilih Metode Sorting : 2

Data setelah diurutkan menggunakan metode Selection Sort : 23 12 5 4 1



```
1.cpp - Visual Studio Code [Administrator]
File Edit Selection View Go Run Terminal Help
Tugas > 1.cpp > main()
56 }
57
58 int selectionSort(int data[], int n, int i) {
59     int smallElement, position;
60     smallElement = data[i];
61     position = i;
62     for (int j = i + 1; j < n; j++) {
63         if (data[j] < smallElement) {
64             smallElement = data[j];
65             position = j;
66         }
67     }
68     return position;
69 }
```

### 3.2 Tugas 2



```
2.cpp - Visual Studio Code [Administrator]
File Edit Selection View Go Run Terminal Help
Tugas > 2.cpp > main()
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4
5 int main () {
6     int n, key, jumlah = 0;
7     cout << "Masukan Jumlah Data : ";
8     cin >> n;
9
10    int data[n];
11    for (int i = 0; i < n; i++) {
12        cout << "Masukan Data ke-" << i + 1 << " : ";
13        cin >> data[i];
14    }
15    cout << "List Array : ";
16    for (int i = 0; i < n; i++) {
17        cout << data[i] << " ";
18    }
19    cout << endl;
20    cout << "Masukan Data yang Dicari : ";
21    cin >> key;
22
23    for (int i = 0; i < n; i++) {
24        if (data[i] == key) {
25            jumlah = jumlah+1;
26        }
27    }
28    cout << "Jumlah Bilangan " << key << " Dalam Array Adalah " << jumlah << endl;
29    getch();
30 }
31 }
```

D:\Praktikum\Praktikum 7\Tugas2.exe

Masukan Jumlah Data : 10  
Masukan Data ke-1 : 12  
Masukan Data ke-2 : 3  
Masukan Data ke-3 : 2  
Masukan Data ke-4 : 12  
Masukan Data ke-5 : -2  
Masukan Data ke-6 : 12  
Masukan Data ke-7 : 43  
Masukan Data ke-8 : 12  
Masukan Data ke-9 : 12  
Masukan Data ke-10 : 12  
List Array : 12 3 2 12 -2 12 43 12 12 12  
Masukan Data yang Dicari : 12  
Jumlah Bilangan 12 Dalam Array Adalah 6

### 3.3 Tugas 3

```
D:\Praktikum\Praktikum 7\Tugas3.exe
1. Menampilkan data-data barang
2. Menambah stok barang yang sudah habis
3. Mengganti harga barang
4. Mengurutkan data barang secara descending

Program mana yang ingin anda lakukan? : 1
Kode Barang Nama Barang Harga Stok
1 Pisau 2500 5
2 Indomie 3500 0
3 Aqua 4500 400
4 Mie Sedap 3500 0

Ingin keluar? [Y/N] : N

1. Menampilkan data-data barang
2. Menambah stok barang yang sudah habis
3. Mengganti harga barang
4. Mengurutkan data barang secara descending

Program mana yang ingin anda lakukan? : 4
Kode Barang Nama Barang Harga Stok
4 Mie Sedap 3500 0
3 Aqua 4500 400
2 Indomie 3500 0
1 Pisau 2500 5

Ingin keluar? [Y/N] : N

1. Menampilkan data-data barang
2. Menambah stok barang yang sudah habis
3. Mengganti harga barang
4. Mengurutkan data barang secara descending

Program mana yang ingin anda lakukan? : 2
Barang apa yang ingin anda tambah stoknya : Indomie
Stok barang menjadi : 12

Kode Barang Nama Barang Harga Stok
1 Pisau 2500 5
2 Indomie 3500 12
3 Aqua 4500 400
4 Mie Sedap 3500 0

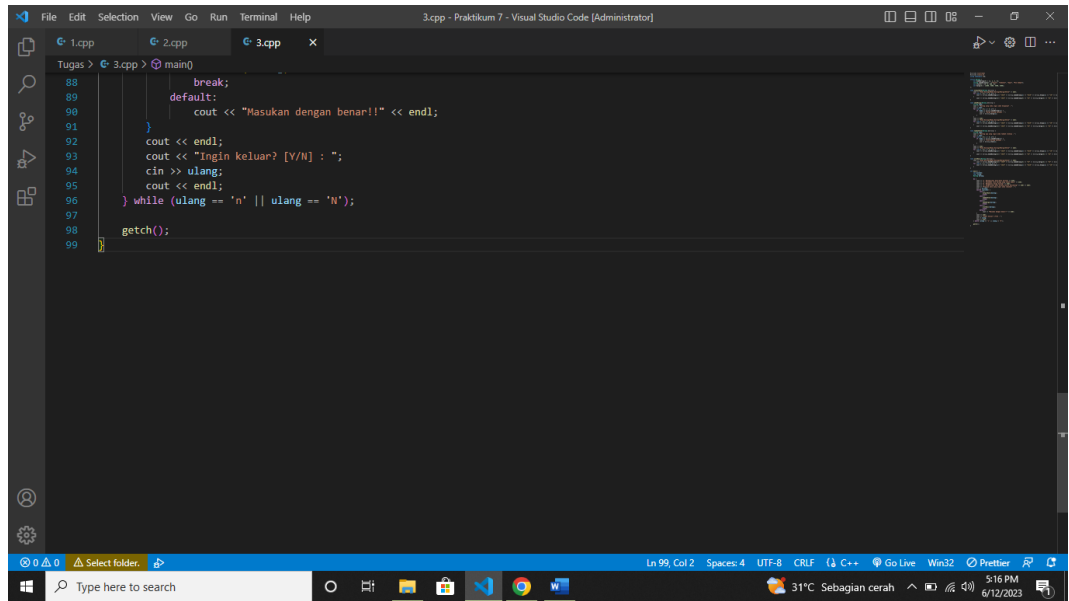
Ingin keluar? [Y/N] :
```

```
File Edit Selection View Go Run Terminal Help 3.cpp - Praktikum 7 - Visual Studio Code [Administrator]
3.cpp x
Tugas > 3.cpp > main()
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4
5 struct Barang {
6     int kodeBarang[4] = {1, 2, 3, 4};
7     string namaBarang[4] = {"Pisau", "Indomie", "Aqua", "Mie Sedap"};
8     int stok[4] = {5, 0, 400, 0};
9     int harga[4] = {2500, 3500, 4500, 3500};
10 };
11
12 void outputData(Barang &barang) {
13     cout << "Kode Barang\tNama Barang\tHarga\tStok" << endl;
14     for (int i = 0; i < 4; i++) {
15         cout << barang.kodeBarang[i] << "\t\t" << barang.namaBarang[i] << "\t\t" << barang.harga[i] << "\t" << barang.stok[i] << endl;
16     }
17     cout << barang.kodeBarang[3] << "\t\t" << barang.namaBarang[3] << "\t" << barang.harga[3] << "\t" << barang.stok[3] << endl;
18 }
19
20 void ubahHarga(Barang &barang) {
21     string nama;
22     cout << "Barang yang anda ingin ubah harganya? : ";
23     cin >> nama;
24     for (int i = 0; i < 4; i++) {
25         if (nama == barang.namaBarang[i]) {
26             cout << "Harga barang menjadi : ";
27             cin >> barang.harga[i];
28         }
29     }
30     cout << endl;
31     cout << "Kode Barang\tNama Barang\tHarga\tStok" << endl;
32     for (int i = 0; i < 4; i++) {
33         cout << barang.kodeBarang[i] << "\t\t" << barang.namaBarang[i] << "\t\t" << barang.harga[i] << "\t" << barang.stok[i] << endl;
34     }
35 }
```



```
File Edit Selection View Go Run Terminal Help 3.cpp - Praktikum 7 - Visual Studio Code [Administrator]
Tugas > 3.cpp > main()
27     cin >> barang.harga[1];
28 }
29 }
30 cout << endl;
31 cout << "Kode Barang\tNama Barang\tHarga\tStok" << endl;
32 for (int i = 0; i < 3; i++) {
33     cout << barang.kodeBarang[i] << "\t\t" << barang.namaBarang[i] << "\t\t" << barang.harga[i] << "\t" << barang.stok[i] << endl;
34 }
35     cout << barang.kodeBarang[3] << "\t\t" << barang.namaBarang[3] << "\t" << barang.harga[3] << "\t" << barang.stok[3] << endl;
36 }
37
38 void tambahStok(Barang &barang) {
39     string nama;
40     cout << "Barang apa yang ingin anda tambah stoknya : ";
41     cin >> nama;
42     for (int i = 0; i < 4; i++) {
43         if (nama == barang.namaBarang[i]) {
44             cout << "Stok barang menjadi : ";
45             cin >> barang.stok[i];
46         }
47     }
48     cout << endl;
49     cout << "Kode Barang\tNama Barang\tHarga\tStok" << endl;
50     for (int i = 0; i < 3; i++) {
51         cout << barang.kodeBarang[i] << "\t\t" << barang.namaBarang[i] << "\t\t" << barang.harga[i] << "\t" << barang.stok[i] << endl;
52     }
53     cout << barang.kodeBarang[3] << "\t\t" << barang.namaBarang[3] << "\t" << barang.harga[3] << "\t" << barang.stok[3] << endl;
54 }
55
56 void urutDesc(Barang &barang) {
57     cout << "Kode Barang\tNama Barang\tHarga\tStok" << endl;
58     cout << barang.kodeBarang[3] << "\t\t" << barang.namaBarang[3] << "\t" << barang.harga[3] << "\t" << barang.stok[3] << endl;
59     for (int i = 2; i >= 0; i--) {
```

```
File Edit Selection View Go Run Terminal Help 3.cpp - Praktikum 7 - Visual Studio Code [Administrator]
Tugas > 3.cpp > main()
54 }
55 }
56 void urutDesc(Barang &barang) {
57     cout << "Kode Barang\tNama Barang\tHarga\tStok" << endl;
58     cout << barang.kodeBarang[3] << "\t\t" << barang.namaBarang[3] << "\t" << barang.harga[3] << "\t" << barang.stok[3] << endl;
59     for (int i = 2; i >= 0; i--) {
60         cout << barang.kodeBarang[i] << "\t\t" << barang.namaBarang[i] << "\t\t" << barang.harga[i] << "\t" << barang.stok[i] << endl;
61     }
62 }
63
64 int main() {
65     int pilihan;
66     char ulang;
67     Barang barang;
68
69     do {
70         cout << "1. Menampilkan data-data barang" << endl;
71         cout << "2. Menambah stok barang yang sudah habis" << endl;
72         cout << "3. Mengganti harga barang" << endl;
73         cout << "4. Mengurutkan data barang secara descending" << endl << endl;
74         cout << "Program mana yang ingin anda lakukan? : ";
75         cin >> pilihan;
76         switch (pilihan) {
77             case 1:
78                 outputData(barang);
79                 break;
80             case 2:
81                 tambahStok(barang);
82                 break;
83             case 3:
84                 ubahHarga(barang);
85                 break;
```



```
88 break;
89 default:
90     cout << "Masukan dengan benar!!" << endl;
91 }
92 cout << endl;
93 cout << "Ingin keluar? [Y/N] : ";
94 cin >> ulang;
95 cout << endl;
96 } while (ulang == 'n' || ulang == 'N');
97
98 getch();
99
```



## **BAB IV**

### **ANALISA**

#### **4.1 Tugas x**

##### **4.1.1 Poin A**

Program dimulai dengan mendefinisikan fungsi selectionSort. Fungsi ini menerima array data, panjang array n, dan indeks awal i sebagai argumen. Fungsi ini mengimplementasikan algoritma Selection Sort untuk mencari elemen terkecil dalam array yang belum diurutkan.

##### **4.1.2 Poin B**

Di dalam fungsi main, program meminta pengguna untuk memasukkan jumlah data (n) yang akan diurutkan. Program membuat array data dengan ukuran sesuai input pengguna.

Program menggunakan perulangan for untuk meminta pengguna memasukkan data satu per satu ke dalam array. Setelah pengguna selesai memasukkan data, program menampilkan pilihan metode pengurutan, yaitu Bubble Sort atau Selection Sort, dan pengguna diminta untuk memilih salah satu metode.

##### **4.1.3 Poin C**

Jika pengguna memilih Bubble Sort (input 1), program mengurutkan data menggunakan algoritma Bubble Sort. Program menggunakan nested loop for untuk membandingkan dan menukar elemen-elemen array jika diperlukan. Setelah selesai melakukan pengurutan menggunakan Bubble Sort, program mencetak data yang sudah diurutkan secara menurun (dari terbesar ke terkecil).

##### **4.1.4 Poin D**

Jika pengguna memilih Selection Sort (input 2), program mengurutkan data menggunakan algoritma Selection Sort. Program menggunakan loop for untuk mengiterasi setiap elemen array dan memanggil fungsi selectionSort untuk mencari elemen terkecil yang belum diurutkan.

Setelah selesai melakukan pengurutan menggunakan Selection Sort, program mencetak data yang sudah diurutkan secara menurun (dari terbesar ke terkecil).

#### **4.2 Tugas 2**

##### **4.2.1 Poin A**

Program meminta pengguna untuk memasukkan jumlah data (n) yang akan digunakan.

Program membuat array data dengan ukuran sesuai input pengguna. Menggunakan perulangan for, program meminta pengguna untuk memasukkan data satu per satu ke dalam array. Setelah pengguna selesai memasukkan data, program mencetak list array dengan menggunakan perulangan for.

#### 4.2.2 Poin B

Program meminta pengguna untuk memasukkan angka yang ingin dicari (key). Menggunakan perulangan for, program mencari jumlah kemunculan angka tersebut dalam array. Jika angka pada indeks i sama dengan key, maka jumlah diincrement. Setelah selesai mencari, program mencetak jumlah kemunculan angka tersebut dalam array.

### 4.3 Tugas 3

#### 4.3.1 Poin A

Program mendefinisikan sebuah struktur Barang yang memiliki empat array, yaitu kodeBarang, namaBarang, stok, dan harga. Setiap array memiliki empat elemen untuk menyimpan informasi tentang empat barang.

#### 4.3.2 Poin B

Program mendefinisikan empat fungsi yaitu outputData, ubahHarga, tambahStok, dan urutDesc. Fungsi outputData digunakan untuk mencetak data barang, fungsi ubahHarga digunakan untuk mengubah harga barang, fungsi tambahStok digunakan untuk menambah stok barang, dan fungsi urutDesc digunakan untuk mengurutkan data barang secara descending (menurun).

#### 4.3.3 Poin C

Di dalam fungsi main, program menggunakan loop do-while untuk memungkinkan pengguna melakukan berbagai tindakan pada data barang. Program menampilkan menu pilihan tindakan kepada pengguna dan meminta pengguna untuk memilih tindakan yang diinginkan. Berdasarkan pilihan pengguna, program akan memanggil fungsi yang sesuai untuk menjalankan tindakan tersebut pada data barang.

#### 4.3.4 Poin D

Setelah tindakan selesai dilakukan, program akan menanyakan apakah pengguna ingin melanjutkan atau keluar dari program. Jika pengguna memilih untuk melanjutkan, program akan kembali ke menu pilihan. Jika pengguna memilih untuk keluar, program akan berakhir.



## **BAB V**

### **PENUTUP**

#### **5.1 Kesimpulan**

- 5.1.1 Pengurutan (sorting) data dalam struktur data sangat penting terutama untuk data yang beripe data numerik ataupun karakter. Bubble Sort adalah metode yang membandingkan elemen yang sekarang dengan elemen-elemen berikutnya. Selection sort merupakan kombinasi antara sorting dan searching. Untuk setiap proses, akan dicari elemen-elemen yang belum diurutkan yang memiliki nilai terkecil atau terbesar akan dipertukarkan ke posisi yang tepat di dalam array.
- 5.1.2 Searching merupakan proses fundamental dalam pengelolaan data. Proses pencarian adalah menemukan nilai (data) tertentu di dalam sekumpulan data yang bertipe sama (baik bertipe dasar atau bertipe bentukan). Sequential Search atau sering disebut pencarian linier. Menggunakan prinsip data yang ada di bandingkan satu persatu secara berurutan dengan yang dicari. Binary search adalah sebuah algoritma pencarian dengan cara membagi data menjadi dua bagian setiap kali terjadi proses pencarian untuk menemukan nilai tertentu dalam sebuah larik (array) linear.

#### **5.2 Saran**

Dalam membuat program C++ perlu diperhatikan huruf kecil atau besar nama array, indeks yang diinput juga harus sesuai dengan yang telah dibuat karena jika indeksnya lebih dari yang ada akan berpengaruh pada programnya sendiri serta bisa sampai menyebabkan error. Selain itu juga harus lebih sering berlatih dan belajar membuat program menggunakan metoda sorting dan searching agar lebih cepat dalam memahami bagaimana pengaplikasian metoda sorting dan searching dalam C++.



## DAFTAR PUSTAKA

Anonim. (2023). *MODUL 7 PRAKTIKUM ALGORITMA DAN PEMECAHAN MASALAH*. Laboratorium Komputer dan Jaringan.

Sanctuarynyx, R. (2014, Mei 04). Retrieved from  
<https://reionnote.wordpress.com/2014/05/04/searching-dan-sorting/>

Ulum, M. B. (2018). *Algoritma Pencarian dan Pengurutan*. Universitas Esa Unggul.

