MODUL VIII
SEARCHING

I. TUJUAN

1. Memahami dan menerapkan beberapa algoritma pencarian (searching) dalam menyelesaikan berbagai studi kasus

II. DASAR TEORI

1. Konsep Dasar Searching

- Searching adalah proses mendapatkan (retrieve) informasi berdasarkan kunci tertentu dari sejumlah informasi yang telah disimpan.

2. Sequential Search

Konsep dasar Algoritma Sequential Search:

- Terdapat sebuah larik yang berisi n buah data (L[0], L[1], … , L[n-1])
- i adalah bilangan indeks terkecil , yang memenuhi kondisi $0 \leq k \leq n-1$
- k adalah data yang dicari → L[i]=k

Algoritma Sequential Search:

- Berikut ini merupakan implementasi algoritma pencarian secara sekuensial.
- Subrutin akan menghasilkan nilai balik berupa:

a. -1 jika data yang dicari tidak ditemukan dan
b. Bilangan antara 0 sampai dengan n-1 (dengan n adalah jumlah elemen larik) jika data yang dicari ditemukan

Pseudocode Sequential Search:

```
SUBRUTIN cari (L, n, k)
       JIKA n ≤ 0 MAKA
              posisi ← - 1
       SEBALIKNYA
              ketemu ← SALAH
              i ← 0
              ULANG SELAMA (i<n-1) DAN (TIDAK ketemu)
                     JIKA K = L[i] MAKA
                            posisi ← i
                            ketemu ← BENAR
                     SEBALIKNYA
                            i ← i + 1
                     AKHIR-JIKA
              AKHIR ULANG
              JIKA TIDAK ketemu MAKA
                     posisi ← -1
              AKHIR-JIKA
            AKHIR-JIKA
       NILAI-BALIK posisi
AKHIR-SUBRUTIN
```

Dengan: **L = larik; n = jml elemen larik;** dan **k = data yang dicari**

Fungsi Sequential Search dalam bahasa C++:

```
int sequential_search(int data [], int n, int k) {
    int posisi, i, ketemu;

    if (n <= 0)
        posisi = -1;
    else {
        ketemu = 0;
        i = 1;
        while ((i < n -1) && ! ketemu) {
            if (data [i] == k)  {
                posisi = i;
                ketemu = 1;
            } else {
                i++;
            }

            if (!ketemu) {
                posisi = -1;
            }
        }
    }
    return posisi;
}
```

Dari fungsi tersebut dapat diketahui jika program menemukan kunci yang dicari maka fungsi akan mengembalikan posisi indeks dari kunci tersebut pada array. Sedangkan jika program tidak menemukan kunci maka program akan mengembalikan nilai -1.

Contoh

Diketahui array suatu integer terdiri dari **7** elemen sebagai berikut; **{1, 0, 3, 2, 5, 7, 9}.** Contoh Program untuk mencari angka/ kunci **3** pada array tersebut adalah sebagai berikut:

```
#include <iostream>

using namespace std;

int sequential_search(int data [], int n, int k) {
    int posisi, i, ketemu;

    if (n <= 0)
        posisi = -1;
    else {
        ketemu = 0;
        i = 1;
        while ((i < n -1) && ! ketemu) {
            if (data [i] == k)  {
                posisi = i;
                ketemu = 1;
            } else {
                i++;
            }

            if (!ketemu) {
                posisi = -1;
```

```
                }
            }
        }
        return posisi;
    }

    int main() {

        int n       = 7;
        int data[]  = {1, 0, 3, 2, 5, 7, 9};
        int k       = 4;

        int posisi = sequential_search(data, n, k);

        if(posisi != -1) {
            cout << "kunci " << k << " ditemukan pada posisi indeks ke-"
    << posisi << endl;
        } else {
            cout << "kunci " << k << " tidak ditemukan" << endl;
        }

        return 0;
}
```

3. Binary Search

Konsep dasar Algoritma Bubble Sort

- Apabila kumpulan data sudah dalam keadaan **terurut**, pencarian data dengan menggunakan pencarian sekuensial akan memakan waktu yang lama jika jumlah data dalam kumpulan data tersebut sangat banyak.
- Pencarian biner dilakukan dengan membagi larik menjadi dua bagian dengan jumlah yang sama atau berbeda 1 jika jumlah data semula ganjil
- Data yang dicari kemudian dibandingkan dengan data tengah pada bagian pertama
- Dalam hal ini akan terjadi 3 kemungkinan yang terjadi :
    - Data yang dicari sama dengan elemen tengah pada bagian pertama dalam larik. Jika kondisi ini terpenuhi, data yang dicari berarti ditemukan.
    - Data yang dicari bernilai kurang dari nilai elemen tengah pada bagian pertama dalam larik. Pada keadaan ini, pencarian diteruskan pada bagian pertama.
    - Data yang dicari bernilai lebih dari nilai elemen tengah pada bagian pertama dalam larik. Pada keadaan ini, pencarian diteruskan pada bagian kedua.

Pseudocode Binary Search

```
SUBRUTIN cari_biner (L, n, k)
      ada ← SALAH
      bawah ← 0
      atas ← n – 1
      ULANG SELAMA  atas ≥ bawah
            tengah ← (atas + bawah) / 2  // Pembagi Bulat
            JIKA k >L[tengah] MAKA   // Data dicari dikanan
            bawah ← tengah + 1
            SEBALIKNYA
            JIKA k >L[tengah] MAKA // Data dicari dikiri
            atas = tengah – 1
```

```
            SEBALIKNYA
            ada ← BENAR
            posisi ← tengah
            bawah ← atas + 1  // Supaya perulangan berakhir
                  AKHIR-JIKA
                   AKHIR-JIKA
        AKHIR-ULANG
        JIKA TIDAK ada MAKA
               posisi ← -1
        AKHIR-JIKA
        NILAI-BALIK posisi
AKHIR-SUBRUTIN
```

Fungsi Binary Search dalam Bahasa C++:

```
int function_binary_search (int data [], int n, int k) {
    int atas, bawah, tengah, posisi;
    bool ada;

    ada     = false;
    bawah   = 0;
    atas    = n - 1;
    posisi  = -1;

    while (bawah <= atas) {
       tengah = (atas + bawah)/2;
       if(k == data[tengah]) {
              posisi = tengah;
                    break;
              }
       else if(k < data[tengah])  atas = tengah - 1;
       else if(k > data[tengah]) bawah = tengah + 1;
       }

       return posisi;
}
```

III. GUIDED

1. Pencarian data numerik bertipe double menggunakan Algoritma Sequential Search:

```
#include <iostream>

using namespace std;

int sequential_search(double data [], int n, double k) {
    int posisi, i, ketemu;

    if (n <= 0)
        posisi = -1;
    else {
        ketemu = 0;
        i = 1;
        while ((i < n -1) && ! ketemu) {
            if (data [i] == k)  {
                posisi = i;
                ketemu = 1;
            } else {
                i++;
            }
```

```
            if (!ketemu) {
                posisi = -1;
            }
        }
    }
    return posisi;
}

int main() {

    int n           = 7;
    double data[]   = {1.3, 0.9, 3.2, 2.1, 5.8, 7.7, 9.2};
    double k        = 0.9;

    int posisi = sequential_search(data, n, k);

    if(posisi != -1) {
        cout << "kunci " << k << " ditemukan pada posisi indeks ke-" <<
posisi << endl;
    } else {
        cout << "kunci " << k << " tidak ditemukan" << endl;
    }

    return 0;
}
```

2. Pencarian data karakter menggunakan Algoritma Binary Search

```
#include <iostream>

using namespace std;

int function_binary_search (char data [], int n, char k) {
    int atas, bawah, tengah, posisi;
    bool ada;

    ada     = false;
    bawah   = 0;
    atas    = n - 1;
    posisi  = -1;

    while (bawah <= atas) {
       tengah = (atas + bawah)/2;
       if(k == data[tengah]) {
              posisi = tengah;
                     break;
              }
       else if(k < data[tengah])  atas = tengah - 1;
       else if(k > data[tengah]) bawah = tengah + 1;
       }

       return posisi;
}

int main() {

    int n       = 7;
    char data[] = {'a', 'b', 'c', 'e', 'g', 'h', 'j'};
    char k      = 'c';
```

```
    int posisi = function_binary_search(data, n, k);

    if(posisi != -1) {
        cout << "kunci " << k << " ditemukan pada posisi indeks ke-" <<
posisi << endl;
    } else {
        cout << "kunci " << k << " tidak ditemukan" << endl;
    }

    return 0;
}
```

IV. TUGAS

1. Pak Polisi memiliki database plat nomor mobil. Terdapat 10 nomor dalam database tersebut, yaitu sebagai berikut: R 300 SR, R 1234 DJ, R 3218 RR, R 701 LP, R 007 TU, R 3 ST, R 999 RT, R 210 RO, R 1111 II, dan R 4987 LH. Pada suatu hari Pak Polisi tersebut melihat kendaaraan ber-nomor R 999 RS berada di area dilarang parkir. Bantulah Pak Polisi tersebut untuk mengecek apakah nomor tersebut terdapat di dalam database atau tidak! Gunakan Algoritma Sequential Search! (Score: 30)

2. Dalam satu kelas terdapat 13 mahasiswa yang memiliki nim sebagai berikut: 12102001, 12102002, 12102003, 12102004, 12102005, 12102006, 12102007, 12102008, 12102009, 12102010, 12102011, 12102012, dan 12102013. Dengan menggunakan Algoritma Binary Search, carilah NIM 12102011 apakah berada di kelas tersebut atau tidak. (Score: 30).

3. Pak Anto membuat program untuk men-generate bilangan acak. Saat program dijalankan, program memberikan daftar bilangan acak sebagai berikut: 21, 61, 28, 72, 44, 68, 37, 52, 75, dan 75. Bantulah Pak Anto membuat program pencarian untuk bilangan acak tersebut dengan menggunakan Algoritma Binary Search. Angka yang dicari adalah 71. (Score: 40)