BAB 3
SEARCHING

A. TUJUAN

1. Mahasiswa dapat melakukan perancangan aplikasi menggunakan struktur *Searching* (Pencarian)
2. Mahasiswa mampu melakukan analisis pada algoritma *Searching* yang dibuat
3. Mahasiswa mampu mengimplementasikan algoritma *Searching* pada sebuah aplikasi secara tepat dan efisien
4. Mahasiswa mampu menjelaskan mengenai algoritma *Searching*.
5. Mahasiswa mampu membuat dan mendeklarasikan struktur algoritma *Searching.*
6. Mahasiswa mampu menerapkan dan mengimplementasikan algoritma *Searching.*

B. ALOKASI WAKTU

4js (4x50 menit)

C. PETUNJUK

1. Awali setiap aktivitas dengan do'a yang khusuk, semoga berkah dan mendapat kemudahan.
2. Lakukan praktikum dengan cara;
   a) Mengamati tujuan, dasar teori, dan latihan-latihan praktikum dengan baik dan benar.
   b) Mencoba mengerjakan tugas-tugas sesuai dengan perintah.
   c) Membuat hasil laporan praktikum

D. DASAR TEORI

1. Linked List

Secara umum search dapat diartikan mencari data dengan cara menelusuri tempat penyimpanan data tersebut. Tempat penyimpanan data dalam memory dapat berupa array atau dapat juga dalam bentuk Linked List.

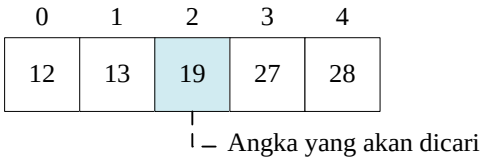
Pencarian dapat dilakukan terhadap data yang secara keseluruhan berada dalam memory komputer ataupun terhadap data yang berada dalam penyimpanan eksternal (*hard disk*).

1.1 Sequential Search

Teknik pencarian data dari array yang paling mudah adalah dengan cara sequential search, dimana data dalam array dibaca 1 demi satu, diurutkan dari index terkecil ke index terbesar, maupun sebaliknya.

Array

int A[5] = {12, 13, 19, 27, 28}



Gambar 1.1 Data Pencarian Sekuensial

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 dicocokan dengan angka yang akan dicari, jika tidak sama makapencarian 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 Binary Search

Metode pencarian yang kedua adalah binary search, pada metode pencarian ini, data harus diurutkan terlebih dahulu. Pada metode pencarian ini, data dibagi menjadi dua bagian (secara logika), untuk setiap tahap pencarian.

Algoritma binary search :

a) Data diambil dari posisi 1 sampai posisi akhir N
b) Kemudian cari posisi data tengah dengan rumus: (posisi awal + posisi akhir) / 2
c) Kemudian data yang dicari dibandingkan dengan data yang di tengah, apakah sama atau lebih kecil, atau lebih besar?
d) Jika lebih besar, maka proses pencarian dicari dengan posisi awal adalah posisi tengah+1
e) Jika lebih kecil, maka proses pencarian dicari dengan posisi akhir adalah posisi tengah–1
f) Jika data sama, berarti ketemu.

1.3 Fibonacci Search

Fibonacci Search adalah pencarian sebuah elemen dalam array satu dimensi dengan menggunakan angka fibonacci sebagai titik-titik (indeks) elemen array yang isinya dibandingkan dengan nilai yang dicari.Sama halnya dengan Binary Search, Fibonacci Search juga mengharuskan data yang sudah terurut baik menaik (*ascending*) maupun menurun (*descending*).

1.4 Interpolation Search

Interpolation Search adalah pencarian sebuah elemen dalam array satu dimensi dengan metode interpolasi atau perkiraan secara interpolasi, dimana data harus diurutkan terlebih dahulu.

a) Jika data[posisi] > data yg dicari, high = pos – 1

b) Jika data[posisi] < data yg dicari, low = pos + 1

E. LATIHAN

1. Percobaan Program Sequential Search: mencoba, membuat, menampilkan sebuah program *Searching.*

   Source Code :

```c
#include <stdio.h>

int cari (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;
}
```

```
26
27  int main ()
28  {
29      int data[5]= {12, 13, 19, 27, 28};
30      int dicari = 19;
31
32      printf("\tMetode Sequentian Search\n\n");
33      printf("Data: 12, 13, 19, 27, 28\n\n");
34      printf("Posisi %d berada pada index ke-: %d \n ", dicari, cari(data, 5, dicari));
35
36      return 0;
37
38  }
```

Tampilan :

```
        Metode Sequentian Search
Data: 12, 13, 19, 27, 28

Posisi 19 berada pada index ke-: 2

--------------------------------
```

2. Percobaan Program *Binary Search*: mencoba, membuat, menampilkan sebuah program *Binary Searching.*

Source Code :

```
1   #include <stdio.h>
2   #include <string.h>
3
4   int poskar (char st[], char m)
5   {
6       int i, posisi, panjang;
7
8       i =0;
9       posisi = -1;
10      panjang = strlen(st);
11      while((i < panjang-1) && posisi == -1)
12      {
13          if(st[i] == m)
14              posisi = i;
15              i++;
16      }
17      return posisi;
18  }
```

```
19
20  int main()
21  {
22      printf("\t Metode Binary Search\n\n");
23
24      char kalimat[] = "Teknik Informatika";
25      char dicari = 'm';
26
27          printf("\nPosisi %c dalam string %s berada pada index ke- [%d] ", dicari, kalimat, poskar(kalimat, dicari));
28
29      dicari = 'n';
30          printf("\nPosisi %c dalam string %s berada pada index ke- [%d]", dicari, kalimat, poskar(kalimat, dicari));
31
32      return 0;
33  }
```

Tampilan :

```
         Metode Binary Search


Posisi m dalam string Teknik Informatika berada pada index ke- [12]
Posisi n dalam string Teknik Informatika berada pada index ke- [3]
--------------------------------
```

3. Percobaan Program *Fibonacci Search*: mencoba, membuat, menampilkan sebuah program *Fibonacci Searching.*

   Source Code :

binarysearch.cpp | [*] FibonacciSearch.cpp

```
1    #include <stdio.h>
2    #include <conio.h>
3    #include<iostream>
4    using namespace std;
5
6    int main()
7    {
8        int i, j, F0, F1, Fibo, n, m, N, Flag, x;
9        int FK, FK1, FK2, FK3, s, p, q, t;
10       int A[10] = {8, 15, 21, 28, 31, 37, 39, 46, 48, 50};
11       int FIBO[8];
12       cout<<"\tMetode Fibonacci Search\n\n";
13       cout<<"Data : ";
14       for(x=0; x<10; x++)
15       {
16           cout<<A[x]<<" ";
17           n = 9;
18           F0 = 1; F1 = 1; Fibo = 1;
19           j =1;
20       }
21
22       while (Fibo<= n+1)
23       {
24           FIBO[j] = Fibo;
25           Fibo = F0+F1; F0=F1; F1=Fibo;
26           j++;
27       }
28
29       s = j - 1;
30       FK = FIBO[s];
31       FK1 = FIBO[s-1]; i = FK1;
32       FK2 = FIBO[s-2]; p = FK2;
33       FK3 = FIBO[s-3]; q = FK3;
34
```

```
35          m = (n+1) - FK;
36          printf("\n\nMasukan data yang ingin dicari : ");
37          scanf("%d", &N);
38
39          if(N > A[i]) i = i+m;
40              Flag = 0;
41              while(i != 0 && Flag ==0)
42              {
43              if(N == A[i]) Flag = 1;
44
45              else if(N < A[i])
46              {
47                  if(q == 0) i = 0;
48                  else
49                  {
50                      i = i - q;
51                      t = p;
52                      p = q;
53                      q = t - q;
54                  }
55              }
56              else
57              {
58                  if(p == 1)
59                      i = 0;
60                  else
61                  {
62                      i = i + q;
63                      p = p - q;
64                      q = q - p;
65                  }
66              }
67          }
68          if(Flag == 1)
69              printf("\nData ditemukan");
70          else
71              printf("\nData tidak ditemukan");
72      }
```

Tampilan :

```
        Metode Fibonacci Search
Data : 8 15 21 28 31 37 39 46 48 50
Masukan data yang ingin dicari : 40
Data tidak ditemukan
--------------------------------
```

```
        Metode Fibonacci Search
Data : 8 15 21 28 31 37 39 46 48 50
Masukan data yang ingin dicari : 50
Data ditemukan
--------------------------------
```

4. Percobaan Program *Interpolation Search*: mencoba, membuat, menampilkan sebuah program *Interpolation Searching.*

Source Code :

```
#include <iostream>
#include <math.h>
using namespace std;

int main()
{
    int data[10] = {9, 17, 24, 25, 36, 49, 52, 54, 59, 60};
    int low,high,cari,posisi;
    float posisi1;
    int N = 10,tanda=0;
    low=0,high=N-1;

    cout<<"\tMetode Interpolation Search\n\n";
    cout<<"Data : 9, 17, 24, 25, 36, 49, 52, 54, 59, 60\n";
    cout<<"\nMasukan data yang dicari : ";
    cin>>cari;
    do
    {
        posisi1 = (cari-data[low])/(data[high]-data[low])*(high-low)+low;
        posisi = floor(posisi1); //pembulatan ke bawah
        if(data[posisi]==cari) {
        tanda =1;
        break;
    }

    if(data[posisi]>cari)
        high=posisi-1;
    else if(data[posisi]<cari)
        low=posisi+1;
    }

    while (cari>=data[low]&&cari<=data[high]);
    {
        if(tanda==1)
            cout<<"\nData ditemukan\n";
        else
            cout<<"\nData tidak ada\n";
    }
}
```

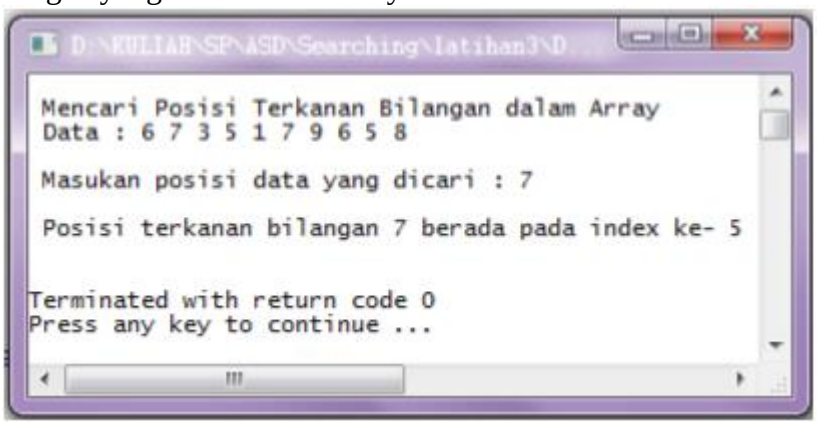
Tampilan :

```
        Metode Interpolation Search
Data : 9, 17, 24, 25, 36, 49, 52, 54, 59, 60
Masukan data yang dicari : 25
Data ditemukan
--------------------------------
```

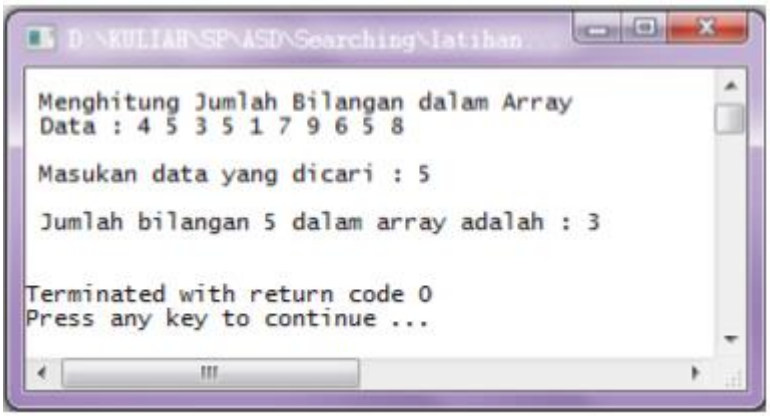
```
        Metode Interpolation Search
Data : 9, 17, 24, 25, 36, 49, 52, 54, 59, 60
Masukan data yang dicari : 17
Data ditemukan
--------------------------------
```

F. TUGAS PRAKTIKUM

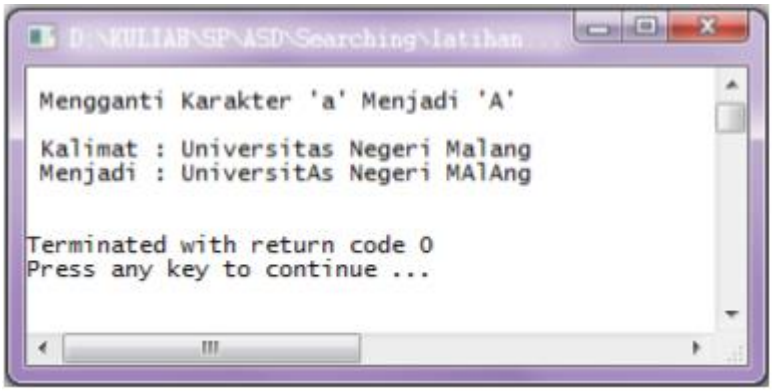
1. Buatlah sebuah program untuk melakukan pencarian data berupa posisi terkanan dari suatu bilangan yang dicari dalam array satu dimensi.



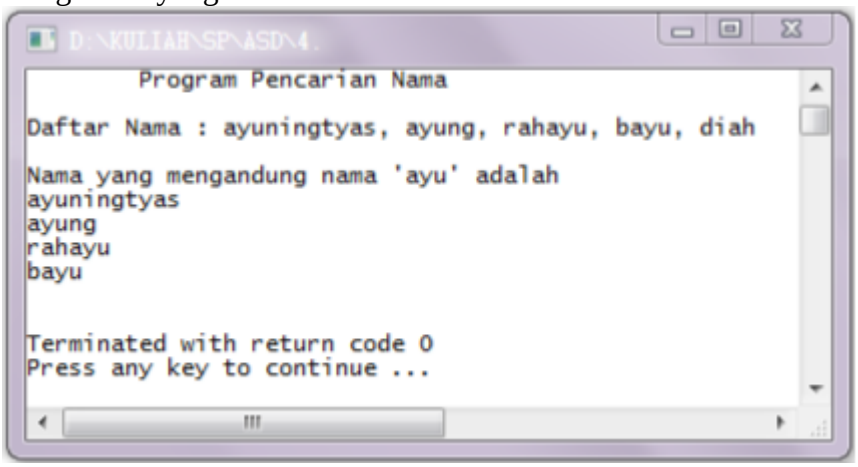
2. Buatlah sebuah program untuk menghitung jumlah suatu bilangan dalam sebuah array satu dimensi yang berisi n buah elemen.



3. Buatlah sebuah program yang dapat melakukan pencarian karakter dalam kalimat string, kemudian hasil pencarian karakter tersebut diubah menjadi karakter lain dan ditampilkan.



4. Buatlah sebuah program untuk melakukan pencarian nama dari beberapa daftar nama dalam array yang disediakan, kemudian tampilkan nama-nama yang didalamnya mengandung nama yang dicari



G. TUGAS RUMAH

Ada sebuah Supermarket memiliki sebuah gudang, Supermaket tersebut menginginkan untuk membuat sebuah aplikasi yang dapat mendata data-data barang yang ada di dalam gudang tersebut yang terdiri dari.

1. Kode Barang
2. Nama Barang
3. Harga
4. Stok (Jumlah Barang)

Buatlah aplikasi yang dapat memasukkan informasi barang-barang yang ada di gudang tersebut. Selain dapat menginputkan data, Aplikasi juga mempunya kemampuan untuk

1. Menambahkan/Mengurangi STOK barang yang sudah ada
2. Merubah Harga
3. Dan Menghapus Data

(Gunakan Prinsip Searching untuk melakukan ketiga perintah diatas)