

# **Laporan Praktikum 7**

## **Struktur Data Algoritma**



### **Materi**

**"Binary Search, Array 2 Dimensi, Linked List"**

**Nama :**

**Muhammad Azhar Rasyad**

**NIM :**

**0110217029**

**Program Studi :**

**Teknik Informatika 1**



## 2. Penjelasan Program Binary Search :

### 2.1 Bagian Kepala Program

```
1 // C program to implement recursive Binary Search
2 #include <stdio.h>
3
```

**#include <stdio.h>** digunakan untuk memanggil library yang berisi perintah-perintah untuk menjalankan bahasa pemrograman C.

**#include** digunakan untuk memanggil file yang berisi library bahasa pemrograman C.

**<stdio.h>** digunakan sebagai library bahasa pemrograman C yang berisi perintah-perintah untuk menjalankan bahasa pemrograman C.

### 2.2 Fungsi Rekursif Binary Search

```
4 // A recursive binary search function. It returns
5 // location of x in given array arr[l..r] is present,
6 // otherwise -1
7 int binarySearch(int arr[], int l, int r, int x)
8 {
9     if (r >= l)
10    {
11        int mid = l + (r - l)/2;
12
13        // If the element is present at the middle
14        // itself
15        if (arr[mid] == x)
16            return mid;
17
18        // If element is smaller than mid, then
19        // it can only be present in left subarray
20        if (arr[mid] > x)
21            return binarySearch(arr, l, mid-1, x);
22
23        // Else the element can only be present
24        // in right subarray
25        return binarySearch(arr, mid+1, r, x);
26    }
27
28    // We reach here when element is not
29    // present in array
30    return -1;
31 }
```

Berikut masing-masing penjelasan dari fungsi binary search, sebagai berikut :

```
7 int binarySearch(int arr[], int l, int r, int x)
```

`int binarySearch(int arr[], int l, int r, int x)` merupakan fungsi rekursif binary search dengan type data `int` dengan nama fungsi `binarySearch` dan terdapat variabel-variabel untuk dimasukkan kedalam fungsi yaitu `(int arr[], int l, int r, int x)`.

`int` pada fungsi digunakan supaya mengembalikan nilai dari hasil rekursif binary search.

`binarySearch` digunakan sebagai nama dari fungsi rekursif binary search.

`(int arr[], int l, int r, int x)` digunakan untuk mendeklarasikan nilai dari variabel-variabel yang diterima dari program utama, berikut masing-masing penjelasannya :

- `int arr[]` menyimpan nilai seluruh data dari kumpulan data.
- `int l` menyimpan nilai awal / pertama atau index ke 0.
- `int r` menyimpan nilai ujung / terakhir atau index ke n-1.
- `int x` menyimpan nilai data yang sedang dicari.

```
9     if (r >= l)
```

`if (r >= l)` berfungsi untuk membandingkan apakah index terakhir lebih besar atau sama dengan dengan index awal, kondisi ini supaya rekursif akan terus mengulang hingga index terakhir lebih kecil dari index pertama.

```
11         int mid = l + (r - l)/2;
```

`int mid = l + (r - l)/2;` berfungsi untuk menyimpan nilai tengah dari hasil banyak data dibagi 2, nantinya nilai tengah akan menjadi kunci utama untuk mencari data yang dicari karena metode Binary Search membandingkan nilai yang dicari dengan nilai tengah.

```
15             if (arr[mid] == x)
16                 return mid;
```

`if (arr[mid] == x) return mid;` berfungsi membandingkan apakah isi nilai data tengah sama dengan data yang dicari atau tidak, Jika Ya

maka hasil yang diberikan adalah index nilai tengah dan proses rekursif berhenti. Jika Tidak maka lanjut ke proses dibawah ini :

```
20         if (arr[mid] > x)
21             return binarySearch(arr, l, mid-1, x);
```

**if (arr[mid] > x) return binarySearch(arr, l, mid-1, x);** berfungsi membandingkan nilai data tengah apakah lebih besar dari data yang dicari atau tidak, Jika Ya maka nilai data tengah akan bergeser ke kiri satu index dan disimpan ke dalam variabel r kemudian melanjutkan proses rekursif binary search seperti diawal. Jika Tidak maka lanjut ke proses dibawah ini :

```
25         return binarySearch(arr, mid+1, r, x);
```

**return binarySearch(arr, mid+1, r, x);** berfungsi menggeser nilai tengah ke kanan satu index dan disimpan ke dalam variabel l kemudian melanjutkan proses rekursif binary search seperti awal.

```
30     return -1;
```

**return -1;** Jika data yang dicari tidak ada dalam data tersebut maka akan dikembalikan dengan nilai -1 kemudian pada program utama akan memberikan notifikasi bahwa data yang dicari tidak ada.

## 2.3 Program Utama

```
33 int main(void)
34 {
35     int arr[] = {2, 3, 4, 10, 40};
36     int n = sizeof(arr)/ sizeof(arr[0]);
37     int x = 10;
38     int result = binarySearch(arr, 0, n-1, x);
39     (result == -1)? printf("Element is not present in array")
40                   : printf("Element is present at index %d",
41                           result);
42     return 0;
43 }
```

Berikut masing-masing penjelasan dari program utama, sebagai berikut :

```
33 int main(void)
```

`int main (void)` berfungsi sebagai kepala program utama supaya program yang ada di dalam `int main` merupakan program utama dan variabelnya bersifat lokal bukan global.

```
35  int arr[] = {2, 3, 4, 10, 40};
36  int n = sizeof(arr)/ sizeof(arr[0]);
37  int x = 10;
```

`int arr[] = {2, 3, 4, 10, 40};` digunakan untuk mendeklarasikan isi data variabel `arr[]` atau menyimpan nilai-nilai yaitu 2, 3, 4, 10, dan 40.

`int n = sizeof(arr)/sizeof(arr[0]);` berfungsi mendeklarasikan isi data variabel `n` atau menyimpan banyaknya data yang ada pada variabel `arr`.

`int x = 10;` digunakan untuk mendeklarasikan isi data variabel `x` atau untuk menyimpan nilai yang ingin dicari.

```
38  int result = binarySearch(arr, 0, n-1, x);
39  (result == -1)? printf("Element is not present in array")
40                  : printf("Element is present at index %d",
41                          result);
```

`int result = binarySearch(arr, 0, n-1, x);` berfungsi untuk menyimpan hasil dalam variabel `result` dengan melalui proses fungsi rekursif `binarySearch` dan memasukkan variabel `arr`, `n`, dan `x` yang telah dideklarasikan sebelumnya.

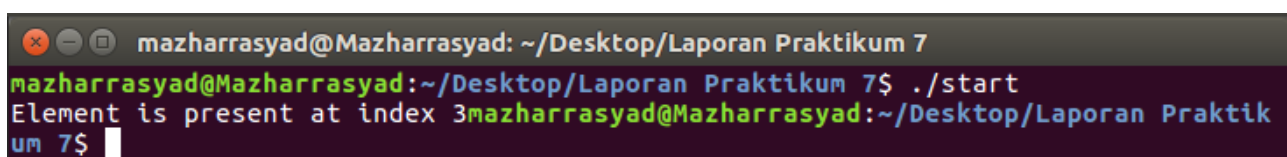
```
(result == -1)? printf("Element is not present in array")
               : printf("Element is present at index %d",result);
```

berfungsi untuk menampilkan hasil yang didapat dari proses fungsi `binarySearch` sebelumnya apakah data yang dicari ketemu atau tidak, Jika Tidak maka akan menampilkan notifikasi tidak ketemu, Jika Ya maka akan menampilkan notifikasi ketemu beserta data yang dicari ada pada index ke berapa.

```
42  return 0;
```

`return 0;` berfungsi sebagai berhentinya sebuah program.

### 3. Hasil Kompilasi :



```
mzharrasyad@mzharrasyad: ~/Desktop/Laporan Praktikum 7
mzharrasyad@mzharrasyad:~/Desktop/Laporan Praktikum 7$ ./start
Element is present at index 3mzharrasyad@mzharrasyad:~/Desktop/Laporan Praktik
um 7$
```

## B. Array 2 Dimensi

### 1. Source Code Array 2 Dimensi :

```
1#include <iostream>
2
3using namespace std;
4
5int main()
6{
7    int a[3][3] =
8    {
9        {1,0,3},
10       {4,0,3},
11       {0,2,0}
12    };
13
14    cout << "|   || A || B || C |\n";
15
16    for (int i = 0; i < 3; i++)
17    {
18        if (i == 0)
19            cout << "| A |";
20        else if (i == 1)
21            cout << "| B |";
22        else if (i == 2)
23            cout << "| C |";
24
25        for (int j = 0; j < 3; j++)
26        {
27            cout << "| " << a[i][j] << " |";
28        }
29
30        cout << endl;
31    }
32
33    cout << endl;
34 }
```

### 2. Penjelasan Program Array 2 Dimensi :

#### 2.1 Bagian Kepala Program

```
1#include <iostream>
2
3using namespace std;
4
5int main()
```

`#include <iostream>` digunakan untuk memanggil library yang berisi perintah-perintah untuk menjalankan bahasa pemrograman C++.

**#include** digunakan untuk memanggil file yang berisi library bahasa pemrograman C++.

**<iostream>** digunakan sebagai library bahasa pemrograman C++ yang berisi perintah-perintah untuk input output dan lainnya.

**using namespace std;** digunakan sebagai bentuk standar dari bahasa pemrograman C++.

**int main ()** merupakan tempat program utama yang akan diproses.

## 2.1 Isi Program

```
7      int a[3][3] =
8      {
9          {1,0,3},
10         {4,0,3},
11         {0,2,0}
12     };
```

**int a[3][3]** berfungsi mendeklarasikan variabel array yang menyimpan nilai **{1,0,3},{4,0,3},{0,2,0}** kedalam array 2 dimensi.

```
14     cout << "|   || A || B || C |\n";
15
16     for (int i = 0; i < 3; i++)
17     {
18         if (i == 0)
19             cout << "| A |";
20         else if (i == 1)
21             cout << "| B |";
22         else if (i == 2)
23             cout << "| C |";
24
25         for (int j = 0; j < 3; j++)
26         {
27             cout << "| " << a[i][j] << " |";
28         }
29         cout << endl;
30     }
31
32     cout << endl;
```

Program diatas berfungsi untuk menampilkan seluruh isi dari array a sebagai sebuah matriks, untuk lebih jelasnya dapat dilihat pada hasil dari kompilasi program diatas.



### 3. Hasil Kompilasi :

```
mazharrasyad@Mazharrasyad: ~/Desktop/Laporan Praktikum 7
mazharrasyad@Mazharrasyad:~/Desktop/Laporan Praktikum 7$ ./start
|  |  | A |  | B |  | C |
| A |  | 1 |  | 0 |  | 3 |
| B |  | 4 |  | 0 |  | 3 |
| C |  | 0 |  | 2 |  | 0 |

mazharrasyad@Mazharrasyad:~/Desktop/Laporan Praktikum 7$
```

## C. Linked List

### 1. Source Code Linked List :

---

```
1 #include <iostream>
2 using namespace std;
3 |
4 struct node
5 {
6     int data;
7     node* next;
8 };
9
10 int main()
11 {
12     node* n;
13     node* head;
14     node* temp;
15     node* sekarang = head;
16
17     int count = 0;
18
19     n = new node;
20     n -> data = 1;
21     temp = n;
22     head = n;
23
24     n = new node;
25     n -> data = 2;
26     temp -> next = n;
27
28     n = new node;
29     n -> data = 3;
30     temp -> next = NULL;
31
32     while (sekarang != NULL)
33     {
34         count++;
35         sekarang = sekarang -> next;
36         cout << sekarang << " -> ";
37     }
38 }
```

## 2. Penjelasan Program Linked List :

### 2.1 Bagian Kepala Program

```
1 #include <iostream>
2 using namespace std;
```

**#include <iostream>** digunakan untuk memanggil library yang berisi perintah-perintah untuk menjalankan bahasa pemrograman C++.

**#include** digunakan untuk memanggil file yang berisi library bahasa pemrograman C++.

**<iostream>** digunakan sebagai library bahasa pemrograman C++ yang berisi perintah-perintah untuk input output dan lainnya.

**using namespace std;** digunakan sebagai bentuk standar dari bahasa pemrograman C++.

### 2.2 Isi Program

```
4 struct node
5 {
6     int data;
7     node* next;
8 };
```

**struct node** digunakan sebagai bentuk standar dari bahasa pemrograman C++.

**struct** digunakan karena linked list memiliki field yang berbeda type data.

**node** merupakan nama dari setiap kumpulan data yang akan disambung-sambungkan.

```

10 int main()
11 {
12     node* n;
13     node* head;
14     node* temp;
15     node* sekarang = head;
16
17     int count = 0;
18
19     n = new node;
20     n -> data = 1;
21     temp = n;
22     head = n;
23
24     n = new node;
25     n -> data = 2;
26     temp -> next = n;
27
28     n = new node;
29     n -> data = 3;
30     temp -> next = NULL;
31
32     while (sekarang != NULL)
33     {
34         count++;
35         sekarang = sekarang -> next;
36         cout << sekarang << " -> ";
37     }
38 }

```

---

Berikut masing-masing penjelasan dari program utama, sebagai berikut :

```

12     node* n;
13     node* head;
14     node* temp;
15     node* sekarang = head;

```

Gambar diatas merupakan deklarasi dari linked list, sebagai berikut :

- Variabel n untuk menyimpan alamat dari masing-masing node.
- Variabel head sebagai alamat node yang pertama.
- Variabel temp untuk menyimpan alamat dari alamat sebelumnya atau setelahnya.

```

17         int count = 0;
18
19         n = new node;
20         n -> data = 1;
21         temp = n;
22         head = n;
23
24         n = new node;
25         n -> data = 2;
26         temp -> next = n;
27
28         n = new node;
29         n -> data = 3;
30         temp -> next = NULL;

```

Gambar diatas merupakan cara membuat sebuah node baru dimulai dengan perintah new kemudian untuk mengakses setiap node menggunakan perintah -> dari head hingga tail atau NULL.

```

32         while (sekarang != NULL)
33         {
34             count++;
35             sekarang = sekarang -> next;
36             cout << sekarang << " -> ";
37         }

```

Gambar diatas merupakan proses untuk menampilkan setiap node-node yang telah dibuat dengan variabel count untuk menghitung dan variabel sekarang merupakan tempat node berada hingga node berada pada node terakhir.

## Referensi

- <https://google.com>
- <https://www.geeksforgeeks.org/binary-search/>