

# **Laporan Hasil Ujian Tengah Semester (UTS) Pemikiran Komputasional**

## **“Sistem Manajemen Pengelolaan Data Mahasiswa Berbasis CLI”**



Disusun Oleh:

1. Muhammad Refdi Pasaribu / 231110267
2. Muhammad Siddik Syahputra / 231111876
3. Ahmad Yuda Zaki Yamani / 231112933
4. Nurihsan Febrianto / 231111759

**TEKNIK INFORMATIKA  
FAKULTAS INFORMATIKA  
UNIVERSITAS MIKROSKIL  
2024**

# 1. Daftar Isi

1. Daftar Isi .....	2
2. Algoritma Sistem Aplikasi CRUD Data Mahasiswa Menggunakan Bahasa Pemrograman Python .....	3
3. Kompleksitas Program .....	5
3.1 Tambah Data Mahasiswa $O(1)$ – Konstan .....	5
3.2 Tampilkan Data Mahasiswa $O(1)$ - Konstan .....	5
3.3 Ubah Data Mahasiswa $O(1)$ – Konstan .....	6
3.4 Pengurutan Data Mahasiswa $O(n^2)$ .....	6
3.5 Pencarian Data Mahasiswa $O(\log n)$ .....	7
4. Alur Sistem Aplikasi CRUD Data Mahasiswa Menggunakan Bahasa Pemograman Pyhton .....	9
4.1 Main menu .....	9
4.2 First menu (Create Student Data) .....	9
4.3 Second menu(Read Student Data) .....	9
4.4 Third menu(Update Student Data) .....	10
4.5 Fourth menu(Sorting Student Data by Field) .....	11
4.6 Fifth menu(Search Student Data by NIM, Name) .....	12
4.7 Last menu(Exit) .....	14
5. Pembagian Tugas Anggota .....	15

## 2. Algoritma Sistem Aplikasi CRUD Data Mahasiswa Menggunakan Bahasa Pemrograman Python

1. Mulai
2. Masukkan perintah yang ingin dijalankan 1-6 : int
  - a. Tambah Data Mahasiswa (1)
    - i. Masukkan *`NIM`* Mahasiswa
    - ii. Masukkan *`Nama`* Mahasiswa
    - iii. Masukkan *`Tempat Lahir`* Mahasiswa
    - iv. Masukkan *`Tanggal Lahir`* Mahasiswa
    - v. Masukkan *`Program Studi`* Mahasiswa
    - vi. Masukkan *`Tahun Masuk`* Mahasiswa
  - b. Tampilkan Data Mahasiswa (2)
    - i. Semua data mahasiswa akan ditampilkan dalam bentuk tabel dengan format {nim, nama, tempat\_tanggal\_lahir, program\_studi, tahun\_masuk}
  - c. Ubah Data Mahasiswa (3)
    - i. Masukkan *`NIM`* Mahasiswa yang ingin di ubah datanya
    - ii. Data mahasiswa akan di tampilkan dalam bentuk tabel dengan format {field, detail}
    - iii. Masukkan *`Nama`* mahasiswa jika ingin mengubahnya
    - iv. Masukkan *`Tempat Lahir`* mahasiswa jika ingin mengubahnya
    - v. Masukkan *`Tanggal Lahir`* mahasiswa jika ingin mengubahnya
    - vi. Masukkan *`Program Studi`* mahasiswa jika ingin mengubahnya
    - vii. Masukkan *`Tahun Masuk`* mahasiswa jika ingin mengubahnya
  - d. Urutkan Data Mahasiswa (4)
    - i. Masukkan pilihan urutan berdasarkan field (1-4) : int
      1. *`NIM`*
      2. *`Nama`*
      3. *`Program Studi`*
      4. *`Tahun Masuk`*
    - ii. Setelah memilih salah satu ke-empat menu tersebut data akan di tampilkan secara (ascending) sesuai field yang di pilih, format untuk tabel yang akan ditampilkan {nim, nama, tempat\_tanggal\_lahir, program\_studi, tahun\_masuk}
  - e. Cari Data Mahasiswa (5)
    - i. Masukkan pilihan pencarian berdasarkan (1-2) : int
      1. *`NIM`*
        - a. Masukkan *`NIM`* Mahasiswa yang ingin dicari
        - b. Jika *`NIM`* dari mahasiswa ketemu maka
          - i. Tampilkan dalam bentuk tabel dengan format {nim, nama, tempat\_tanggal\_lahir, program\_studi, tahun\_masuk}

- c. Jika tidak maka
    - i. Tampilkan pesan kesalahan “*Data tidak ditemukan.*”
- 2. `Nama`
  - a. Masukkan `Nama` Mahasiswa yang ingin dicari
  - b. Jika `Nama` dari mahasiswa ketemu maka
    - i. Tampilkan dalam bentuk tabel dengan format  
{nim, nama, tempat\_tanggal\_lahir,  
program\_studi, tahun\_masuk}
  - c. Jika tidak maka
    - i. Tampilkan pesan kesalahan “*Data tidak ditemukan.*”
- f. Keluar (6)
    - i. Program akan dihentikan
- 3. Selesai

## 3. Kompleksitas Program

### 3.1 Tambah Data Mahasiswa $O(1)$ – Konstan

A screenshot of a code editor with a dark background and light-colored text. The code is a Python function named `add_data` that takes `new_data` as an argument and returns `None`. It uses `os.path` to find the script directory and a relative path `'data.json'` to open a file. The file is loaded with `json.load`. A check is performed to see if a student with the same NIM already exists; if so, a `ValueError` is raised. The new data is appended to the `'colleger'` list in the JSON data. The file is then truncated, seeking to the beginning, and the updated data is dumped back to the file with an indent of 4 spaces.

```
1 def add_data(new_data) -> None:
2     script_dir = os.path.dirname(__file__)
3     rel_path = 'data.json'
4
5     with open(os.path.join(script_dir, rel_path)) as file:
6         data = json.load(file)
7
8     if len(search_data(Column.NIM, new_data["nim"])) > 0:
9         raise ValueError("Data NIM telah ada, gunakan NIM yang lain")
10
11     data["colleger"].append(new_data)
12
13     with open(os.path.join(script_dir, rel_path), 'r+') as file:
14         file.seek(0)
15         json.dump(data, file, indent=4)
16
17     with open(os.path.join(script_dir, rel_path)) as file:
18         data = json.load(file)
19
20     data["colleger"] = sort_data(Column.YEAR, quick_sort.Sort.DSC)
21
22     with open(os.path.join(script_dir, rel_path), 'r+') as file:
23         file.truncate(0)
24         file.seek(0)
25         json.dump(data, file, indent=4)
```

Gambar 3.1

Kompleksitasnya bisa dikatakan  $O(1)$ , karena jumlah operasi yang dilakukan tidak bergantung pada jumlah masukan yang sudah ada dalam aplikasi. Jadi, tidak peduli seberapa banyak data yang telah ada, proses penambahan data baru ini akan memiliki kompleksitas yang relatif konstan.

### 3.2 Tampilkan Data Mahasiswa $O(1)$ - Konstan

A screenshot of a code editor with a dark background and light-colored text. The code is a Python function named `get_list_data` that returns a `set`. It uses `os.path` to find the script directory and a relative path `'data.json'` to open a file. The data is loaded with `json.load` from the `'colleger'` list. The file is then closed, and the data is returned.

```
1 def get_list_data() -> set:
2     script_dir = os.path.dirname(__file__)
3     rel_path = 'data.json'
4
5     f = open(os.path.join(script_dir, rel_path))
6     data = json.load(f)["colleger"]
7
8     f.close()
9     return data
```

Gambar 3.2

Kompleksitas ini juga bisa dikatakan  $O(1)$ , karena jumlah operasi hanya dilakukan satu kali.

### 3.3 Ubah Data Mahasiswa $O(1)$ – Konstan



```
1 def update_data(index, value):
2     script_dir = os.path.dirname(__file__)
3     rel_path = 'data.json'
4
5     data = sort_data(Column.NIM)
6     data[index] = value
7
8     with open(os.path.join(script_dir, rel_path)) as file:
9         data_source = json.load(file)
10
11     data_source["colleger"] = data
12
13
14     with open(os.path.join(script_dir, rel_path), 'r+') as file:
15         file.truncate(0)
16         file.seek(0)
17         json.dump(data_source, file, indent=4)
```

Gambar 3.3

Kompleksitasnya bisa dikatakan  $O(1)$ , karena jumlah operasi yang dilakukan tidak bergantung pada jumlah masukan yang sudah ada dalam aplikasi. Jadi, tidak peduli seberapa banyak data yang telah ada, proses perubahan data ini akan memiliki kompleksitas yang relatif konstan.

### 3.4 Pengurutan Data Mahasiswa $O(n^2)$

```

1  from enum import Enum
2
3
4  class Sort(Enum):
5      ASC = 'asc'
6      DSC = 'dsc'
7
8
9  def quicksort(arr: list, key: str, sort: Sort = Sort.ASC) -> list:
10     if len(arr) <= 1:
11         return arr
12     else:
13         pivot = arr[0][key]
14
15         if sort == Sort.ASC:
16             less_than_pivot = [x for x in arr[1:] if x[key] <= pivot]
17             greater_than_pivot = [x for x in arr[1:] if x[key] > pivot]
18
19         if sort == Sort.DSC:
20             less_than_pivot = [x for x in arr[1:] if x[key] >= pivot]
21             greater_than_pivot = [x for x in arr[1:] if x[key] < pivot]
22
23         return quicksort(less_than_pivot, key) + [arr[0]] + quicksort(greater_than_pivot, key)
24
25
26 def sort_data(column: Column, sort: quick_sort.Sort = quick_sort.Sort.ASC) -> list:
27     return quick_sort.quicksort(get_list_data(), column.value, sort)
28
29

```

Gambar 3.4

Kompleksitas waktu dari algoritma quicksort  $O(n^2)$ , karena pivot dipilih selalu merupakan elemen terbesar atau terkecil dalam array. Jadi, jika list sudah terurut maka algoritma ini akan melakukan iterasi dari elemen yang lebih besar.

### 3.5 Pencarian Data Mahasiswa $O(\log n)$

```

1  from enum import Enum
2
3  def binary_search(list, value, key) -> tuple:
4      start = 0
5      last = len(list) - 1
6      result_index = []
7      result = []
8
9      while start <= last:
10         middle = (start + last) // 2
11         if list[middle][key].startswith(value):
12             result_index.append(middle)
13
14         # check match entry in left
15         i = middle - 1
16         while i >= 0 and list[i][key].startswith(value):
17             result_index.append(i)
18             i -= 1
19
20         # check match entry in right
21         i = middle + 1
22         while i < len(list) and list[i][key].startswith(value):
23             result_index.append(i)
24             i += 1
25
26         break
27     elif list[middle][key] < value:
28         start = middle + 1
29     else:
30         last = middle - 1
31
32     for i in result_index:
33         result.append(list[i])
34
35     return (result_index, result)
36
37
38 def search_data(column: Column, value: str, search_by: SearchBy = SearchBy.VALUE):
39     sort = sort_data(column)
40     result = binary_search.binary_search(sort, value, column.value)
41
42     match search_by:
43         case SearchBy.VALUE:
44             return result[1]
45         case SearchBy.INDEX:
46             return result[0]
47         case SearchBy.INDEX_VALUE:
48             return result
49

```

Gambar 3.5

Kompleksitas dari Binary searchnya adalah  $O(\log n)$ , namun di fungsi `sort\_data` adalah  $O(n^2)$ , karena Binary search lebih dominan digunakan makanya kompleksitas pada fungsi `search\_data` ini adalah  $O(\log n)$

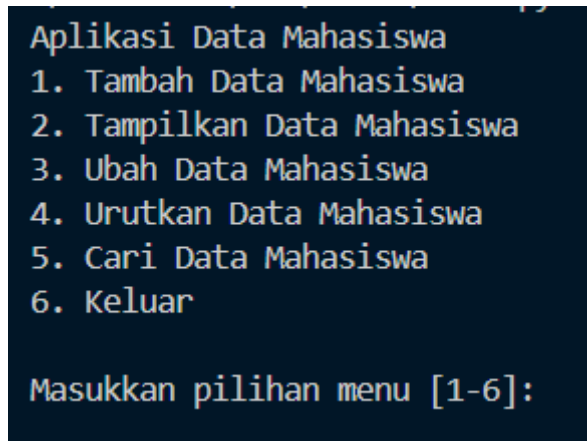


## 4. Alur Sistem Aplikasi CRUD Data Mahasiswa Menggunakan Bahasa Pemograman Pyhton

### 4.1 Main menu

#### Description:

When run, the program immediatly presents an interactive menu that makes it easy for users to use this data CRUD application.



```
Aplikasi Data Mahasiswa
1. Tambah Data Mahasiswa
2. Tampilkan Data Mahasiswa
3. Ubah Data Mahasiswa
4. Urutkan Data Mahasiswa
5. Cari Data Mahasiswa
6. Keluar

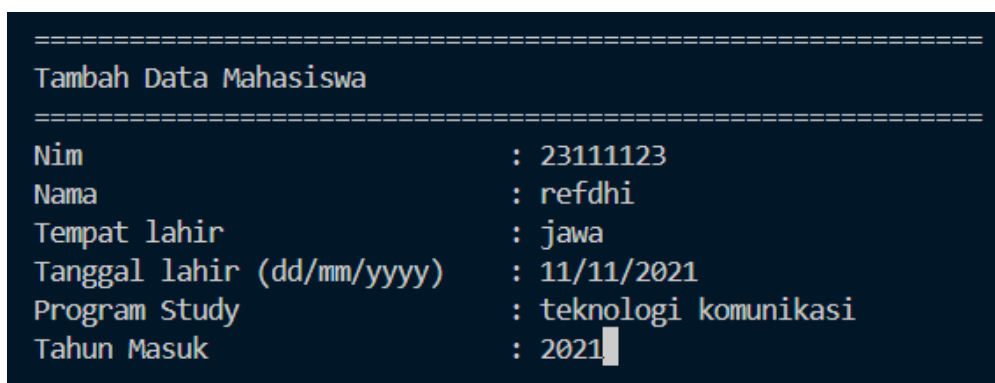
Masukkan pilihan menu [1-6]:
```

Gambar 4.1

### 4.2 First menu (Create Student Data)

#### Description:

In this first menu, users can enter the desired student data in a structured manner.



```
=====
Tambah Data Mahasiswa
=====
Nim           : 23111123
Nama          : refdhi
Tempat lahir  : jawa
Tanggal lahir (dd/mm/yyyy) : 11/11/2021
Program Study : teknologi komunikasi
Tahun Masuk   : 2021
```

Gambar 4.2

### 4.3 Second menu(Read Student Data)

#### Description:

This second menu will display the student data table that has been inputted from the first menu.

nim	name	place_date_birth	major	year
aa	ahmad zakui	binjai, 11 11 2004	teknik informatika	2024
am	ahmad adi	binjai, 11 11 2004	teknik informatika	1111
231112933	yamani	kisaran, 11 11 2004	tekik ifnformatika	2011
23111123	refdhi	jawa, 11/11/2021	teknik informatika	2021

Gambar 4.3

#### 4.4 Third menu(Update Student Data)

Description:

on this menu, allows users to change data that already exists in the table, based on name or nim.

```

Urutan Data Mahasiswa
1. NIM
2. Nama

Pilih opsi urutan [1-2]: 

```

Gambar 4.4

If selecting NIM then, the user must input the appropriate nim in the previous data and change the data as desired. Likewise, if selecting NAMA.

```

NIM mahasiswa yang akan diubah: aa

```

Field	Detail
nim	aa
name	ahmad zakui
place_date_birth	binjai, 11 11 2004
major	teknik informatika
year	2024

```

Note: Kosongkan Jika Tidak Ingin Diubah
Nama :
Tempat lahir :
Tanggal lahir (dd/mm/yyyy) :
Program Study :
Tahun Masuk : 

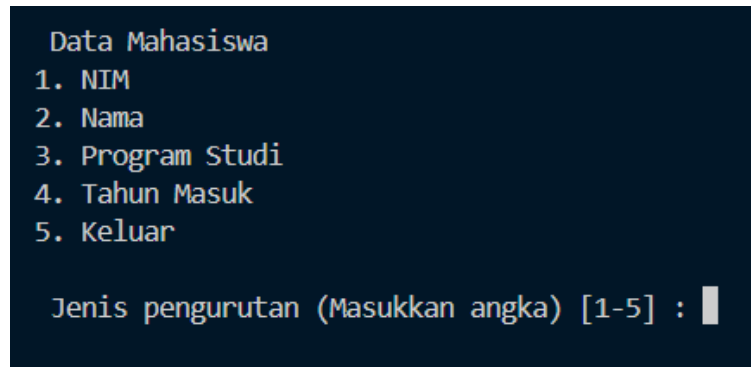
```

Gambar 4.5

## 4.5 Fourth menu(Sorting Student Data by Field)

Description:

in the fourth menu the user will be given the option to sort the data according to the options provided.



Gambar 4.6

if the user selects the NIM option, the data will be sorted by NIM.

Jenis pengurutan (Masukkan angka) [1-5] : 1

nim	name	place_date_birth	major	year
23111123	refdhi	jawa, 11/11/2021	teknik informatika	2021
231112933	yamani	kisaran, 11 11 2004	tekik ifnformatika	2011
aa	ahmad zakui	binjai, 11 11 2004	teknik informatika	2024
am	ahmad adi	binjai, 11 11 2004	teknik informatika	1111

Gambar 4.7

if the user selects the Nama option, the data will be sorted by Nama.

nim	name	place_date_birth	major	year
am	ahmad adi	binjai, 11 11 2004	teknik informatika	1111
aa	ahmad zakui	binjai, 11 11 2004	teknik informatika	2024
23111123	refdhi	jawa, 11/11/2021	teknik informatika	2021
231112933	yamani	kisaran, 11 11 2004	tekik ifnformatika	2011

Gambar 4.8

if the user selects the Program Studi option, the data will be sorted by Program Studi.

nim	name	place_date_birth	major	year
231112933	yamani	kisaran, 11 11 2004	tekik ifnformatika	2011
am	ahmad adi	binjai, 11 11 2004	teknik informatika	1111
aa	ahmad zakui	binjai, 11 11 2004	teknik informatika	2024
23111123	refdhi	jawa, 11/11/2021	teknik informatika	2021

Gambar 4.9

if the user selects the Tahun Masuk option, the data will be sorted by Tahun Masuk.

nim	name	place_date_birth	major	year
am	ahmad adi	binjai, 11 11 2004	teknik informatika	1111
231112933	yamani	kisaran, 11 11 2004	tekik ifnformatika	2011
23111123	refdhi	jawa, 11/11/2021	teknik informatika	2021
aa	ahmad zakui	binjai, 11 11 2004	teknik informatika	2024

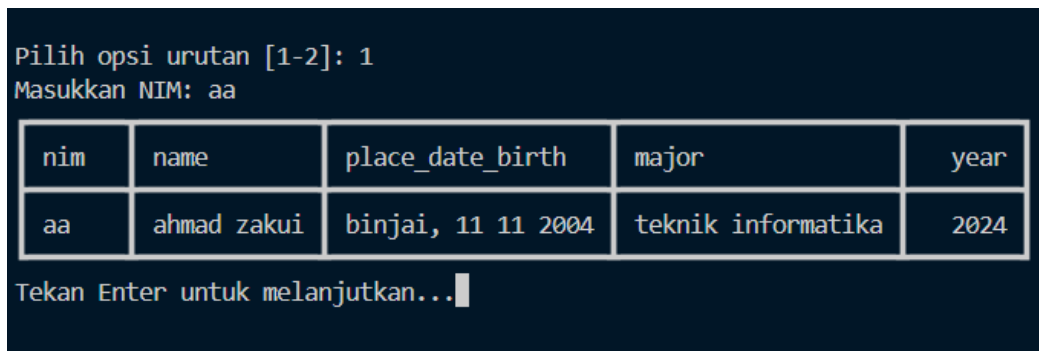
Gambar 4.10

## 4.6 Fifth menu(Search Student Data by NIM, Name)

Description:

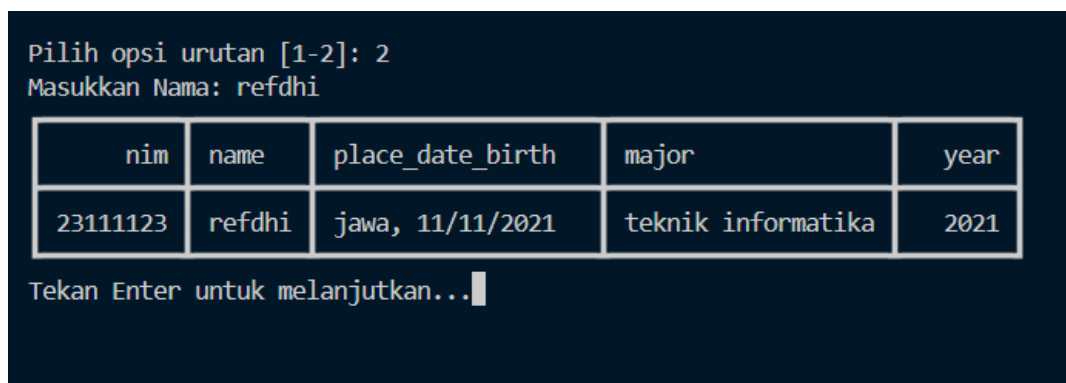
in this menu the user is allowed to search for the desired student data with the NIM and Name options.

if you choose the NIM option, then the student data search will be based on the NIM inputted by the user and if the input is correct, it will display the student data.



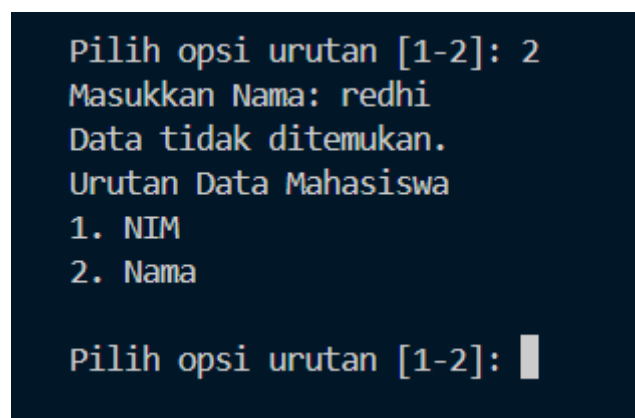
Gambar 4.11

if you choose the Nama option, then the student data search will be based on the Nama inputted by the user and if the input is correct, it will display the student data.



Gambar 4.12

if the user inputs data that does not match then no data is displayed and the user will be asked to re-input.



Gambar 4.13

## 4.7 Last menu(Exit)

to quit or exit the application

## 5. Pembagian Tugas Anggota

1. Muhammad Refdi Pasaribu
  - a. Merancang alur dan penggunaan library pada sistem aplikasi
  - b. Menentukan kompleksitas program
  - c. Membuat Laporan
  - d. Membuat Algoritma
2. Muhammad Siddik Syahputra
  - a. Merancang alur dan penggunaan library pada system aplikasi
  - b. Menentukan kompleksitas program
  - c. Membuat Laporan
  - d. Membuat Alur Program
3. Ahmad Yuda Zaki Yamani
  - a. Setup projek aplikasi hasil dari rancangan yang sudah didiskusikan
  - b. Implementasi dari fungsi fungsi yang sudah dirancang
  - c. Maintance sistem aplikasi termasuk perbaikan bug
  - d. Membuat Laporan
4. Nurihsan Febrianto
  - a. Merancang struktur data json untuk penampungan data
  - b. Merancang semua fungsi untuk pengolahan data
  - c. Membuat Laporan
  - d. Testing aplikasi, memastikan aplikasi sudah berjalan dengan benar dan effisien