

Tugas Pendahuluan Modul 5
STRUKTUR DATA - Ganjil 2024/2025
"Single_Linked_List_Bagian_2"

Ketentuan Tugas Pendahuluan

1. Tugas Pendahuluan dikerjakan secara **Individu**.
2. TP ini bersifat **WAJIB**, tidak mengerjakan = **PENGURANGAN POIN JURNAL / TES ASESMEN**.
3. Hanya **MENGUMPULKAN** tetapi **TIDAK MENERJAKAN** = **PENGURANGAN POIN JURNAL / TES ASESMEN**.
4. Deadline pengumpulan TP Modul 2 adalah Senin, 30 September 2024 pukul 07.30 WIB.
5. **TIDAK ADA TOLERANSI KETERLAMBATAN, TERLAMBAT ATAU TIDAK MENGUMPULKAN TP MAKA DIANGGAP TIDAK MENERJAKAN**.
6. **DILARANG PLAGIAT (PLAGIAT = E)**.
7. Kerjakan TP dengan jelas agar dapat dimengerti.
8. Codingan diupload di Github dan upload Laporan di Lab menggunakan format **PDF** dengan ketentuan:
TP_MOD_[XX]_NIM_NAMA.pdf

CP (WA):

- Andini (082243700965)
- Imelda (082135374187)

Ketentuan Tugas Pendahuluan

- Untuk soal teori **JAWABAN DIKETIK DENGAN RAPI** dan untuk soal algoritma **SERTAKAN SCREENSHOOT CODINGAN DAN HASIL OUTPUT**.
- Deadline pengumpulan TP Modul 5 adalah **Senin, 14 Oktober 2024** pukul 06.00 WIB.
- **TIDAK ADA TOLERANSI KETERLAMBATAN, TERLAMBAT ATAU TIDAK MENGUMPULKAN TP ONLINE MAKA DIANGGAP TIDAK MENGERJAKAN.**
- **DILARANG PLAGIAT (PLAGIAT = E).**
- Kerjakan TP dengan jelas agar dapat dimengerti.
- Untuk setiap soal nama fungsi atau prosedur **WAJIB** menyertakan **NIM**, contoh: **insertFirst_130122xxxx**.
- File diupload di LMS menggunakan format **PDF** dengan ketentuan : **TP_MODX_NIM_KELAS.pdf**

```
Contoh:  
int searchNode_130122xxxx (List L, int X);
```

CP:

- Raihan (089638482851)
- Kayyisa (085105303555)
- Abiya (082127180662)
- Rio (081210978384)

SELAMAT MENGERJAKAN^^

LAPORAN PRAKTIKUM
PERTEMUAN 5
STRUKTUR DATA



Nama :

Zulfa Mustafa Akhyar Iswahyudi (2311104010)

Dosen :

Yudha Islami Sulistya

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2024

A. Tujuan

Untuk melatih kompetensi Mahasiswa untuk memperdalam skill pemrograman C++

B. Tools

Codeblocks, VSCode, Github

TUGAS PENDAHULUAN – UNGUIDED

1.) searchElementIdSLL (Cari nilai data tertentu dalam SLL)

Sintaks program kali adalah untuk mencari nilai tertentu yang ingin kita cari.

Seperti yang biasa dilakukan untuk pembuatan inisial pertama pada barisan SLL beserta dengan isi data yang bertipe Node, kita buat masing-masing constructor untuk Node dan List-nya. Untuk Node perlu kita inisialisasi tipe data integer beserta pembentuk sumber utama dari struktur SLL, yaitu Node. Dalam Node ini kita perlukan deklarasi pointer 'next' guna pembacaan data dalam SLL dapat berjalan baik.

Selanjutnya buat method untuk memasukkan data yaitu 'insertLast' dengan parameter List &L dan pendefinisian tipe data value yang akan mengisi SLL dengan tipe data integer. Dalam method ini kita harus deklarasi Node baru agar sumber utama untuk membentuk struktur SLL dapat dibuat dahulu. Node tadi harus diatur dengan variabel newNode agar prosesnya dapat berjalan, jangan lupa untuk mendeklarasi data beserta pointer-nya dengan newNode agar di-set default bahwa data adalah 'value' dan SLL ini masih berstatus kosong dulu.

Buat pengkondisian jika data pertama dalam SLL masih kosong maka SLL harus diinisialisasi dengan newNode agar data-data eksternal dapat dimasukkan ke dalam SLL sewaktu kita inputkan pada sintaks utamanya.

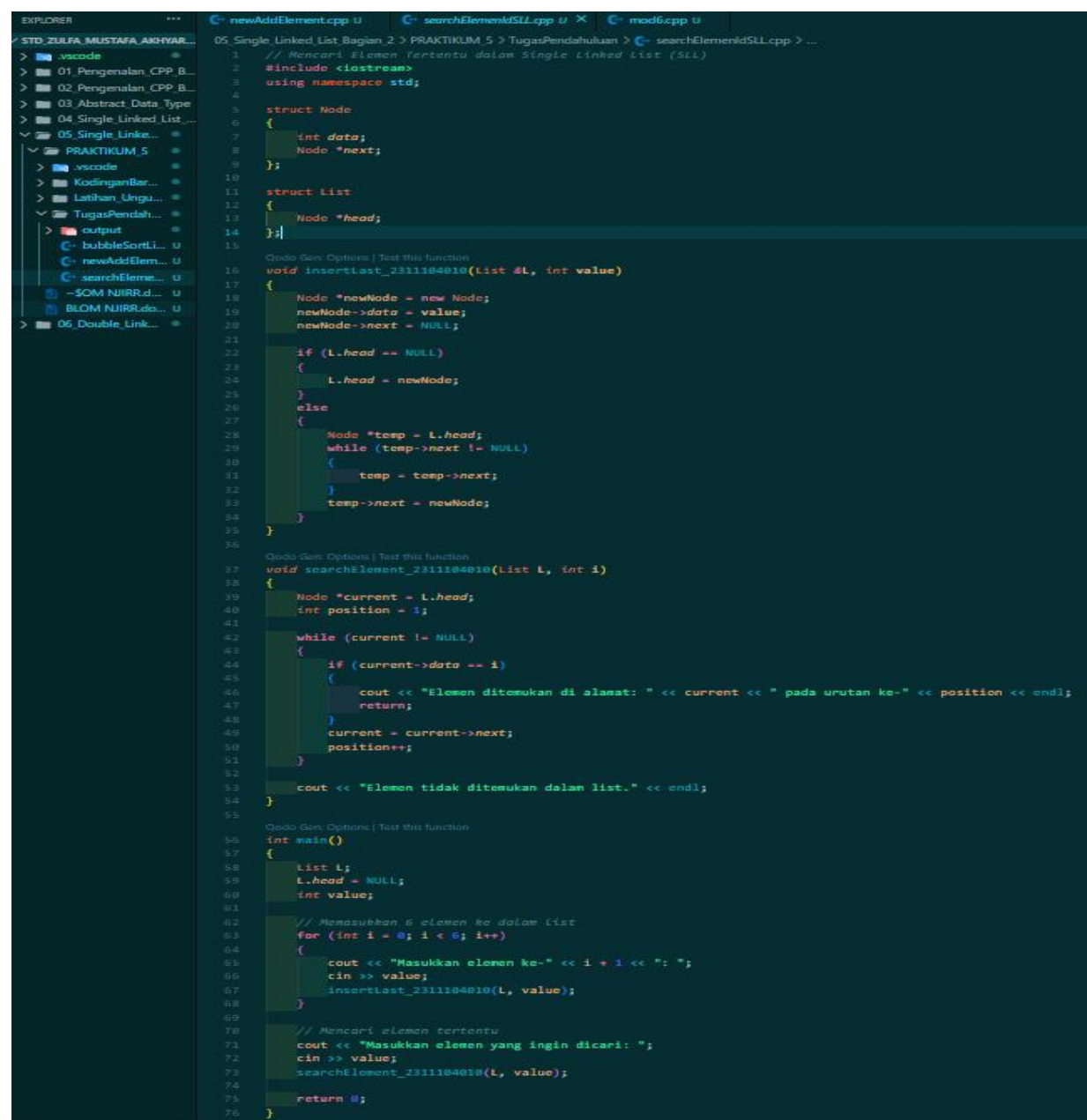
Namun jika data dalam SLL tidak kosong, maka data eksternal yang kita inputkan tadi akan diidentifikasi dengan variabel sementara 'temp' dan akan terisi di posisi belakang setelah data pertama.

Method kedua adalah untuk mencari nilai yang kita inginkan, yaitu 'searchElement' dengan parameter List untuk pemanggilan SLL dan deklarasi 'i' dengan tipe data integer. Didalam method ini SLL harus diintegrasikan dengan fungsi *current agar Node saat ini akan di-set sebagai data pertama dalam SLL. Nantinya juga posisi pencarian ini juga akan di-set dengan nilai 1 yang berarti default nilainya dimulai dari data pada posisi pertama atau data terdepan dalam SLL.

Perulangan while kali ini digunakan untuk mendeteksi keseluruhan data dari Node pertama hingga Node terakhir, artinya keseluruhan data akan di crosscheck oleh perulangan ini. Kondisikan lagi jika data yang dicari oleh fungsi current telah mendeteksi kesesuaian antara nilai 'i' dengan data didalam SLL, maka program akan mengatakan kalo elemen yang kita cari ditemukan pada alamat sekian dan pada urutan sekian.

Pengkondisian tersebut akan terpenuhi juga apabila fungsi current tidak menemukan data yang sesuai dengan pencarian yang dilakukan oleh pointer 'next', maka variabel position yang di-set pada nilai 1 tadi akan bertambah iterasinya sesuai jumlah nilai dalam SLL.

Namun jika sampai akhir belum menemukan nilainya, maka program menyatakan elemen gada.



```
05 Single_Linked_List_Bagian_2 > PRAKTIKUM_5 > TugasPendahuluan > searchElemenIdSLL.cpp > ...
1 // Mencari Elemen Tertentu dalam Single Linked List (SLL)
2 #include <iostream>
3 using namespace std;
4
5 struct Node
6 {
7     int data;
8     Node *next;
9 };
10
11 struct List
12 {
13     Node *head;
14 };
15
16 // Code Snippets | Test this function
17 void insertLast_2311184010(List &L, int value)
18 {
19     Node *newNode = new Node;
20     newNode->data = value;
21     newNode->next = NULL;
22
23     if (L.head == NULL)
24     {
25         L.head = newNode;
26     }
27     else
28     {
29         Node *temp = L.head;
30         while (temp->next != NULL)
31         {
32             temp = temp->next;
33         }
34         temp->next = newNode;
35     }
36 }
37
38 // Code Snippets | Test this function
39 void searchElement_2311184010(List L, int i)
40 {
41     Node *current = L.head;
42     int position = 1;
43
44     while (current != NULL)
45     {
46         if (current->data == i)
47         {
48             cout << "Elemen ditemukan di alamat: " << current << " pada urutan ke-" << position << endl;
49             return;
50         }
51         current = current->next;
52         position++;
53     }
54     cout << "Elemen tidak ditemukan dalam list." << endl;
55 }
56
57 // Code Snippets | Test this function
58 int main()
59 {
60     List L;
61     L.head = NULL;
62     int value;
63
64     // Memasukkan 6 elemen ke dalam list
65     for (int i = 0; i < 6; i++)
66     {
67         cout << "Masukkan elemen ke-" << i + 1 << ": ";
68         cin >> value;
69         insertLast_2311184010(L, value);
70     }
71
72     // Mencari elemen tertentu
73     cout << "Masukkan elemen yang ingin dicari: ";
74     cin >> value;
75     searchElement_2311184010(L, value);
76
77     return 0;
78 }
```

Output :

```
• .\'searchElemenIdSLL.exe\'
Masukkan elemen ke-1: 10
Masukkan elemen ke-2: 15
Masukkan elemen ke-3: 8
Masukkan elemen ke-4: 4
Masukkan elemen ke-5: 6
Masukkan elemen ke-6: 9
Masukkan elemen yang ingin dicari: 10
Elemen ditemukan di alamat: 0x27950ba88c0 pada urutan ke-1

• .\'searchElemenIdSLL.exe\'
• Masukkan elemen ke-1: 9
  Masukkan elemen ke-2: 4
  Masukkan elemen ke-3: 5
  Masukkan elemen ke-4: 8
  Masukkan elemen ke-5: 2
  Masukkan elemen ke-6: 7
  Masukkan elemen yang ingin dicari: 3
  Elemen tidak ditemukan dalam list.
```

2.) bubbleSortList (Pencarian SLL dengan Bubble Sort)

Struktural pembangun SLL-nya masih dengan sintaks deklarasi yang sama. **Untuk method penambahan data maupun method untuk menampilkan data SLL mulai dari deklarasi, parameter dan pengkondisiannya masih sama seperti program yang awal.** Atau bisa dibilang untuk method penambahan data dan method menampilkan data setiap file class C++ tetap menggunakan poin-poin sintaks yang sama. Jadi tidak perlu dijelaskan panjang lebar lagi.

Namun ada method yang berbeda untuk membuat operasi urutan bubblesort, yaitu method bubbleSort. Didalamnya ada fungsi baru yaitu 'swapped'. Fungsi tersebut adalah untuk melakukan cek iterasi dalam proses pengecekan nilai-nilai dalam SLL apakah ada pertukaran data atau tidak. Sementara kita inisiasi fungsi tersebut dengan nilai false yang menandakan bahwa pertukaran data belum terjadi. Pengkondisian didalamnya adalah proses pertukaran data depan-belakang maupun belakang-depan.

Yang membedakan hanya proses operasinya sekarang. Namun ada variabel yang harus di set default sebelum mengoperasikan SLL, yaitu 'List L' dan value yang bertipe integer.

Buat perulangan sebuah inputan yang dimana berjumlah 5 dengan pemanggilan method 'insertLast' supaya data dapat masuk dalam SLL.

Setelah selesai, panggil method bubbleSort untuk menukar-nukar urutan posisi nilai data dan method displayList untuk menampilkan keseluruhan SLL setelah ditukar-tukar posisinya.

```
EXPLORER
...
STD ZULHA MUSTAWA ARIYAN...
> .vscode
> 01_Pengenalan_CPP_B...
> 02_Pengenalan_CPP_B...
> 03_Abstract_Data_Type...
> 04_Single_Linked_List...
> 05_Single_Linked_List...
  > PRAKTIKUM_5
    > .vscode
    > KodinBar...
    > Latihan_Ungu...
    > TugasPendah...
    > output
    > bubbleSortL... U
    > newAddElem... U
    > searchElem... U
    > SOM NARR.d... U
    > BLOM NARR.d... U
    > 06_Double_Link...

C- bubbleSortList.cpp u x
05_Single_Linked_List_Bagian_2 > PRAKTIKUM_5 > TugasPendahuluan > C- bubbleSortList.cpp > ...
1 // Mengurutkan List Menggunakan Bubble Sort
2 #include <iostream>
3 using namespace std;
4 struct Node
5 {
6     int data;
7     Node *next;
8 };
9
10 struct List
11 {
12     Node *head;
13 };
14
15 void insertLast_2311104010(List &L, int value)
16 {
17     Node *newNode = new Node;
18     newNode->data = value;
19     newNode->next = NULL;
20
21     if (L.head == NULL)
22     {
23         L.head = newNode;
24     }
25     else
26     {
27         Node *temp = L.head;
28         while (temp->next != NULL)
29         {
30             temp = temp->next;
31         }
32         temp->next = newNode;
33     }
34 }
35
36 void bubbleSort_2311104010(List &L)
37 {
38     if (L.head == NULL)
39         return;
40
41     bool swapped;
42     Node *current;
43     Node *last = NULL;
44
45     do
46     {
47         swapped = false;
48         current = L.head;
49
50         while (current->next != last)
51         {
52             if (current->data > current->next->data)
53             {
54                 swap(current->data, current->next->data);
55                 swapped = true;
56             }
57             current = current->next;
58         }
59         last = current;
60     } while (swapped);
61 }
62
63 void displayList_2311104010(List L)
64 {
65     Node *current = L.head;
66     while (current != NULL)
67     {
68         cout << current->data << " ";
69         current = current->next;
70     }
71     cout << endl;
72 }
73
74 int main()
75 {
76     List L;
77     L.head = NULL;
78     int value;
79
80     // Memasukkan 5 elemen ke dalam List
81     for (int i = 0; i < 5; i++)
82     {
83         cout << "Masukkan elemen ke-" << i + 1 << ": ";
84         cin >> value;
85         insertLast_2311104010(L, value);
86     }
87
88     // Mengurutkan List dengan Bubble sort
89     bubbleSort_2311104010(L);
90
91     // Menampilkan List yang sudah terurut
92     cout << "List setelah diurutkan: ";
93     displayList_2311104010(L);
94
95     return 0;
96 }
```

Output :

```
\'bubbleSortList.exe'  
Masukkan elemen ke-1: 5  
Masukkan elemen ke-2: 9  
Masukkan elemen ke-3: 2  
Masukkan elemen ke-4: 4  
Masukkan elemen ke-5: 7  
List setelah diurutkan: 2 4 5 7 9
```

3.) newAddElementIDSLL (Penambahan Nilai Data Baru)

Kali ini yang berbeda hanya method insertSorted. Ini adalah method untuk memasukkan nilai data baru kedalam SLL. Panggil parameter 'List &L' dan variabel value yang bertipe integer.

Kondisikan jika data pertama kosong dalam SLL, otomatis data baru ini bisa lebih besar atau sama dengan nilai data pertama dan akan berada diposisi pertama. Jika list tidak kosong, kondisikan selama proses pengecekan interaksi data dalam SLL tidak kosong dan data yang diiterasi tidak lebih besar dari data baru, program akan terus mengiterasi keseluruhan data sampai data baru ini ada diposisi yang sesuai nilainya.

Dalam operasi programnya, buat perulangan untuk 4 buah inputan dengan method 'insertLast'. Lalu kita bisa memasukkan nilai baru yang akan ditampung variabel value dan kita bisa gunakan variabel 'insertSorted'.

Terakhir kita sudah bisa melihat tampilan keseluruhan nilai data SLL dengan method displayList.

EXPLORER

05_Single_Linked_List_Bagian_2 > PRAKTIKUM_5 > TugasPendahuluan > newAddElement.cpp > main()

01_Pengenalan_CPP_B...

02_Pengenalan_CPP_B...

03_Abstract_Data_Type

04_Single_Linked_List...

05_Single_Linked_List...

PRAKTIKUM_5

output

bubbleSortLi... U

newAddElem... U

searchEleme... U

~\$OM NUJRR.d... U

BLOM NUJRR.do... U

06_Double_Link...

16 void insertLast_2311104010(List &L, int value)

17 {

18 Node *newNode = new Node;

19 newNode->data = value;

20 newNode->next = NULL;

21

22 if (L.head == NULL)

23 {

24 L.head = newNode;

25 }

26 else

27 {

28 Node *temp = L.head;

29 while (temp->next != NULL)

30 {

31 temp = temp->next;

32 }

33 temp->next = newNode;

34 }

35 }

36

Qodo Gen: Options | Test this function

37 void insertSorted_2311104010(List &L, int value)

38 {

39 Node *newNode = new Node;

40 newNode->data = value;

41

42 if (L.head == NULL || L.head->data >= value)

43 {

44 newNode->next = L.head;

45 L.head = newNode;

46 }

47 else

48 {

49 Node *current = L.head;

50 while (current->next != NULL && current->next->data < value)

51 {

52 current = current->next;

53 }

54 newNode->next = current->next;

55 current->next = newNode;

56 }

57 }

58

Qodo Gen: Options | Test this function

59 void displayList_2311104010(List L)

60 {

61 Node *current = L.head;

62 while (current != NULL)

63 {

64 cout << current->data << " ";

65 current = current->next;

66 }

67 cout << endl;

68 }

69

Qodo Gen: Options | Test this function

70 int main()

71 {

72 List L;

73 L.head = NULL;

74 int value;

75

76 // Memasukkan 4 elemen ke dalam List

77 for (int i = 0; i < 4; i++)

78 {

79 cout << "Masukkan elemen ke-" << i + 1 << ": ";

80 cin >> value;

81 insertLast_2311104010(L, value);

82 }

83

84 cout << "Masukkan elemen baru yang ingin ditambahkan: ";

85 cin >> value;

86 insertSorted_2311104010(L, value);

87

88 cout << "List setelah elemen baru dimasukkan: ";

89 displayList_2311104010(L);

90

91 return 0;

92 }

Output :

```
● _Single_Linked_List_Bagian_2\PRAKTIKUM_5\TugasPendahuluan\output> & .\'newAddElement.exe'  
Masukkan elemen ke-1: 7  
Masukkan elemen ke-2: 2  
Masukkan elemen ke-3: 5  
Masukkan elemen ke-4: 9  
Masukkan elemen baru yang ingin ditambahkan: 6  
List setelah elemen baru dimasukkan: 6 7 2 5 9
```

LATIHAN – UNGUIDED

1.) Mod_5

Kita percepat tempo penjelasannya.

Variabel mahasiswa harus berupa constructor yang isinya Nim dan nama, kemudian untuk deklarasi SLL-nya adalah 'Mahasiswa' yang diberi pointer next supaya data-data baru bisa masuk kedalamnya.

Masuk ke method untuk menambahkan data mahasiswa yaitu **addMhs**, parameterkan dulu SLL-nya dan dua variabel sebelumnya yaitu 'nim' dan 'nama'. Ingat jangan lupa kita set default data pertama SLL-nya juga. Kondisikan apabila data pertama bisa terdeklarasi sebagai data baru untuk data seterusnya yang masuk ke dalam SLL.

Method berikutnya adalah **findMhs**. Method ini berfungsi untuk mencari data mahasiswa berdasarkan nim. Parameterkan data pertama dalam SLL Mahasiswa dan variabel nim agar method ini bisa mendeteksi data berupa integer dari suku data dalam SLL.

Kondisikan selama pengecekan data dimulai dari data pertama tidak kosong maka method akan terus cari nim yang sesuai dengan inputan user, namun jika masih ga ketemu, maka program menyatakan kalo nim tidak ditemukan.

Method terakhir yaitu **displayMhs** untuk menampilkan seluruh data mahasiswa dari nim sampai nama. Pengkondisiannya sama seperti method findMhs, jadi tak perlu dijelaskan ulang.

Oh iya, jangan lupa method **menu**, tak perlu parameter apa-apa. Cukup masukkan output untuk menampilkan semacam menu untuk add, search data, display data, dan keluar.

```
DOLORE
STD, RULFA, MUSTARI, AMIRRA...
> #include <iostream>
> #include <string>
> using namespace std;
> struct Mahasiswa
> {
>     int NIM;
>     string Nama;
>     Mahasiswa *next;
> };
> // Fungsi untuk menambahkan data mahasiswa ke linked list
> void addMhs_2311104010(Mahasiswa *head, int nim, string nama)
> {
>     Mahasiswa *newMahasiswa = new Mahasiswa;
>     newMahasiswa->NIM = nim;
>     newMahasiswa->Nama = nama;
>     newMahasiswa->next = nullptr;
>
>     if (head == nullptr)
>     {
>         head = newMahasiswa;
>     }
>     else
>     {
>         Mahasiswa *temp = head;
>         while (temp->next != nullptr)
>         {
>             temp = temp->next;
>         }
>         temp->next = newMahasiswa;
>     }
> }
> // Fungsi untuk mencari mahasiswa berdasarkan NIM
> // Query User Options / Test the function
> void findMhs_2311104010(Mahasiswa *head, int nim)
> {
>     Mahasiswa *temp = head;
>     while (temp != nullptr)
>     {
>         if (temp->NIM == nim)
>         {
>             cout << "Data mahasiswa ditemukan: " << temp->Nama << " (NIM: " << temp->NIM << ") " << endl;
>             return;
>         }
>         temp = temp->next;
>     }
>     cout << "Data mahasiswa dengan NIM " << nim << " tidak ditemukan." << endl;
> }
> // Fungsi untuk menampilkan seluruh mahasiswa
> // Query User Options / Test the function
> void displayMhs_2311104010(Mahasiswa *head)
> {
>     Mahasiswa *temp = head;
>     while (temp != nullptr)
>     {
>         cout << "NIM: " << temp->NIM << ", Nama: " << temp->Nama << endl;
>         temp = temp->next;
>     }
> }
> // Fungsi menu
> // Query User Options / Test the function
> void menu()
> {
>     cout << "===== Menu Mahasiswa =====" << endl;
>     cout << "1. Add (tambah data)" << endl;
>     cout << "2. search (cari nim-nya ya)" << endl;
>     cout << "3. display (tampilkan semua data-nya)" << endl;
>     cout << "4. Keluar" << endl;
>     cout << "Pilih opsi: ";
> }
> // Query User Options / Test the function
> int main()
> {
>     Mahasiswa *head = nullptr;
>     int pilihan;
>     int nim;
>     string nama;
>
>     do
>     {
>         menu();
>         cin >> pilihan;
>
>         switch (pilihan)
>         {
>             case 1:
>                 cout << "Masukkan NIM: ";
>                 cin >> nim;
>                 cout << "Masukkan Nama: ";
>                 cin.ignore(); // Membersihkan buffer read line
>                 getline(cin, nama);
>                 addMhs_2311104010(head, nim, nama);
>                 cout << "Data mahasiswa berhasil ditambahkan." << endl;
>                 break;
>
>             case 2:
>                 cout << "Masukkan NIM yang ingin dicari: ";
>                 cin >> nim;
>                 findMhs_2311104010(head, nim);
>                 break;
>
>             case 3:
>                 cout << "Daftar seluruh mahasiswa:" << endl;
>                 displayMhs_2311104010(head);
>                 break;
>
>             case 4:
>                 cout << "Keluar dari program." << endl;
>                 break;
>
>             default:
>                 cout << "Pilihan tidak valid!" << endl;
>         }
>     } while (true);
> }
```

Output :

```
PS C:\Users\HUAWEI\OneDrive\Documents\ALL_1
_5\Latihan_Unguided\output> & .\'mod5.exe'
===== Menu Mahasiswa =====
1. Add (nambah data)
2. search (cari nim-nya ya)
3. display (tampilkan semua data-nya)
4. Keluar
Pilih opsi: 1
Masukkan NIM: 231
Masukkan Nama: nam in hak
Data mahasiswa berhasil ditambahkan.

===== Menu Mahasiswa =====
1. Add (nambah data)
2. search (cari nim-nya ya)
3. display (tampilkan semua data-nya)
4. Keluar
Pilih opsi: 1
Masukkan NIM: 232
Masukkan Nama: wang hao ming
Data mahasiswa berhasil ditambahkan.

===== Menu Mahasiswa =====
1. Add (nambah data)
2. search (cari nim-nya ya)
3. display (tampilkan semua data-nya)
4. Keluar
Pilih opsi: 1
Masukkan NIM: 233
Masukkan Nama: han goo rang
Data mahasiswa berhasil ditambahkan.

===== Menu Mahasiswa =====
1. Add (nambah data)
2. search (cari nim-nya ya)
3. display (tampilkan semua data-nya)
4. Keluar
Pilih opsi: 2
Masukkan NIM yang ingin dicari: 233
```

===== Menu Mahasiswa =====

1. Add (nambah data)
2. search (cari nim-nya ya)
3. display (tampilin semua data-nya)
4. Keluar

Pilih opsi: 1

Masukkan NIM: 233

Masukkan Nama: han goo rang

Data mahasiswa berhasil ditambahkan.

===== Menu Mahasiswa =====

1. Add (nambah data)
2. search (cari nim-nya ya)
3. display (tampilin semua data-nya)
4. Keluar

Pilih opsi: 2

Masukkan NIM yang ingin dicari: 233

Data mahasiswa ditemukan: han goo rang (NIM: 233)

===== Menu Mahasiswa =====

1. Add (nambah data)
2. search (cari nim-nya ya)
3. display (tampilin semua data-nya)
4. Keluar

Pilih opsi: 3

Daftar seluruh mahasiswa:

NIM: 231, Nama: nam in hak

NIM: 232, Nama: wang hao ming

NIM: 233, Nama: han goo rang

===== Menu Mahasiswa =====

1. Add (nambah data)
2. search (cari nim-nya ya)
3. display (tampilin semua data-nya)
4. Keluar

Pilih opsi: 4

Keluar dari program.

SOURCE CODE WAKTU DI KELAS – KODINGAN ASPRAK

```
SLL.cpp

#include <iostream>
using namespace std;

// definisi pointer ke elemen
struct Elemen
{
    int data;
    Elemen *next;
};

// definisi tipe data untuk list
struct List
{
    Elemen *first;
};

// buat list kosong
void createList(List &L)
{
    L.first = NULL;
}

// fungsi alokasi elemen baru
Elemen *alokasi(int x)
{
    Elemen *p = new Elemen;
    if (p != NULL)
    {
        p->data = x;
        p->next = NULL;
    }
    return p;
}

// tambah elemen didepan
void insertFirst(List &L, Elemen *P)
{
    P->next = L.first;
    L.first = P;
}

// cetak semua elemen
void printInfo(List L)
{
    Elemen *p = L.first;
    while (p != NULL)
    {
        cout << p->data << " ";
        p = p->next;
    }
    cout << endl;
}

// cari elemen x
Elemen *findElm(List L, int x)
{
    Elemen *p = L.first;
    while (p != NULL)
    {
        if (p->data == x)
        {
            return p;
        }
        p = p->next;
    }
    return NULL;
}

// fungsi hitung total nilai semua elemen
int hitungTotal(List L)
{
    Elemen *p = L.first;
    int total = 0;
    while (p != NULL)
    {
        total += p->data;
        p = p->next;
    }
    return total;
}
```



```

mainSLL.cpp

#include <iostream>
#include "SLL.cpp"

int main()
{
    List L;
    Elemen *P1, *P2, *P3, *P4, *P5 = NULL;
    createList(L);

    // alokasi + insert data
    P1 = alokasi(2);
    insertFirst(L, P1);

    P2 = alokasi(0);
    insertFirst(L, P2);

    P3 = alokasi(8);
    insertFirst(L, P3);

    P4 = alokasi(12);
    insertFirst(L, P4);

    P5 = alokasi(9);
    insertFirst(L, P5);

    // cetak output
    printInfo(L);

    // Cari elemen nilai 8 (contoh)
    Elemen *found = findElm(L, 8);
    if (found != NULL)
    {
        cout << "Elemen dengan nilai 8 ditemukan" << endl;
    }
    else
    {
        cout << "Elemen dengan nilai 8 tidak ditemukan" << endl;
    }

    // hitung total semua elemen
    int total = hitungTotal(L);
    cout << "Total elemen keseluruhan: " << total << endl;
}

```



2311104010_ZULFA MUSTAFA AKHYAR ISWAHYUDI

Han-Goo-Rang

Semoga Selalu diberi kemudahan^^