

PENJELASAN KASUS 6

Studi Kasus 6: Kembangkan program studi kasus 5 yang sebelumnya berupa array statis menjadi sebuah linked list

Untuk menyelesaikan studi kasus ke-6, saya mengubahnya menjadi enam product backlog sebagai berikut:

1. Meningkatkan dan memperbaiki ADT linked list sehingga dapat digunakan dengan mudah dan flexible
2. Mengubah ADT CaseCity bagian manajemen list kota yang sebelumnya array list menjadi linked list
3. Memperbaiki logic untuk menyesuaikan perubahan struktur data
4. Meningkatkan UI program

1.1 Gambaran Struktur data:

Analisis Pengembangan CityCase

Gambaran Struktur data

List kota

```
graph LR
    ListKota[List kota] --> Node1["Bandung"]
    Node1 --> Node2["Asep"]
    Node2 --> Node3["Pudi"]
    Node3 --> Node4["Pori"]
    Node4 --> Node5["Jember"]
    Node5 --> Node6["Rusfel"]
    Node6 --> Node7["Hadi"]
    Node7 --> Node8["Udin"]
    Node8 --> Node9["Patenan"]
    Node9 --> Node10["Merci"]
    Node10 --> Node11["Sula"]
    Node11 --> Node12["Jody"]
```

kebutuhan record:

Record City:

String list;

List Nama; \Rightarrow diambil dari linked list

Cities; \Rightarrow next;

kebutuhan modul:

- o) Create Cities \Rightarrow Mempersiapkan list
- o) Alokasi city \Rightarrow Alokasi memori node + isi nilai dengan nilai parameter yg di passing, untuk list name pastikan sudah di deklare

o) Dealokasi \Rightarrow dealokasi untuk struct cities
Pastikan string di dealokasi juga, karena pengurutan konsep string literat

o) Insertion function \Rightarrow serupa dengan linked list dengan loc beberapa penamban

o) Deletion function \Rightarrow serupa dengan linked list dengan beberapa konfigurasi

Contoh eksploitasi:

- o) Create = konstruktor?
- o) apakah perlu destructor?
- o) butuh getter setter?
- o) Print data 1 kota & seluruh kota
- o) hitung seluruh nama

1.2 Komponen ADT yang akan digunakan:

1. Linked.h (sudah dimodifikasi agar bisa menerima char*)
2. Linked.c (implementasi program linked.h) [#include "linked.h"]
3. Main.c (main program) [#include "controller.h"]
4. CaseCity.h (ADT yang akan menampung kebutuhan tambahan untuk array kota dll) [#include "linked.h"]
5. CaseCity.c (implementasi program dari CaseCity.h) [#include "CaseCity.h"]
6. Ui.h (ADT yang akan menyimpan Ui, dipisahkan sebagai bentuk Latihan bila menggunakan GUI)
7. Ui.c (implementasi program dari ui.h) [#include "ui.h"]
8. controller.h (ADT logic untuk memenuhi studi kasus) [#include "ui.h", #include "CaseCity.h"]
9. controller.c (implementasi program controller.h) [#include "controller.h"]

1.3 Penyelesaian Backlog

1. Meningkatkan dan memperbaiki ADT linked list sehingga dapat digunakan dengan mudah dan flexible

```
1  #ifndef LINKED_H
2  #define LINKED_H
3
4  #include <stdio.h>
5  #include <stdlib.h>
6  #include <string.h>
7  #include "boolean.h"
8
9  typedef enum {
10     TYPE_INT,
11     TYPE_STRING
12 } DataType;
13
14 typedef char* string;
15 typedef struct tElmtList *address;
16 typedef struct tElmtList {
17     union {
18         int intInfo;
19         string strInfo;
20     };
21     address next;
22 } ElmtList;
23
24 typedef struct {
25     DataType type;
26     address First;
27 } List;
28
29 #define Nil NULL
30 #define First(L) ((L).First)
31 #define next(P) ((P)->next)
32 #define infoInt(P) ((P)->intInfo)
33 #define infoStr(P) ((P)->strInfo)
34
```

```

35 boolean isEmpty(address P);
36 boolean ListEmpty(List L);
37 void CreateAddress(address *P);
38 void CreateList(List *L, DataType type);
39 address AlokasiInt(int X);
40 address AlokasiCharS(char* nama);
41 void DeAlokasiInt(address P);
42 void DeAlokasiCharS(address P);
43 void DeAlokasi(address P);
44
45 void InsertFirst(address *P, address newNode);
46 void InsertLast(address *P, address newNode);
47 void InsertAfter(address *pBef, address PNew);
48 void InsertBefore(address *pAft, address *p, address PNew);
49 void InsertFVInt(address *P, int info);
50 void InsertFVCharS(address *P, char* info);
51
52 #define InsertFirstV(a, b) _Generic((b), \
53     int: InsertFVInt, \
54     char*: InsertFVCharS \
55 )(a, b)
56
57 void InsertLVInt(address *P, int info);
58 void InsertLVCharS(address *P, char* info);
59
60 #define InsertLastV(a, b) _Generic((b), \
61     int: InsertLVInt, \
62     char*: InsertLVCharS \
63 )(a, b)
64
65 void InsertAVInt(address *P, int Before, int info);
66 void InsertAVCharS(address *P, string Before, string info);
67
68 #define InsertAfterV(a, b, c) _Generic((b), \
69     int: InsertAVInt, \
70     char*: InsertAVCharS \
71 )(a, b, c)
72
73 void InsertBVInt(address *P, int After, int info);
74 void InsertBVCharS(address *P, string After, string info);
75
76 #define InsertBeforeV(a, b, c) _Generic((b), \
77     int: InsertBVInt, \
78     char*: InsertBVCharS \
79 )(a, b, c)
80
81 void DeleteFirstInt(List *L, int *info);
82 void DeleteFirstCharS(List *L, string *info);
83
84 #define DeleteFirst(a, b) _Generic((b), \
85     int*: DeleteFirstInt, \
86     char**: DeleteFirstCharS \
87 )(a, b)
88

```

```

88
89 void DeleteLastInt(List *L, int *info);
90 void DeleteLastCharS(List *L, string *info);
91
92 #define DeleteLast(a, b) _Generic((b), \
93     int*: DeleteLastInt, \
94     char*: DeleteLastCharS \
95 )(a, b)
96
97 void DeleteByVInt(List *L, int info);
98 void DeleteByVCharS(List *L, string info);
99
100 #define DeleteByValue(a, b) _Generic((b), \
101     int: DeleteByVInt, \
102     char*: DeleteByVCharS \
103 )(a, b)
104
105 void ClearList(List *L);
106
107 void printInfoInt(List L);
108 void printInfoCharS(List L);
109
110 #define PrintList(L) ((L).type == TYPE_INT ? printInfoInt(L) : printInfoCharS(L))
111
112 int CountList(List L);
113 address SearchPrec(List L, int info);
114 address SearchByVInt(List L, int info);
115 address SearchByVCharS(List L, char* info);
116
117 #define SearchByValue(a, b) _Generic((b), \
118     int: SearchByVInt, \
119     char*: SearchByVCharS \
120 )(a, b)
121
122 address getReverseList(List L);
123 void ReverseList(List *L);
124 void CopyList(List L1, List *L2);
125 int GetFrontValueInt(List L);
126 string GetFrontValueCharS(List L);
127
128 #define GetFrontValue(L) ((L).type == TYPE_INT ? GetFrontValueInt(L) :
    GetFrontValueCharS(L))
129
130 int GetTailValueInt(List L);
131 string GetTailValueCharS(List L);
132
133 #define GetTailValue(L) ((L).type == TYPE_INT ? GetTailValueInt(L) : GetTailValueCharS(L))
134
135 #endif

```

Peningkatan utama yang saya lakukan pada ADT linked adalah optimalisasi fleksibilitas dengan menerapkan fitur modern C yaitu macro generic untuk mendukung tipe data integer dan string sekaligus. Peningkatan ini dilakukan dengan tujuan agar ADT linked dapat beradaptasi dengan tipe data integer dan String. Untuk keseluruhan perubahan pada ADT linked saya uraikan menjadi butir – butir berikut:

- A. Optimalisasi fleksibilitas tipe data
 - B. Merapihkan struktur penulisan kode
 - C. Menambahkan penjelasan komentar yang lebih lengkap (pada gambar dihapus)
 - D. Menambahkan beberapa modul baru (untuk mendukung fleksibilitas)
2. Mengubah ADT CaseCity bagian manajemen list kota yang sebelumnya array list menjadi linked list

```
1  #ifndef CASECITY_H
2  #define CASECITY_H
3  #include "linked.h"
4  #include "boolean.h"
5  #define Head(L) ((L).Head)
6  #define next(P) ((P)->next)
7  #define kota(P) ((P)->kt)
8  #define ListName(P) ((P)->L)
9
10 extern int i;
11
12 typedef struct kota *Pcity;
13 typedef struct kota{
14     Pcity next;
15     char* kt;
16     List L;
17 } CityData;
18
19 typedef struct {
20     Pcity Head;
21 } ListCity;
22
23 boolean ListCityEmpty(ListCity C);
24 void CreatePcity(Pcity *A);
25 void CreateCityList(ListCity *C);
26 Pcity Constructor(char* cityName);
27 void Destructor(Pcity A);
28
29 void InsertAwal(Pcity *A, Pcity newNode);
30 void InsertAkhir(Pcity *A, Pcity newNode);
31 void InsertSetelah(Pcity *pBef, Pcity PNew);
32 void InsertSebelum(Pcity *pAft, Pcity *p, Pcity PNew);
33 void InsertAwalV(Pcity *P, char *kota);
34 void InsertAkhirV(Pcity *P, char *kota);
35
36 void DeleteAwal(ListCity *C);
37 void DeleteAkhir(ListCity *C);
38 void DeleteByName(ListCity *C, char* cityName);
39 void ClearListCity(ListCity *C);
```

```

40
41 Pcity FindCity(ListCity C, char* cityName);
42 void PrintAllCityData(ListCity C);
43 void PrintCityData(Pcity cityNode);
44 int countCity(ListCity C);
45 int countNameInCity(Pcity cityNode);
46 int CountTotalNames(Pcity A);
47 void CopyListCity(ListCity C1, ListCity *C2);
48
49 #endif

```

Pada bagian CaseCity perubahan utama yang dilakukan tentu saja pada bagian manajemen struktur data list kota, penggunaan init array sekarang sudah berubah menjadi constructor, destructor dan lain sebagainya. Pengimplementasian konsep linked list diterapkan diseluruh ADT CaseCity. Pembagiannya secara fungsinya dijabarkan melalui butir – butir berikut:

- a. Entry : menggunakan modul insertion yaitu InsertAwal, InsertAkhir, InsertAfter, InsertBefore, InsertAwalV, dan InsertAkhirV
 - b. Delete : Menggunakan bagaana modul deletion yaitu DeleteAwal, DeleteAkhir, DeleteByName, dan ClearListCity
 - c. Display : PrintAllCityData dan PribstCityData
 - d. Count : countCity, countNameInCity, dan countTotalNames
3. Memperbaiki logic untuk menyesuaikan perubahan struktur data
- Perubahan pada ADT logic sebenarnya merupakan bentuk adaptasi dari perubahan manajemen list data kota. Penamaan logic saya ubah menjadi controller karena saya rasa lebih tepat untuk menggambarkan fungsi ADT tersebut. Perubahan yang banyak terjadi pada ADT ini adalah penggunaan string literal (char*) pada parameter dan sebagai bagian dari proses input user (sebelumnya menggunakan array of char). Penerapan ini memerlukan kehati – hatian karena dapat menyebabkan leak of memory jika tidak ditangani dengan baik. Berikut contoh penerapannya
- Sebelum:

```

1 void handleDeleteName(CityData* cities, int cityCount) {
2     char cityName[MAX_CITY_LENGTH];
3     char personName[MAX_NAME_LENGTH];
4
5     readCityName(cityName);
6     int cityIndex = FindCity(cities, cityCount, cityName);
7
8     if (cityIndex >= 0) {
9         readPersonName(personName);
10        DeleteNameFromCity(cities, cityIndex, personName);
11        displayNameDeletedMessage(personName, cityName);
12    } else {
13        displayCityNotFoundMessage(cityName);
14    }
15 }

```


Sesudah:

```
1 void handleDeleteName(ListCity *cities) {
2     char *cityName = NULL;
3     char *personName = NULL;
4
5     system("cls");
6     drawbox(50, 9);
7     gotoxy(10, 2);
8     printf("Delete Person from City");
9
10    gotoxy(5, 4);
11    readCityName(&cityName);
12
13    Pcity city = FindCity(*cities, cityName);
14    if (city == NULL) {
15        gotoxy(5, 6);
16        displayCityNotFoundMessage(cityName);
17        free(cityName);
18        waitForEnter();
19        return;
20    }
21
22    gotoxy(5, 6);
23    readPersonName(&personName);
24
25    if (ListEmpty(ListName(city))) {
26        gotoxy(5, 8);
27        printf("No people registered in city '%s'!", cityName);
28        free(cityName);
29        free(personName);
30        waitForEnter();
31        return;
32    }
33
34    address person = SearchByValue(ListName(city), personName);
35    if (person == NULL) {
36        gotoxy(5, 8);
37        printf("Person '%s' not found in city '%s'!", personName, cityName);
38        free(cityName);
39        free(personName);
40        waitForEnter();
41        return;
42    }
43
44    DeleteByValue(&(ListName(city)), personName);
45
46    gotoxy(5, 8);
47    displayNameDeletedMessage(personName, cityName);
48
49    free(cityName);
50    free(personName);
51
52    waitForEnter();
53 }
```

Dari kedua gambar diatas kita dapat melihat penggunaan string literal sebagai penerima input user disini. Pada modul handleDeleteName input user digunakan untuk mencari node yang akan dihapus, setelah kita mendapatkan node yang kita inginkan kita perlu menghapus atau mendealokasikan memori variabel yang sebelumnya menampung string karena sudah tidak digunakan lagi.

4. Meningkatkan UI program

Pada program kali ini saya menerapkan draw box dan gotoxy sebagai penunjang ui program. Penerapan ini membuat program hasil program lebih baik, berikut hasilnya:

```

      All City Data

List of Cities:
City: Palembang
Names in Palembang: Mercy, Shin, Doley

City: Bandung
Names in Bandung: Asep, Rudi

City: Jakarta
Names in Jakarta: Roufiel Hado, Udin
```

```

D:\SDA\Praktek\Pertemuan 8' x + v

      Add Person to City

Masukkan nama kota: Bandung

Masukkan Nama: Septian

Person 'Septian' has been added to city 'Bandung' successfully!
Press Enter to continue...
```