Contoh Praktikum Algoritma dan Struktur Data

Nama : Agil Deriansyah Hasan
Nim : 4522210125

Dosen Pengajar :

Dra.SRI REZEKI CANDRA NURSARI,M.Kom
Prak. Algoritma dan Struktur Data - I

S1-Teknik Informatika
Fakultas Teknik
Universitas Pancasila 2023/2024

cnthprak13.cpp

```cpp
#include <iostream>
#include <iomanip>
#include <cctype>
#include <cstdlib>
#include <cstring>

#define hashsize 100
#define kodemk 30
#define namamk 13

using namespace std;

struct hashdata {
    char kode[kodemk];
    char nama[namamk];
};

static hashdata *hashtable[hashsize];
```

cnthprak13.cpp

```cpp
void clear();
void insert(hashdata *);
void searching(hashdata *);
int searchingprocess(hashdata *, int);
void deleting(hashdata *);
int hashfunction(hashdata *);
bool isindexempty(int);
int characteramount(char []);
void displaytable();

int main() {
    char ulang = 'Y', menu;
    clear();
    do {
        system("cls");
        puts("Proses Hash Table");
        puts("");
        puts("1. Input");
        puts("2. Delete");
        puts("3. Find");
        puts("4. Display");
        puts("5. Clear");
        puts("x. Keluar");
        puts("");
        cout << "Menu Pilihan Anda : "; cin >> menu;

        switch (menu) {
            case '1': {
                hashdata *array;
                array = new hashdata;
                puts("");
                cout << "KodeMK :"; cin >> array->kode;
                cout << "MataKuliah :"; cin >> array->nama;
                cout << endl;
                insert(array);
                break;
            }
            case '2': {
                hashdata *hapus;
                hapus = new hashdata;
                cout << "Hapus MataKuliah(KodeMK) :"; cin >> hapus->kode;
                deleting(hapus);
                delete hapus;
                break;
            }
```

cnthprak13.cpp

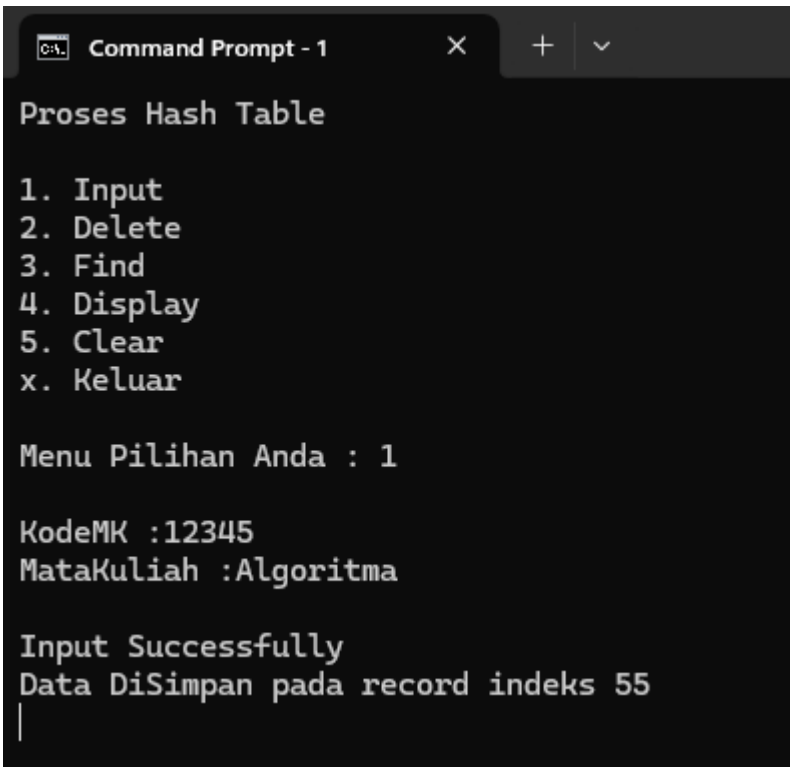
```
64              }
65              case '3': {
66                  hashdata *cari;
67                  cari = new hashdata;
68                  cout << "Cari MataKuliah (KodeMK) : "; cin >> cari->kode;
69                  searching(cari);
70                  delete cari;
71                  break;
72              }
73              case '4':
74                  displaytable();
75                  break;
76              case '5':
77                  clear();
78                  puts("Tabel Hash telah dikosongkan.");
79                  break;
80              case 'x':
81              case 'X':
82                  ulang = 'T';
83                  break;
84              default:
85                  puts("Pilihan diluar Menu yang Tersedia");
86                  break;
87          }
88          cin.get();
89      }
90      while (toupper(ulang) == 'Y');
91  }
92
93  void clear() {
94      for (int i = 0; i < hashsize; i++)
95          hashtable[i] = NULL;
96  }
97
98  void insert(hashdata *array) {
99      int rec;
100     rec = hashfunction(array);
101     hashtable[rec] = array;
102     cout << "Input Successfully" << endl;
103     cout << "Data DiSimpan pada record indeks " << rec << endl;
104     cin.get();
105 }
106
```

cnthprak13.cpp

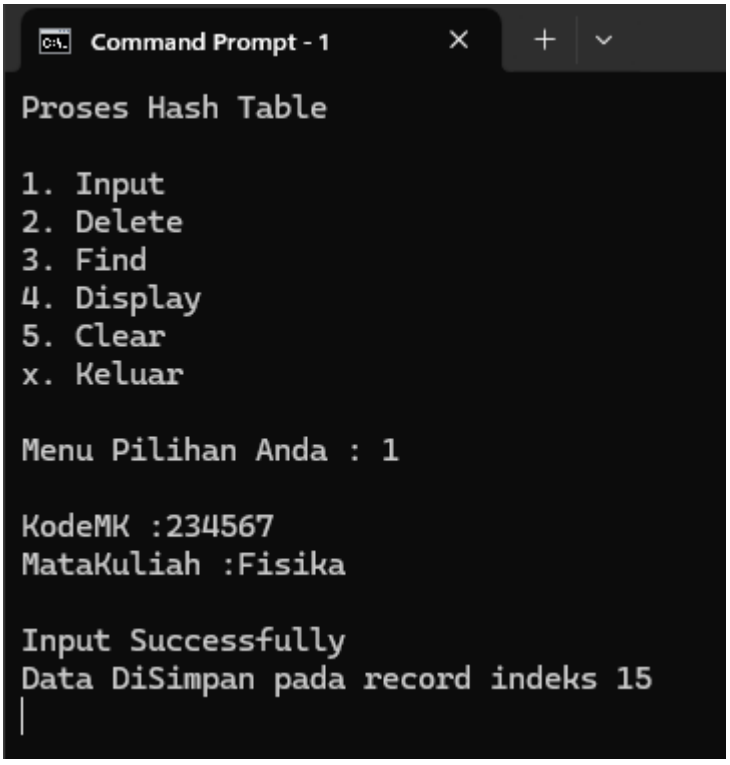
```
106
107 void searching(hashdata *cari) {
108     int rec;
109     rec = hashfunction(cari);
110     rec = searchingprocess(cari, rec);
111     if (rec >= 0) {
112         cout << "Data ditemukan pada record indeks " << rec << endl;
113         cout << "Isi data :" << endl;
114         cout << "=> KodeMK :" << hashtable[rec]->kode << endl;
115         cout << "=> MataKuliah :" << hashtable[rec]->nama << endl;
116     } else
117         cout << "Maaf Data tidak ditemukan." << endl;
118     cin.get();
119 }
120
121 int searchingprocess(hashdata *cari, int rec) {
122     int m, n, j = 0;
123     bool equal = true;
124     if (isindexempty(rec) == false) {
125         n = characteramount(cari->kode);
126         m = characteramount(hashtable[rec]->kode);
127         if (n == m) {
128             while ((equal == true) && (cari->kode[j] != '\0')) {
129                 if (tolower(cari->kode[j]) != tolower(hashtable[rec]->kode[j]))
130                     equal = false;
131                 j++;
132             }
133             if (equal == true)
134                 return rec;
135             else
136                 return -1;
137         } else return -1;
138     } else
139         return -1;
140     cin.get();
141 }
142
```

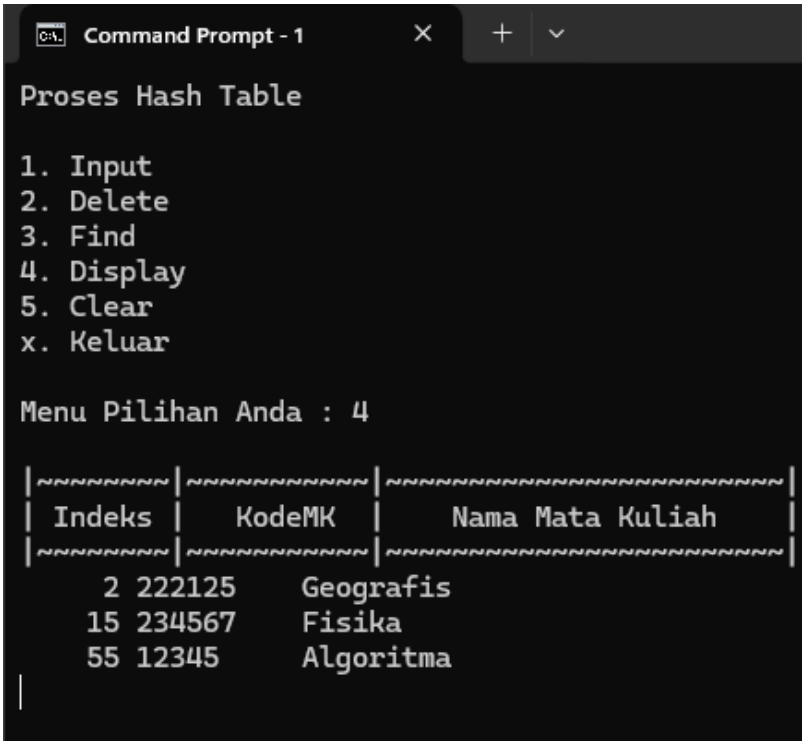
cnthprak13.cpp

```
143 void deleting(hashdata *hapus) {
144     int rec;
145     rec = hashfunction(hapus);
146     rec = searchingprocess(hapus, rec);
147     if (rec >= 0) {
148         cout << "Data :" << endl;
149         cout << "=> KodeMK :" << hashtable[rec]->kode << endl;
150         cout << "=> MataKuliah :" << hashtable[rec]->nama << endl;
151         cout << "Terhapus" << endl;
152         hashtable[rec] = NULL;
153     } else
154         cout << "Maaf Data tidak ditemukan" << endl;
155     cin.get();
156 }
157
158 int hashfunction(hashdata *array) {
159     int value = 0, rec, n;
160     n = characteramount(array->kode);
161     for (int i = 0; i < n; i++)
162         value += array->kode[i];
163     rec = value % hashsize;
164     return rec;
165 }
166
167 bool isindexempty(int rec) {
168     if (hashtable[rec] == NULL)
169         return true;
170     else
171         return false;
172 }
173
174 int characteramount(char array[]) {
175     int jumlah = 0;
176     for (int i = 0; array[i] != '\0'; i++)
177         jumlah++;
178     return jumlah;
179 }
180
181 void displaytable() {
182     cout << "\n";
183     cout << "|~~~~~~~~~~|~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~~~~|" << endl;
184     cout << "| Indeks |   KodeMK   |     Nama Mata Kuliah     |" << endl;
185     cout << "|~~~~~~~~~~|~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~~~~|" << endl;
186     for (int i = 0; i < hashsize; i++) {
187         if (isindexempty(i) == false)
188             cout << setw(6) << i << " " << hashtable[i]->kode << "\t " << hashtable[i]->nama << endl;
189     }
190     cin.get();
191 }
192
```


Command Prompt - 1
Proses Hash Table
1. Input
2. Delete
3. Find
4. Display
5. Clear
x. Keluar
Menu Pilihan Anda : 1
KodeMK :12345
MataKuliah :Algoritma
Input Successfully
Data DiSimpan pada record indeks 55


Command Prompt - 1
Proses Hash Table
1. Input
2. Delete
3. Find
4. Display
5. Clear
x. Keluar
Menu Pilihan Anda : 1
KodeMK :222125
MataKuliah :Geografis
Input Successfully
Data DiSimpan pada record indeks 2


Command Prompt - 1
Proses Hash Table
1. Input
2. Delete
3. Find
4. Display
5. Clear
x. Keluar
Menu Pilihan Anda : 1
KodeMK :234567
MataKuliah :Fisika
Input Successfully
Data DiSimpan pada record indeks 15


Command Prompt - 1
Proses Hash Table
1. Input
2. Delete
3. Find
4. Display
5. Clear
x. Keluar
Menu Pilihan Anda : 4
| Indeks | KodeMK | Nama Mata Kuliah |
2 222125 Geografis
15 234567 Fisika
55 12345 Algoritma

Pseudocode :

```
Kamus/Deklarasi Variabel fungi clear
i = int

Algoritma/Deskripsi fungsi clear
for (int i = 0; i < hashsize; i++)
      hashtable[i] = NULL
endfor

Kamus/Deklarasi Variabel fungsi insert
rec = int

Algoritma/Deskripsi fungsi insert (*array)
rec = hashfunction(array)
hashtable[rec] = array
print rec

Kamus/Deklarasi Variabel fungsi searching
rec = int

Algoritma/Dekripsi fungi searching (*cari)
rec = hashfunction(cari)
rec = searchingprocess(cari, rec)
if (rec >= 0)
rec
hashtable[rec]->kode
hashtable[rec]->nama
else
 print "Maaf Data tidak ditemukan."
endif
```

```
Kamus/Deklarasi Variabel fungsi searchingprocess
m,n,j,rec=int
equal = bool

Algoritma/Dekripsi fungsi searchingprocess(*cari,rec)
m, n, j = 0
equal = true
if (isindexempty(rec) == false)
n = characteramount(cari->kode)
m = characteramount(hashtable[rec]->kode)
if (n == m)
while ((equal == true) && (cari->kode[j] != '\0'))
if (tolower(cari->kode[j]) !=
tolower(hashtable[rec]->kode[j]))
equal = false
j++
endwhile
if (equal == true)
print rec
else
print  -1
endif
else print -1
endif
else
print -1
```

```
Kamus/Deklarasi Variabel fungsi deleting
rec = int

Algoritma/Deskripsi fungsi deleting(*hapus)
rec = hashfunction(hapus)
rec = searchingprocess(hapus, rec)
if (rec >= 0)
hashtable[rec]->kode
hashtable[rec]->nama
hashtable[rec] = NULL
else
print "Maaf Data tidak ditemukan"
endif

Kamus/Deklarasi Variabel fungsi hashfunction
value,rec,n, i =int

Algoritma/Deskripsi fungsi hashfunction(*array)
value = 0
n = characteramount(array->kode)
for (int i = 0; i < n; i++)
value += array->kode[i]
rec = value % hashsize
return rec
```

```
Kamus/Deklarasi Variabel fungsi isindexempty
rec = int

Algoritma/Deskripsi fungsi isindexempty(rec)
if (hashtable[rec] == NULL)
      print true
   else
      print false

Kamus/Deklarasi Variabel fungsi
characteramount
array[] = char
jumlah,i=int

Algoritma/Deskripsi fungi characteramount(array)
jumlah = 0
for (int i = 0; array[i] != '\0'; i++)
jumlah++
print jumlah

Kamus/Deklarasi Variabel fungsi displaytable
i =int

Algoritma/Dekripsi fungsi displaytable
for (int i = 0; i < hashsize; i++)
if (isindexempty(i) == false)
print  i ; hashtable[i]->kode ; hashtable[i]->nama
endfor
endif
```

```
Kamus/Deklarasi Variabel fungsi utama
ulang,menu,kode[kodemk],nama[namamk]

Algoritma/Dekripsi fungsi utama
hashsize = 100
kodemk = 30
namamk = 13
struct hashdata { kode[kodemk], nama[namamk]}
ulang = ‘Y’
do
system("cls");
      puts("Proses Hash Table");
      puts("");
      puts("1. Input");
      puts("2. Delete");
      puts("3. Find");
      puts("4. Display");
      puts("5. Clear");
      puts("x. Keluar");
      puts("");
      input menu
switch (menu)
         case '1':
            hashdata *array
         array = new hashdata;
            puts("")
          input array->kode
          input array->nama
            insert(array)
            break
         case '2':
            hashdata *hapus
         hapus = new hashdata;
          input hapus->kode
            deleting(hapus)
            delete hapus
            break
case '3':
            hashdata *cari
         cari = new hashdata
           input cari->kode
            searching(cari)
            delete cari
            break
         case '4':
            displaytable()
            break
         case '5':
            clear();
 puts("Tabel Hash telah dikosongkan.")
            break
         case 'x':
         case 'X':
            ulang = 'T'
            break
         default:
puts("Pilihan diluar Menu yang Tersedia");
            break
endswitch
enddo
while (toupper(ulang) == 'Y')
```

Algoritma :

1. Membuat fungsi clear
2. Selama (i=0) maka kerjakan baris 3 s.d 4
3. hashtable[i]=NULL
4. i++
5. Membuat fungsi insert (hashdata *array)
6. rec = hashfunction(array)
7. hashtable[rec] = array
8. Mencetak/Menampilkan nilai rec
9. Membuat fungsi searching(hahdata *cari)
10. rec = hashfunction(cari)
11. rec = searchingprocess(cari, rec)
12. Jika (rec >= 0) maka kerjakan bari 13 s.d 15 kalau tidak baris 16
13. Mencetak/Menampilkan nilai rec
14. Mencetak/Menampilkan nilai hashtable[rec]->kode
15. Mencetak/Menampilkan nilai hashtable[rec]->nama
16. Mencetak"Maaf Data tidak ditemukan."
17. Membuat fungsi searchingprocess(hashdata *cari, rec)
18. m,n,j = 0
19. equal = true
20. Jika (isindexempty(rec) == false) maka kerjakan baris 21 s.d 32
21. n = characteramount(cari->kode)
22. m = characteramount(hashtable[rec]->kode)
23. Jika (n==m) maka kerjakan baris 24 s.d 31
24. Selama ((equal == true) && (cari->kode[j] != '\0')) maka kerjakan baris 25 s.d 27
25. Jika (tolower(cari->kode[j]) != tolower(hashtable[rec]->kode[j]))
26. equal = false
27. j++
28. Jika (equal == true)
29. Mencetak rec
30. Mencetak -1
31. Mencetak -1
32. Mencetak -1
33. Membuat fungsi deleting(hashdata *hapuss)
34. rec = hashfunction(hapus)
35. rec = searchingprocess(hapus, rec)
36. Jika (rec >=0) maka kerjakan baris 37 s.d 39 kalau tidak baris 40
37. Mencetak/Menampilkan Nilai hashtable[rec]->kode
38. Mencetak/Menampilkan Nilai hashtable[rec]->nama
39. hashtable[rec]=NULL
40. Mencetak “Maaf Data tidak Ditemukan”
41. Membuat fungsi hashfunction (*array)
42. value = 0
43. n = characteramount(array->kode)
44. Selama (int i = 0)
45. value += array->kode[i]
46. rec = value % hashsize
47. Mencetak rec
48. i++
49. Membuat fungsi isindexempty(rec)
50. Jika (hashtable[rec] == NULL) maka kerjakan baris 51 kalau tidak 52
51. Mencetak true
52. Mencetak false
53. Membuat fungsi characteramount(array[])
54. jumlah = 0
55. Selama (i=0) maka kerjakan baris 56 s.d 58
56. jumlah++
57. Mencetak jumlah
58. i++
59. Membuat fungsi displaytable
60. Selama (i=0) maka kerjakan baris 61 s.d 63
61. Jika (isindexempty(i) == false) maka kerjakan baris 65
62. Mencetak/Menampilkan nilai i
63. Mencetak/Menampilkan nilai hashtable[i]->kode
64. Mencetak/Menampilkan nilai hashtable[i]->nama
65. i++
66. Membuat fungsi utama
67. Deklarasi struktur(struct hashdata{kode[kodemk], nama[namamk]})
68. ulang = ‘Y’
69. Memanggil fungsi clear
70. do
71. system("cls");
72. puts("Proses Hash Table");
73. puts("");
74. puts("1. Input");
75. puts("2. Delete");
76. puts("3. Find");
77. puts("4. Display");
78. puts("5. Clear");
79. puts("x. Keluar");
80. puts("");
81. Menginput/Memasukkan Nilai menu
82. switch(menu)
83. case '1':
84. hashdata *array;
85. array = new hashdata;
86. puts("")
87. Menginput/Memasukkan Nilai array->kode
88. Menginput/Memasukkan Nilai array->nama
89. Memanggil fungsi insert(array)
90. break
91. case '2':
92. hashdata *hapus
93. hapus = new hashdata
94. Menginput/Memasukkan Nilai hapus->kode
95. Memanggil fungsi deleting(hapus)
96. delete hapus
97. break

```
98. case '3':
99. hashdata *cari
100. cari = new hashdata
101. Menginput/Memasukkan Nilai cari->kode
102. Memanggil fungsi searching(cari)
103. delete cari
104. break
105. case '4':
106. displaytable();
107. break
108. case '5':
109. Memanggil fungsi clear();
110. puts("Tabel Hash telah dikosongkan.")
111. break
112. case 'x':
113. case 'X':
114. ulang = 'T'
115. break
116. default:
117. puts("Pilihan diluar Menu yang Tersedia")
118. break
119. while (toupper(ulang) == 'Y')
120. Selesai
```