データ構造と基本アルゴリズム

report 1

提出日：11 月 11 日（月）

所属：工学部情報工学科

学籍番号：135713B

氏名：天願 寛之

演習問題 3.2 を解答し、レポートとして提出しなさい。ただし、作成したプログラムの各行にコメントを入れること。プログラムの各機能が正しく実行できる事を実行例で示すこと。

作成したプログラム

```
01 #include <stdio.h>
02
03 struct vertex{
04   int name,key;
05   struct vertex *next;
06 }*top;
07
08 search(int k);
09 deletion(int k);
10
11 int main(){
12   struct vertex *new;
13   int n,k,q;
14   printf("¥n");
15   printf("::::::::: リストへのデータ入力 :::::::::  ¥n");
16   printf("¥n");
17
18   printf("各要素を入力してください¥n");
19   top = NULL;
20   while(1){
21     printf("name の値を入力してください: ");
22     scanf("%d",&n);
23     if(n < 0)break;/*n が負ならば while 文を終了*/
24     printf("key の値を入力してください: ");
25     scanf("%d",&k);
26     if(search_key(k) == 1){ /*if 文,引数 k を持つ関数 search_key の呼び出し,条件：標準入力された k の値がリスト内の
27 構造体 key に存在する,処理：以下の while 文*/
28       while(1){ /*while 文*/
29         printf("key が既登録です。key が%d の構造体をリストから削除しますか？",k); /*問い*/
30         printf("¥n");
31         printf("yes なら 1 を、no なら 0 を入力してください。"); /*指示の選択*/
32         scanf("%d",&q); /*scanf 関数,q に標準入力された値を代入*/
33         if(q == 1){deletion_key(k); break; /*if 分,条件：入力された値が 1,処理：引数 k を持つ関数 deletion_key の
34 呼び出してループを抜ける*/
35         }else if(q == 0){ /*if 文,条件：入力された値が 0,処理：以下に記述*/
36           new = (struct vertex *)malloc(sizeof(struct vertex)); /*new に malloc 関数で確保した sizeof(struct
37  vertex)サイズの領域の先頭アドレスを代入*/
38                 new->name = n; /*標準入力された n の値を new が指す name に代入*/
39                 new->key = k; /*標準入力された k の値を new が指す key に代入*/
40                 new->next = top; /*top の指すアドレスを new が指す next に代入*/
41                 top = new; /*top に new が指すアドレスを代入*/
42                 break; /*ループを抜ける*/
43         }else{ printf("%d...って馬鹿野郎///",q); printf("¥n"); /*標準入力された値が 0 と 1 以外の場合に出力され,
44 ループする*/
45         }
46       }
47     }else{ /*標準入力された k の値がリスト内の key に存在しない場合,以下の処理*/
48       new = (struct vertex *)malloc(sizeof(struct vertex)); /*new に malloc 関数で確保した sizeof(struct vertex)
49 サイズの領域の先頭アドレスを代入*/
50       new->name = n; /*標準入力された n の値を new が指す name に代入*/
51       new->key = k; /*標準入力された k の値を new が指す key に代入*/
52       new->next = top; /*top の指すアドレスを new が指す next に代入*/
53          top = new; /*top に new が指すアドレスを代入*/
54     }
55   }
56
57   printf("¥n");
```

```
58   printf("-----------------------------------¥n");
59   printf("入力したデータ([next, key]の順番でヘッドに近い構造体から出力)¥n");
60   while(top != NULL){ /*top の指す値が NULL ではない間ループする*/
61     printf("[%d, %d] ",top->name,top->key); /*name と key の値を出力*/
62     top = top->next; /*top に top が指す next の値,つまり top が指す構造体の次の構造体のアドレスを代入している*/
63   }
64   printf("¥n");
65 }
66
67 /*リスト内に key の値が k に等しい構造体が存在するか否かを探索する search_key 関数*/
68 int search_key(int k){ /*引数に k を持つ*/
69   struct vertex *p;
70   p = top;                          /*構造体ポインタ p に top の値つまり top が指す構造体の先頭アドレスを代入する    */
71   while(p!= NULL){                   /*while 文のループで p の指す構造体を 1 つずつ最後尾に向けてズラしていき        */
72     if(p->key == k)                 /*その間に key の値が k に等しい構造体に p がアクセスしたら,                    */
73       return (1);                  /*return を用いて 1 を返している                                               */
74     p = p->next;                    /*もし,key の値が k に等しい構造体に p がアクセスせず最後尾,つまり NULL を持つと*/
75   }                                /*return でもって 0 を返している                                               */
76   return (0);
77 }
78
79 /*key の値が k と等しい構造体を削除する deletion_key 関数*/
80 int deletion_key(int k){ /*引数に k を持つ*/
81   struct vertex *p1,*p2,*p3;
82   if(k == top->key){              /*リスト先頭での構造体を削除する場合,宣言した構造体ポインタ p3 に              */
83       p3 = top;                   /*top の持つ先頭の構造体のアドレスを覚えさせておき,                            */
84     top = top->next;              /*top には,top が指す next の値,つまり先頭から二番目の構造体アドレスを代入する*/
85                       free(p3);  /*結果,リスト元先頭での構造体はアクセスをなくし,リストから切り離される        */
86   }                               /*最後に free 関数を用いて切り離された構造体のメモリ領域を開放する             */
87   else {                          /*key の値が k に等しい構造体がリスト先頭以外にある場合,                       */
88       p1 = top;                   /*宣言した構造体ポインタ p1 に top の持つ先頭の構造体のアドレスを覚えさせ,     */
89     p2 = p1->next;                /*他に宣言した構造体ポインタ p2 に p1 の指す next 値,つまり p1 の持つ構造体の  */
90     while(p2!= NULL){             /*次の構造体のアドレスを代入する                                               */
91       if(p2->key == k){           /*while 文のループで p1,p2 の指す構造体を 1 つずつ最後尾に向けてズラしていき, */
92        p1->next = p2->next;       /*その間に key の値が k に等しい構造体に p2 がアクセスしたら,                  */
93         free(p2);                 /*p1 の指す next の値に p2 の指す next の値,つまり p2 がアクセスした構造体の次の*/
94         break;                    /*構造体のアドレスを代入することで p2 がアクセスしている key の値が k に等しい  */
95       }                           /*構造体が切り離される                                                         */
96       p1 = p2;                    /*最後に free 関数を用いて切り離された構造体のメモリ領域を開放する             */
97       p2 = p2->next;
98     }
99   }
100  return (0);
101 }
```

プログラムの実行例

```
::::::::: リストへのデータ入力 :::::::::

各要素を入力してください
name の値を入力してください: 11
key の値を入力してください: 1
name の値を入力してください: 12
key の値を入力してください: 2
name の値を入力してください: 13
key の値を入力してください: 3
name の値を入力してください: 14
key の値を入力してください: 1
key が既登録です。key が 1 の構造体をリストから削除しますか？
yes なら 1 を、no なら 0 を入力してください。1
name の値を入力してください: 15
key の値を入力してください: 2
key が既登録です。key が 2 の構造体をリストから削除しますか？
```

```
yes なら 1 を、no なら 0 を入力してください。0
name の値を入力してください: 16
key の値を入力してください: 3
key が既登録です。key が 3 の構造体をリストから削除しますか？
yes なら 1 を、no なら 0 を入力してください。123456789
123456789...って馬鹿野郎///
key が既登録です。key が 3 の構造体をリストから削除しますか？
yes なら 1 を、no なら 0 を入力してください。1
name の値を入力してください: 17
key の値を入力してください: 2
key が既登録です。key が 2 の構造体をリストから削除しますか？
yes なら 1 を、no なら 0 を入力してください。123456789
123456789...って馬鹿野郎///
key が既登録です。key が 2 の構造体をリストから削除しますか？
yes なら 1 を、no なら 0 を入力してください。0
name の値を入力してください: -1

------------------------------------
入力したデータ([next, key]の順番でヘッドに近い構造体から出力)
[17, 2] [15, 2] [12, 2]
```

追記

key と k が等しいときに削除するかどうかの質問に対して、0 か 1 の入力を促していますが、誤って 0 と 1 以外の値の入力が行われても正常に動作するように工夫してみました。