リストⅣ

０．目次

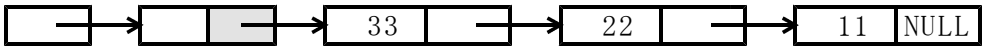
３．応用

３．１　リストの反転

リストの反転を行うプログラムを考察する。

反転前

反転後

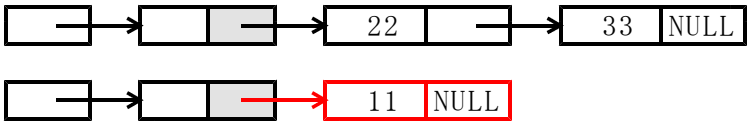


●考え方

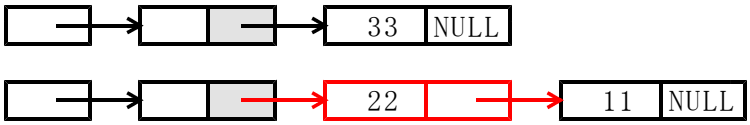
元のリストの先頭から要素を取り出し、新たなリストの先頭に挿入していく。

（１）元のリスト

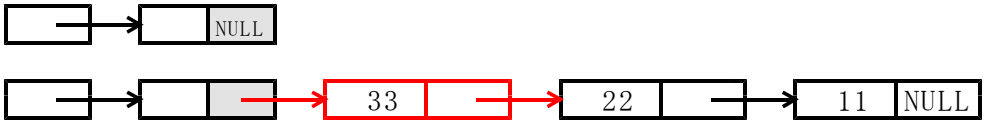
（２）要素11を取り出し、新リストの先頭に挿入する。



（３）要素22を取り出し、新リストの先頭に挿入する。

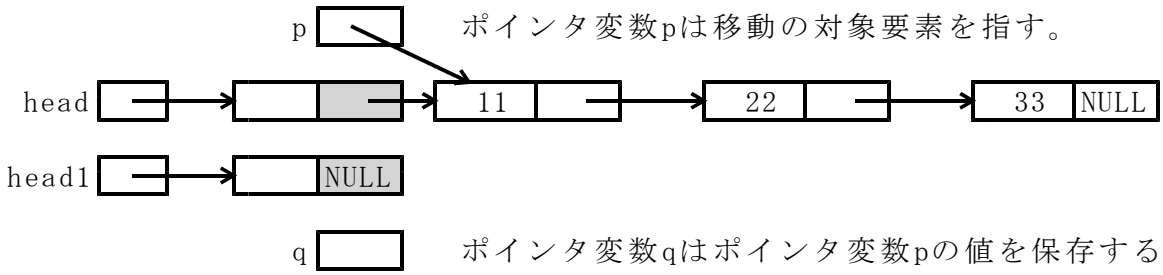


（４）要素33を取り出し、新リストの先頭に挿入する。

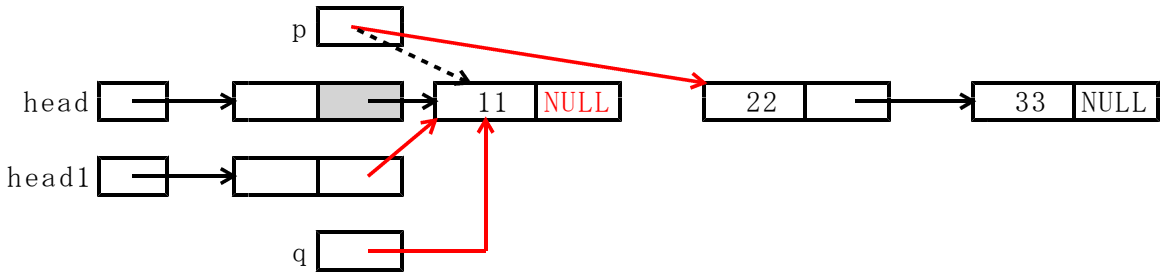


●ひとつずつ要素を移動

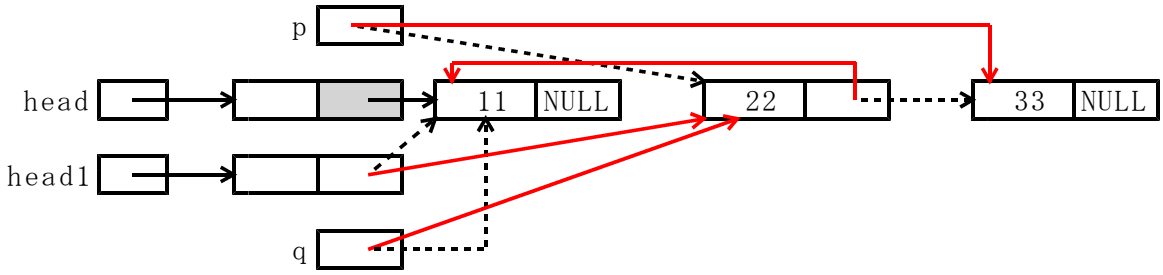
（１）移動前



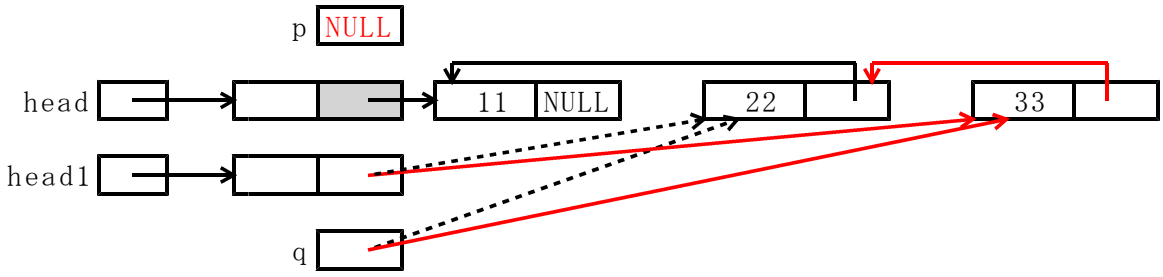
（２）要素11を取り出し、新リストの先頭に挿入する。



（３）要素22を取り出し、新リストの先頭に挿入する。



（４）要素33を取り出し、新リストの先頭に挿入する。



（５）終了

●プログラム（ls311.c）

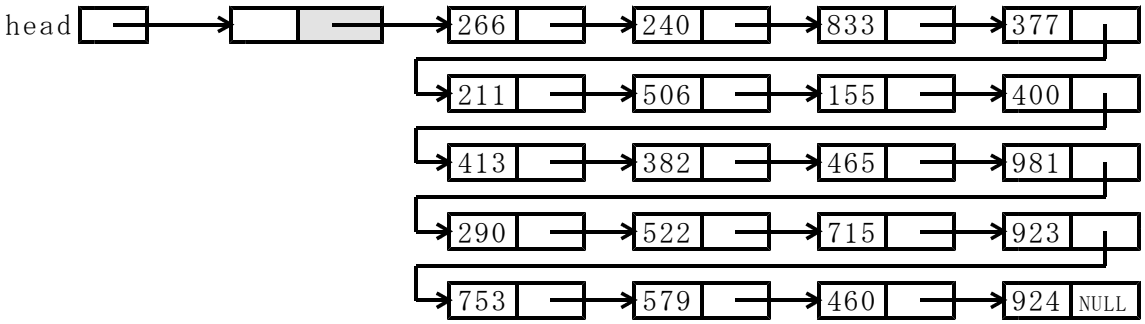
```
1  /* << ls311.c >> */
2  /* リストの反転 */
3  #include <stdio.h>
4  #include <stdlib.h>
5  struct NODE {
6    int info;
7    struct NODE *next;
8  };
9  struct NODE *mlist();
10 void slist(struct NODE *p);
11
12 int main() {
13   struct NODE *head,  /* 反転前リストの先頭を指すポインタ変数。*/
14               *head1, /* 反転後リストの先頭を指すポインタ変数。*/
15               *p,     /* 移動の対象要素を指すポインタ変数。*/
16               *q;     /* ポインタ変数pの値を保存するポインタ変数。*/
17
18   /* リストの作成と表示。*/
19   head = mlist();
20   printf("反転前リスト：");
21   slist(head);
22
23   /* リストの反転。*/
24   /* リストヘッドの作成。*/
25   head1 = (struct NODE *)malloc(sizeof(struct NODE));
26   head1->next = NULL;
27
28   p = head->next;
29
30   while( p != NULL ) {
31     q = p;
32     p = p->next;
33     q->next = head1->next;
34     head1->next = q;
35   }
36
37   /* 反転リストの表示。*/
38   printf("反転後リスト：");
39   slist(head1);
40 }
```

実行結果

```
% cc ls311.c
% ./a.out
0
反転前リスト：
反転後リスト：

% ./a.out
11 0
反転前リスト：11
反転後リスト：11

% ./a.out
11 22 0
反転前リスト：11 22
反転後リスト：22 11

% ./a.out
11 22 33 0
反転前リスト：11 22 33
反転後リスト：33 22 11
```

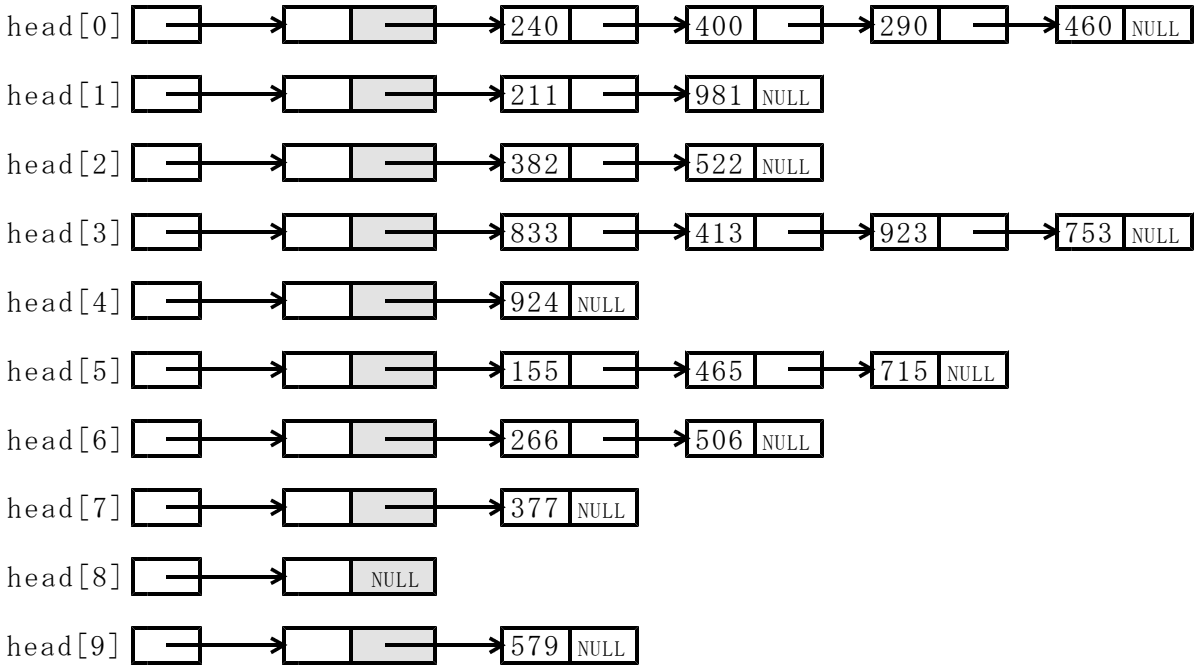
３．２　基数ソート

ひとつのリストの数値データをある桁の数字で数字別リストに分配していく。これを位の小さい桁から大きい桁まで繰り返すと、数値データが昇順に並べ替えられる。リストを使ったプログラムを作成せよ。以下のデータで実行し結果を示せ。

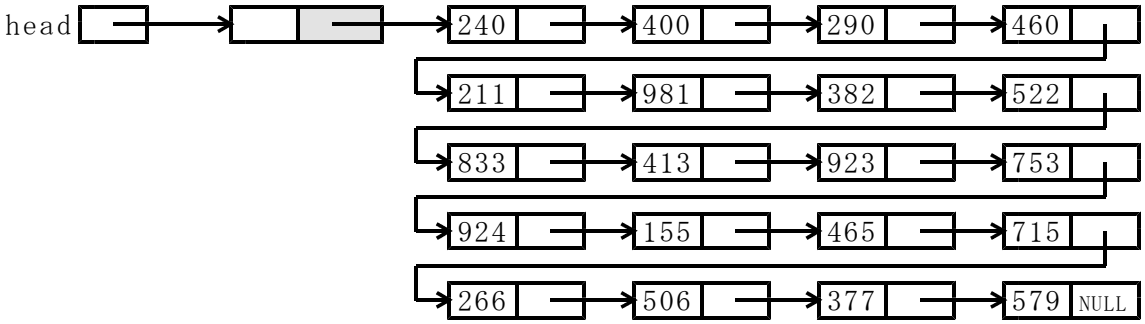


（手順１）

リストの先頭のデータから順に１の位の数字で分類していく。データは各数字のリスト末尾に追加していく。

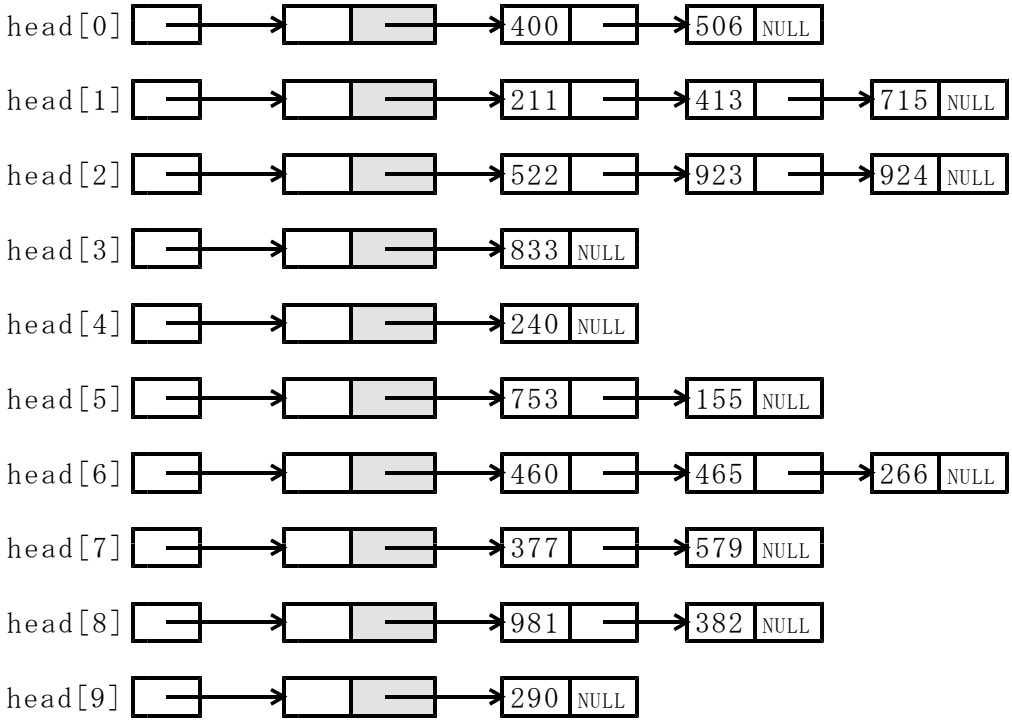


数字のリストをひとつにまとめる。

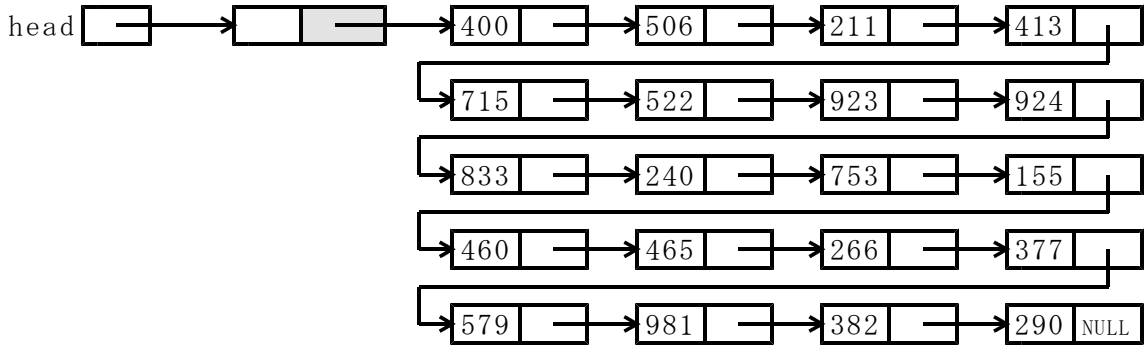


（手順２）

リストの先頭のデータから順に１０の位の数字で分類していく。データは各数字のリスト末尾に追加していく。

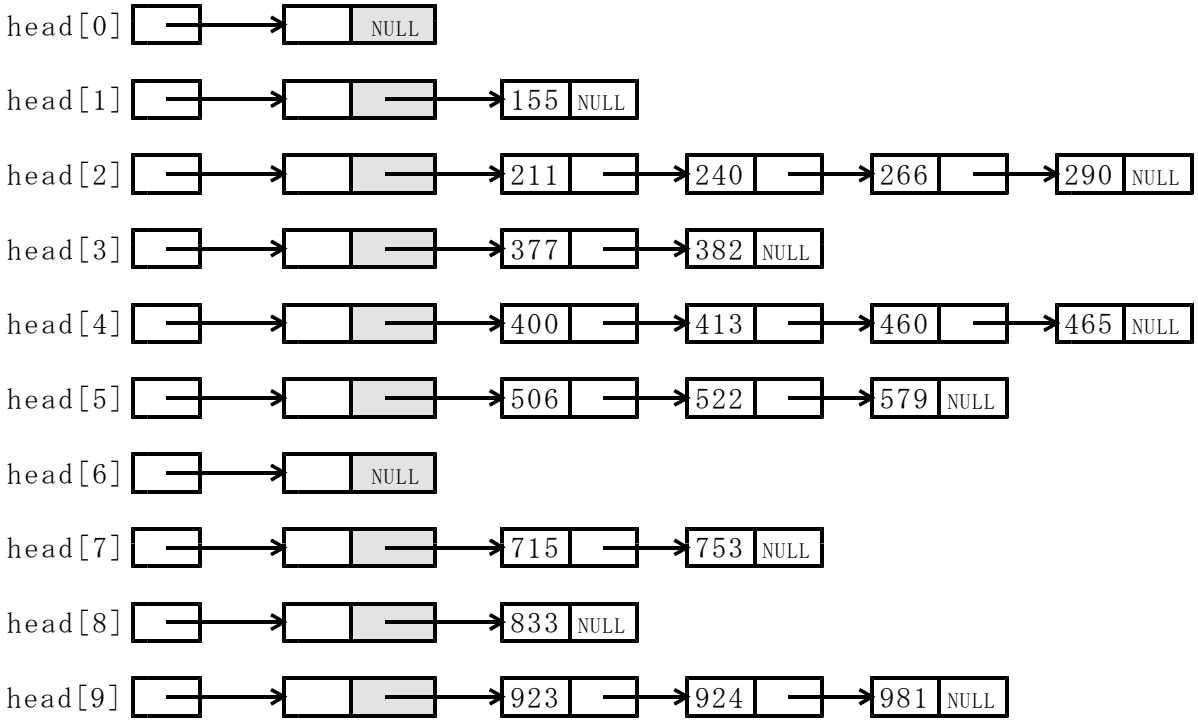


数字のリストをひとつにまとめる。

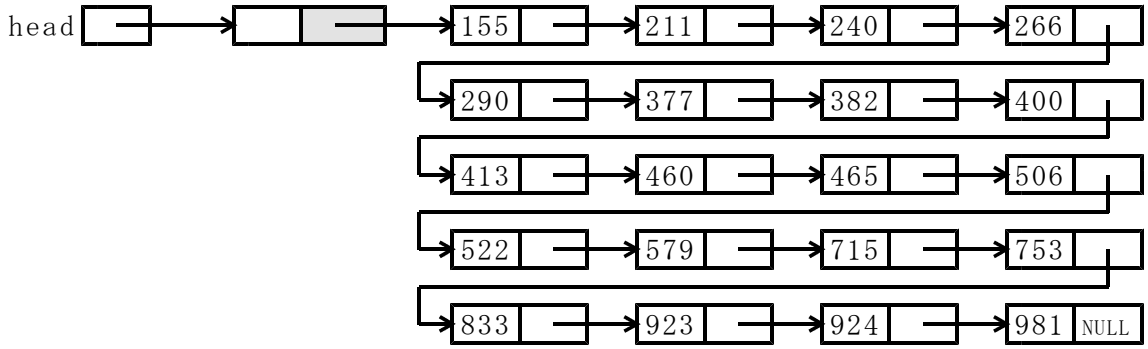


（手順３）

リストの先頭のデータから順に１００の位の数字で分類していく。データは各数字のリスト末尾に追加していく。



数字のリストをひとつにまとめる。



●プログラム（ls321.c）

```
1  /* << ls321.c >> */
2  #include <stdio.h>
3  #include <stdlib.h>
4  struct NODE {
5    int info;
6    struct NODE *next;
7  };
8  struct NODE *mlist();
9  void slist(struct NODE *p);
10
11 int main() {
12   int d,      /* 10のべき乗。*/
13       i,j,k,
14       keta,   /* 桁数。*/
15       n,      /* データの個数。*/
16       t;      /* 数字。*/
17   struct NODE *head[10], /* 数字別リストの先頭を指すポインタ変数。*/
18               *p,        /* 作業用ポインタ変数。*/
19               *tail[10], /* 数字別リストの末尾を指すポインタ変数。*/
20               *temp;     /* 作業用ポインタ変数。*/
21
22   /* ketaの入力。*/
23   scanf("%d",&keta);
24
25   /* 初期リストの作成。*/
26   temp = mlist();
27
28   /* 初期設定。*/
29   d = 1;
30
31   /* リストヘッドの作成。*/
32   for( i=0; i<=9; i++ ) {
33     head[i] = (struct NODE *)malloc(sizeof(struct NODE));
34   }
35
36   /* 並べ替え。*/
37   for( k=1; k<=keta; k++ ) {
38     /* 途中のリストの表示。 */
39     printf("途中のリスト¥n");
40     slist(temp);
41
42     /* 数字別リストの初期設定。*/
43     for( i=0; i<=9; i++ ) {
44       head[i]->next = NULL;
45       tail[i] = head[i];
46     }
47
48     /* 数字別リストの作成。*/
```

```
49      d = d*10;
50      p = temp->next;
51
52      while( p != NULL ) {
53        /* 整数の下からk桁目の数字を抽出する。*/
54        t = (p->info)%d; t = t/(d/10);
55
56        /* 数字に対応するリストに分配する。*/
57        tail[t]->next = p;
58        tail[t] = p;
59        p = p->next;
60        tail[t]->next = NULL;
61      }
62
63      /* 途中の状況を表示。*/
64      printf("数字別リスト¥n");
65      for(i=0; i<=9; i++ ) {
66        printf("[%d] ",i);
67        slist(head[i]);
68      }
69
70      /*数字別リストを一つにまとめる。*/
71      i = 0;
72      while( head[i]->next == NULL ) { i++; }
73      temp->next = head[i]->next;
74      for( j=i+1; j<=9; j++ ) {
75        if( head[j]->next != NULL ) {
76          tail[i]->next = head[j]->next;
77          i = j;
78        }
79      }
80    }
81
82    /* ソート後。*/
83    printf("ソート後¥n");
84    slist(temp);
85  }
```

実行結果

```
% cat ls321.dat
3
266 240 833 377 211 506 155 400 413 382 465 981 290 522
715 923 753 579 460 924
0

% cc ls321.c
% ./a.out < ls331.dat
途中のリスト
266 240 833 377 211 506 155 400 413 382 465 981 290 522 715
```

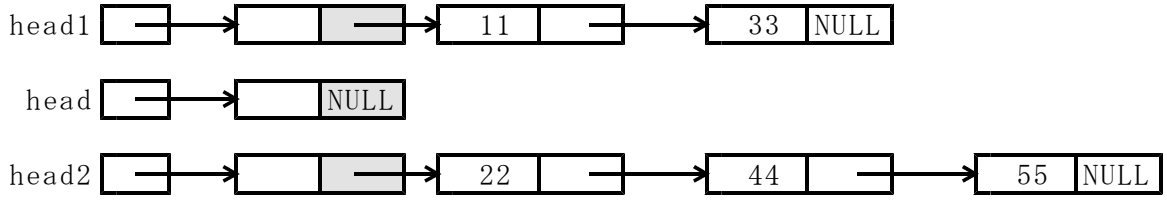
```
923 753 579 460 924
数字別リスト
[0] 240 400 290 460
[1] 211 981
[2] 382 522
[3] 833 413 923 753
[4] 924
[5] 155 465 715
[6] 266 506
[7] 377
[8]
[9] 579

途中のリスト
240 400 290 460 211 981 382 522 833 413 923 753 924 155 465
715 266 506 377 579
数字別リスト
[0] 400 506
[1] 211 413 715
[2] 522 923 924
[3] 833
[4] 240
[5] 753 155
[6] 460 465 266
[7] 377 579
[8] 981 382
[9] 290

途中のリスト
400 506 211 413 715 522 923 924 833 240 753 155 460 465 266
377 579 981 382 290
数字別リスト
[0]
[1] 155
[2] 211 240 266 290
[3] 377 382
[4] 400 413 460 465
[5] 506 522 579
[6]
[7] 715 753
[8] 833
[9] 923 924 981

ソート後
155 211 240 266 290 377 382 400 413 460 465 506 522 579 715
753 833 923 924 981
```

３．３　問題

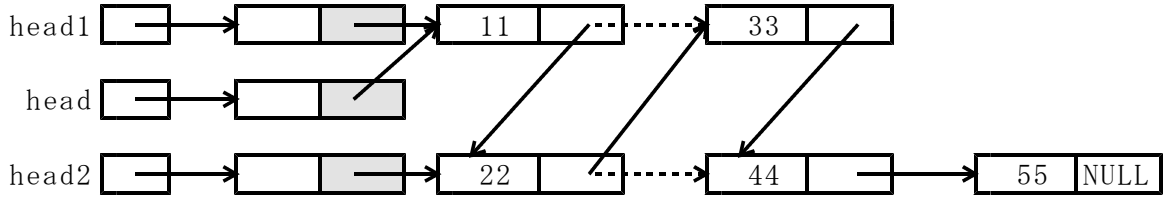
問題１　リストの併合

２つの昇順のリストを併合し、１つの昇順のリストとする。

（ａ）併合前。

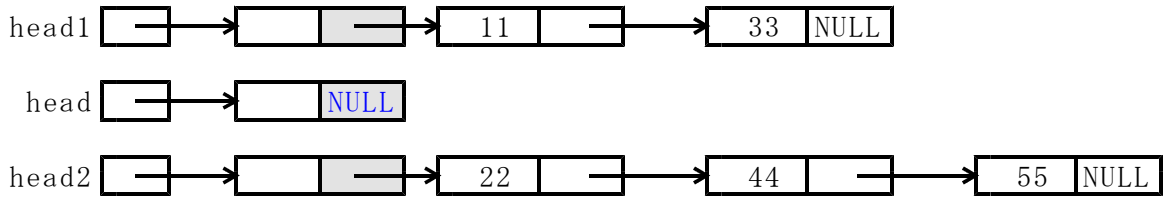


（ｂ）併合後。

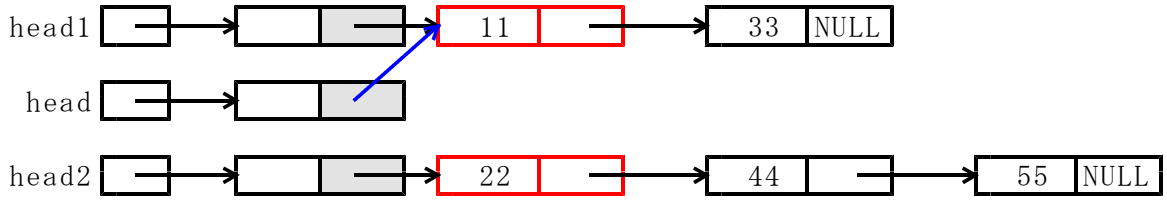


●考え方

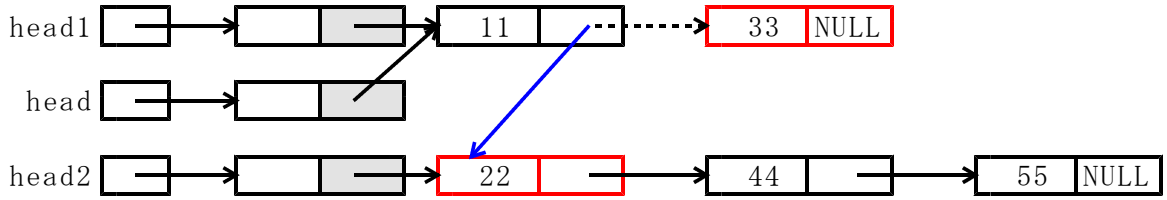
（１）リストヘッドを作成。



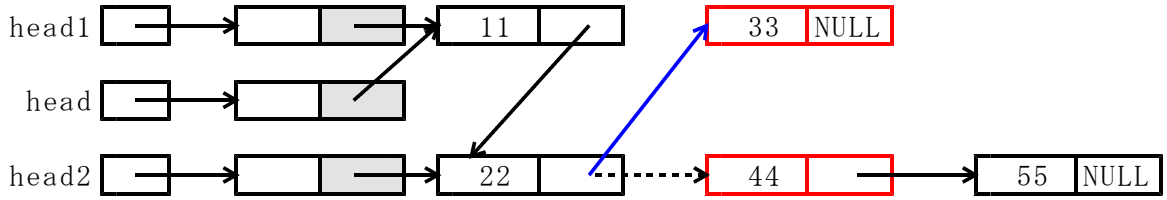
（２）要素11を併合



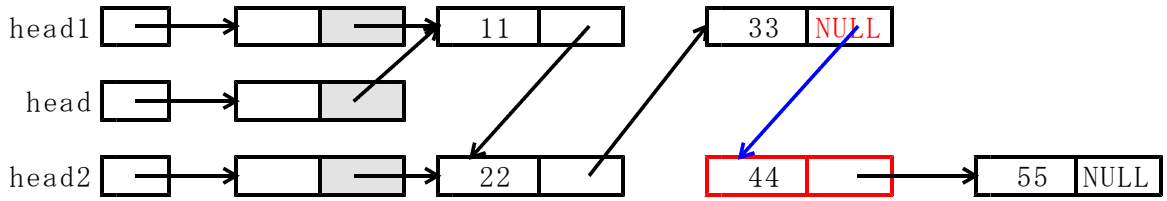
（３）要素22を併合



（４）要素33を併合



（５）要素44を併合



●プログラム(ls331.c)

```
1  /* << ls331.c >> */
2  /* ２つのリストの併合。
3     昇順のリストを併合し、昇順のリストを作成する。*/
4  #include <stdio.h>
5  #include <stdlib.h>
6  struct NODE {
7    int info;
8    struct NODE *next;
9  };
10 struct NODE *mlist();
11 void slist(struct NODE *head);
12
13 int main() {
14   struct NODE *head, /* 併合リストの先頭を指すポインタ変数。*/
15               *head1,/* リスト１の先頭を指すポインタ変数。*/
16               *head2,/* リスト２の先頭を指すポインタ変数。*/
17               *p1,   /* リスト１の要素を先頭からたどるポインタ変数。*/
18               *p2,   /* リスト２の要素を先頭からたどるポインタ変数。*/
19               *q;    /* 併合リストの末尾を指すポインタ変数。*/
20
21   /* 昇順のリスト１の作成と表示。*/
22   head1 = mlist();
23   printf("昇順のリスト１：");
24   slist(head1);
25
26   /* 昇順のリスト２の作成と表示。*/
27   head2 = mlist();
28   printf("昇順のリスト２：");
29   slist(head2);
30
31   /* 併合リストのリストヘッド作成。*/
32   head = (struct NODE *)malloc(sizeof(struct NODE));
33   head->next = NULL;
34
35   /* 初期設定。*/
36   p1 = head1->next;
37   p2 = head2->next;
38
39   /* リスト１が空の場合。*/
40   if( p1 == NULL ) {
41     head->next = ①      ;
42     exit;
43   }
44
45   /* リスト１が空でなく、リスト２が空の場合。*/
46   if( p2 == NULL ) {
47     head->next = ②      ;
48     exit;
```

```
49    }
50
51    /* リスト１もリスト２が空でない場合。*/
52    q = head;
53    while( (p1 != NULL)&&(p2 != NULL) ) {
54      if( p1->info <= p2->info ) {
55        /* リスト１の先頭を併合リストに追加。*/
56        q->next = ③      ;
57        q = ④      ;
58        p1 = ⑤           ;
59      } else {
60        /* リスト２の先頭を併合リストに追加。*/
61        q->next = p2;
62        q = p2;
63        p2 = p2->next;
64      }
65    }
66
67    /* リスト１の残りがあれば、併合リストに追加。*/
68    if( p1 != NULL ) { q->next =  ⑥      ; }
69    /* リスト２の残りがあれば、併合リストに追加。*/
70    if( p2 != NULL ) { q->next = p2; }
71
72    /* 併合リストの表示。*/
73    printf("併合リスト：");
74    slist(head);
75  }
```

実行結果

```
% cc ls331.c
% ./a.out
0
昇順のリスト１：
0
昇順のリスト２：
併合リスト：

% ./a.out
11 33 0
昇順のリスト１：11 33
0
昇順のリスト２：
併合リスト：11 33

% ./a.out
0
昇順のリスト１：
22 44 55 0
昇順のリスト２：22 44 55
併合リスト：22 44 55

% ./a.out
11 33 0
昇順のリスト１：11 33
22 44 55 0
昇順のリスト２：22 44 55
併合リスト：11 22 33 44 55
```