プログラミング実習 II レポート課題第 8 回

西田直人

2019 年 1 月 28 日

1　課題 8-1

1.1　source

Listing 1　a8-1.c

```
#include <stdio.h>
#include <stdlib.h>
#include "header1.h"

int main(){
  char com[10];
  LinkedList list;

  ListInit(&list);


  while(1){
    printf("\n>");

    scanf("%s", com);

    if(0 == strcmp(com, "display"))
      {
        PrintData(&list);
      }
    else if(0 == strcmp(com, "add"))
      {
        int no;
        char name[256];
        if( 2 != scanf("%d %s", &no, name)){
          printf("Input correctly mother fxxker\n");
          exit(1);
        }

        AddData(&list, CreateNode(no, name));
        printf("add: no: %d, name: %s\n", no, name);
      }
    else if(0 == strcmp(com, "del"))
      {
        DeleteData(&list);
      }
    else if(0 == strcmp(com, "clear"))
      {
        ClearData(&list);
      }
    else if(0 == strcmp(com, "find"))
      {
        char name[256];
        if( 1 != scanf("%s", name)){
```

```
          printf("Input correctly mother fxxker!\n");
          exit(1);
        }
        FindData(&list, name);
      }
    else if(0 == strcmp(com, "save"))
      {
        char filename[256];
        if( 1 != scanf("%s", filename)){
          printf("Input correctly mother fxxker!\n");
          exit(1);
        }
        SaveData(&list, filename);
      }
    else if(0 == strcmp(com, "load"))
      {
        char filename[256];
        if( 1 != scanf("%s", filename)){
          printf("Input correctly mother fxxker!\n");
          exit(1);
        }
        LoadData(&list, filename);
      }
    else if(0 == strcmp(com, "bye"))
      {
        printf("bye!\n");
        return 0;
      }
    else{
      printf("Input a correct command!\n");
      exit(1);
    }
  }

}
```

Listing 2　8-1Functions.c

```
#include <stdio.h>
#include <stdlib.h>
#include "header1.h"

void ListInit(LinkedList *list)
{
    list->head = NULL;   /* リストの先頭を NULL に */
    list->tail = NULL;   /* リストの末尾を NULL に */
}

Node *CreateNode(int no, const char *name)
{
    Node *n = (Node *)malloc(sizeof(Node));

    if (n == NULL)
    {
        fprintf(stderr, "CreateNode: error: memory allocation failed\n");
        return NULL;
    }
```

```
    n->data.no = no;
    strcpy(n->data.name, name);

    return n;
}

void PrintData(LinkedList *list){
  Node *nowplace;
  nowplace = list->head;

  if(list->head == NULL){
    printf("(NULL)\n");
  }
  else{
    while(nowplace != NULL){

      printf("no: %d, name: %s\n", nowplace->data.no, nowplace->data.name);
      nowplace = nowplace->next;
    }
  }
}

void AddData(LinkedList *list, Node *node){

  if (list->head == NULL) /* 要素がまだないとき */
    {
      /* 要素をひとつ追加すると、それが先頭かつ末尾になる */
      list->head = node;
      node->next = NULL;
      list->tail = node;
    }
  else
    {
      list->tail->next = node;
      node->next = NULL;
      list->tail = node;
    }

}

void DeleteData(LinkedList *list){
  Node *nowplace;
  Node *nextplace;

  if(list->head == NULL){
    printfいやリストの要素があらへんがな("\n");
  }
  else if(list->head->next == NULL){
    free(list->head);
    list->tail = NULL;
    list->head = NULL;
  }
  else{
    nextplace = list->head->next;
    nowplace = list->head;

    while(nextplace->next != NULL){
```

```
      nowplace = nextplace;
      nextplace = nextplace->next;
    }

    list->tail = nowplace;
    free(nextplace);
    list->tail->next = NULL;
  }
  printf("del: tail deleted\n");
}

void ClearData(LinkedList *list)
{
    Node *p = list->head;

    while ( p != NULL )
    {
        Node *tmp = p;  /* 現在のノードへのポインタを保存しておく */
        p = p->next; /* p を先に進める…進める前に p を解放するとリストを辿るためのポインタも失われる */
        free(tmp);      /* 現在のノードのメモリを解放する */
    }

    /* リストを初期化して、次の利用に備える */
    ListInit(list);

    printf("clear: list cleared\n");
}

void FindData(LinkedList *list, char *name){
  Node *nowplace;
  int count =0;

  if(list->head == NULL){
    printf("\"%s\" not found\n", name);
  }
  else{
    nowplace = list->head;

    while(nowplace->next != NULL){
      if(strcmp(nowplace->data.name, name) == 0){
        printf("no: %d, name: %s\n", nowplace->data.no, nowplace->data.name);
        count++;
      }
      nowplace = nowplace->next;
    }
    if(count == 0){
      printf("\"%s\" not found\n", name);
    }
  }
}

void SaveData(LinkedList *list, char *filename){
  FILE *fp;
  Node *nowplace;

  if ((fp = fopen(filename, "w")) == NULL) {
    printf("file open error!!\n");
```

```
    exit(1);
  }

  nowplace = list->head;

  while(nowplace != NULL){
    fprintf(fp, "no: %d, name: %s\n", nowplace->data.no, nowplace->data.name);
    nowplace = nowplace->next;
  }

  printf("list saved in \"%s\"\n", filename);

  fclose(fp);

}

void LoadData(LinkedList *list, char *filename){
  FILE *fp;
  Node *nowplace;
  int N;
  char NAME[256];

  if ((fp = fopen(filename, "r")) == NULL) {
    printf("file open error!!\n");
    exit(1);
  }


  while(fscanf(fp, "no: %d, name: %s\n", &N, NAME) != EOF){
    if( list->head == NULL){
      AddData(list, CreateNode(N, NAME));
      nowplace = list->head;
    }
    else{

      AddData(list, CreateNode(N, NAME));
      nowplace = nowplace->next;
    }

    NAME[0] = '\0'; 初期化してる//
  }

  AddData(list, CreateNode(N, NAME));

  nowplace->next = NULL;
  list->tail = nowplace;

  printf("list loaded out of \"%s\"\n", filename);

  fclose(fp);
}
```

Listing 3　Makefile1

```
a8-1: a8-1.o 8-1Functions.o
        cc -o a8-1 a8-1.o 8-1Functions.o
8-1Functions.o: 8-1Functions.c header1.h
        cc -c 8-1Functions.c header1.h
```

```
a8-1.o: a8-1.c header1.h
        cc -c a8-1.c header1.h
run:
        ./a8-1
```

Listing 4　header1.h

```
#ifndef HEAD
#define HEAD

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct {
  char name[32];
  int no;
} Data;

typedef struct _node{
  Data data;
  struct _node *next;
} Node;

typedef struct {
  Node *head;
  Node *tail;
} LinkedList;
/*
enum command{
  display,
  add,
  del,
  clear,
  find,
  save,
  load,
  bye
  };*/

void PrintData(LinkedList *list);

void AddData(LinkedList *list, Node *node);

void DeleteData(LinkedList *list);

void ClearData(LinkedList *list);

void FindData(LinkedList *list, char *name);

void SaveData(LinkedList *list, char *filename);

void LoadData(LinkedList *list, char *filename);

void ListInit(LinkedList *list);

Node *CreateNode(int no, const char *name);
```

```
#endif
```

1.2 result

```
s1811433@LC2RR-P009:~/prog2/08/08kadai$ make -f Makefile1
cc -c 8-1Functions.c header1.h
cc -o a8-1 a8-1.o 8-1Functions.o
s1811433@LC2RR-P009:~/prog2/08/08kadai$ make -f Makefile1 run
./a8-1

>add 100 Sato
add: no: 100, name: Sato

>add 200 Yamamoto
add: no: 200, name: Yamamoto

>add 300 Imai
add: no: 300, name: Imai

>find Yamamoto
no: 200, name: Yamamoto

>find Kanamori
``Kanamori'' not found

>del
del: tail deleted

>display
no: 100, name: Sato
no: 200, name: Yamamoto

>save hoge.txt
list saved in ``hoge.txt''

>clear
clear: list cleared

>display
(NULL)

>load hoge.txt
list loaded out of ``hoge.txt''

>display
no: 100, name: Sato
no: 200, name: Yamamoto

>add 400 Kato
add: no: 400, name: Kato

>display
no: 100, name: Sato
no: 200, name: Yamamoto
no: 400, name: Kato
```

```
  >bye
  bye!
  s1811433@LC2RR-P009:~/prog2/08/08kadai$
```

2　課題 8-2

2.1　source

```
構造体
  //Point,の定義ColoredCurve

  typedef struct _point {
    int xi, yi;
    struct _point *next;
  } Point;


  typedef struct {
    Point *head;
    Point *tail;
    unsigned char r, g, b;
  } ColoredCurve;
```

Listing 5　ColoredCurve.c

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include "GL/glut.h"
#include "ColoredCurve.h"

Point *CreatePoint(int xi, int yi){
  Point *n;

  n = (Point *)malloc(sizeof(Point));
  n->xi = xi;
  n->yi = yi;

  return n;
}

/* 折れ線の初期化…リスト構造の初期化と同様 */
void CurveInit(ColoredCurve *curve){
  curve->head = NULL;
  curve->tail = NULL;
  curve->r = 0;
  curve->g = 0;
  curve->b = 0;
}

/* 折れ線をリスト構造として見たとき、要素つまり頂点 () が何もなければ、そうでなければ 1 0 */
int CurveIsEmpty(ColoredCurve *curve){
  if(curve->head == NULL){
```

```
    return 1;
  }
  else{
    return 0;
  }
}

/* 折れ線に色を設定 */
void CurveSetColor(ColoredCurve *curve, unsigned char r, unsigned char g, unsigned char
    b){
  curve->r = r;
  curve->g = g;
  curve->b = b;
}

/* 折れ線を表示…各頂点も DrawCircle 関数を使って表示する */
void CurveDraw(ColoredCurve *curve){
  Point *nowplace;
  unsigned char r, g, b;

  if(CurveIsEmpty(curve)){
    return;
  }

  nowplace = curve->head;

  CurveSetColor(curve, curve->r, curve->g, curve->b);

  glColor3ub(curve->r, curve->g, curve->b);   // 適当な色を指定…頂点ごとに色を指定することもできる

  /* 折れ線の頂点は glBegin と glEnd の間に並べて書く */
  glBegin(GL_LINE_STRIP); // 折れ線の描画の開始
  while(nowplace != NULL){
    glVertex2i(nowplace->xi, nowplace->yi);
    nowplace = nowplace->next;        // 折れ線の頂点の座標を指定
  }
  glEnd();        // 折れ線の描画の終了

  nowplace = curve->head;
  /* DrawCircle は glBegin() ... glEnd() の間に入れてはいけない */
  while(nowplace != NULL){
    DrawCircle(nowplace->xi, nowplace->yi, 5); が半径でいいのか？//5
    nowplace = nowplace->next;
  }

}

/* 折れ線の末尾に頂点 (xi, yi) を追加する */
void CurveAddPoint(ColoredCurve *curve, int xi, int yi){
  if(curve->head == NULL){
    curve->head = CreatePoint(xi, yi);
    curve->tail = curve->head;
    curve->tail->next = NULL;
  }
  else{
    curve->tail->next = CreatePoint(xi, yi);
    curve->tail = curve->tail->next;
```

```
    curve->tail->next = NULL;
  }
}

/* 座標 (xi, yi) の半径 radius 以内に頂点があれば、その頂点を折れ線から削除する */
/* 返り値は、頂点が見つかって削除されれば 1 、そうでなければ 0 */
int CurveErasePoint(ColoredCurve *curve, int xi, int yi, int radius){
  Point *nowplace;

  nowplace = curve->head;
  while(nowplace != NULL){
    if( pow((nowplace->xi - xi), 2) + pow((nowplace->yi - yi), 2) <= pow(radius, 2)){
      break;
    }
    nowplace = nowplace->next;
  }

  if(nowplace == curve->head){
    Point *tmp;
    tmp = curve->head;
    curve->head = curve->head->next;
    free(tmp);
    return 1;
  }
  else if(nowplace == curve->tail){
    Point *nextplace;
    nextplace = curve->head->next;
    nowplace = curve->head;

    while(nextplace->next != NULL){
      nowplace = nextplace;
      nextplace = nextplace->next;
    }

    free(nextplace);
    nowplace->next = NULL;
    curve->tail = nowplace;

    return 1;
  }
  else if(nowplace == NULL){
    return 0;
  }
  else{
    Point *beforeplace;
    beforeplace = curve->head;

    while(beforeplace->next != nowplace){
      beforeplace = beforeplace->next;
    }
    beforeplace->next = nowplace->next;
    free(nowplace);
    return 1;
  }
}

/* 折れ線の中で確保されたメモリを解放する */
```

```
void CurveClear(ColoredCurve *curve){
  Point *p = curve->head;

  while ( p != NULL )
  {
    Point *tmp = p;  /* 現在のノードへのポインタを保存しておく */
    p = p->next; /* p を先に進める…進める前に p を解放するとリストを辿るためのポインタも失われる */
    free(tmp);       /* 現在のノードのメモリを解放する */
  }

  /* リストを初期化して、次の利用に備える */
  CurveInit(curve);


}

/* 配列 curves[] に格納された n 本の折れ線のデータを、ファイル名 filename のファイルに書き込む */
/* 返り値は、ファイルへの書き込みに成功すれば 1 、失敗すれば 0 */
int CurveSaveFile(const char *filename, int n, ColoredCurve curves[]){
  FILE *fp;
  int i;
  Point *nowplace;

  if ((fp = fopen(filename, "w")) == NULL) {
    printf("file open error!!\n");
    return 0;
  }

  fprintf(fp, "%d\n", n);

  for(i=0; i<n; i++){
    fprintf(fp, "%d:: (%d, %d, %d)\n", i, curves[i].r, curves[i].g, curves[i].b);
    nowplace = curves[i].head;

    while(nowplace != NULL){
      fprintf(fp, "%d:: (%d, %d)\n", i, nowplace->xi, nowplace->yi);
      nowplace = nowplace->next;
    }
    fprintf(fp, "%d:: (-1, -1)\n", i);
  }

  fclose(fp);

  return 1;

}

/* ファイル名 filename のファイルから、配列 curves[] に折れ線のデータを読み込む。n は折れ線の本数 */
/* 返り値は、ファイルからの読み込みに成功すれば 1 、失敗すれば 0 */
int CurveLoadFile(const char *filename, int *n, ColoredCurve curves[]){
  FILE *fp;
  Point *nowplace;
  Point *beforeplace;
  int i=0, gomi, X, Y;

  if ((fp = fopen(filename, "r")) == NULL) {
    printf("file open error!!\n");
```

```
    return 0;
  }

  fscanf(fp, "%d\n", n);

  for(i=0; i<(*n); i++){
    fscanf(fp, "%d:: (%hhu, %hhu, %hhu)\n", &gomi, &(curves[i].r), &(curves[i].g), &(
        curves[i].b));


    while(fscanf(fp, "%d:: (%d, %d)\n", &gomi, &X, &Y) != EOF){
      if((X != -1)  && (curves[i].head != NULL)){
        CurveAddPoint(&curves[i], X, Y);
        nowplace = nowplace->next;
      }
      else if(curves[i].head == NULL){
        CurveAddPoint(&curves[i], X, Y);
        nowplace = curves[i].head;
      }
      else{
        break;
      }
    }

    nowplace->next = NULL;
    curves[i].tail = nowplace;
  }


  fclose(fp);
  return 1;
}
```

Listing 6　Makefile2

```
curve: curve.o ColoredCurve.o
        cc -o curve curve.o ColoredCurve.o -L. -I. -lglut -lGL -lGLU -lXi -lXrandr -lm
ColoredCurve.o: ColoredCurve.c glut.h ColoredCurve.h
        cc -c ColoredCurve.c ColoredCurve.h -L. -I. -lglut -lGL -lGLU -lXi -lXrandr -lm
curve.o: curve.c ColoredCurve.h glut.h
        cc -c curve.c ColoredCurve.h glut.h -L. -I. -lglut -lGL -lGLU -lXi -lXrandr -lm
run:
        ./curve
```

2.2　result

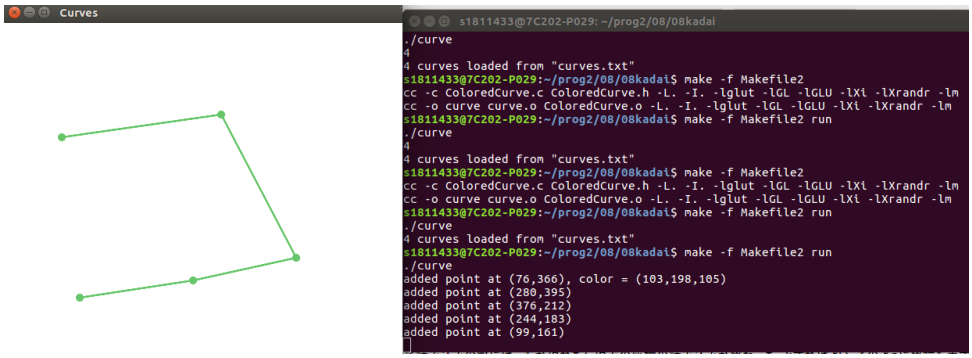

図 1　クリックして折れ線の頂点を追加

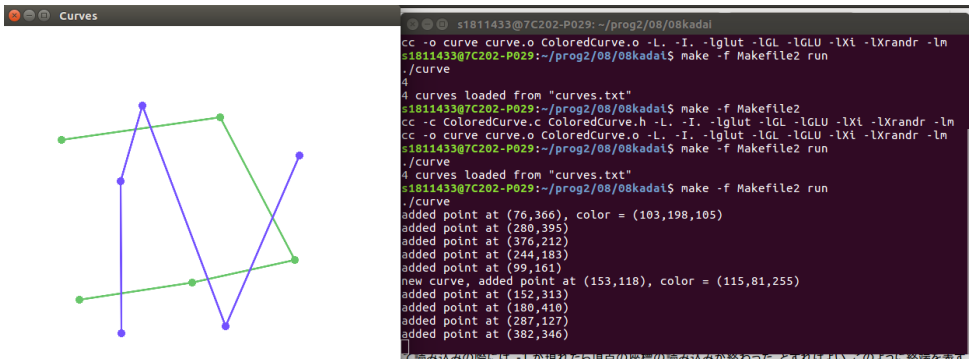

図 2　Ctrl キー + クリックで新しい折れ線を開始

.

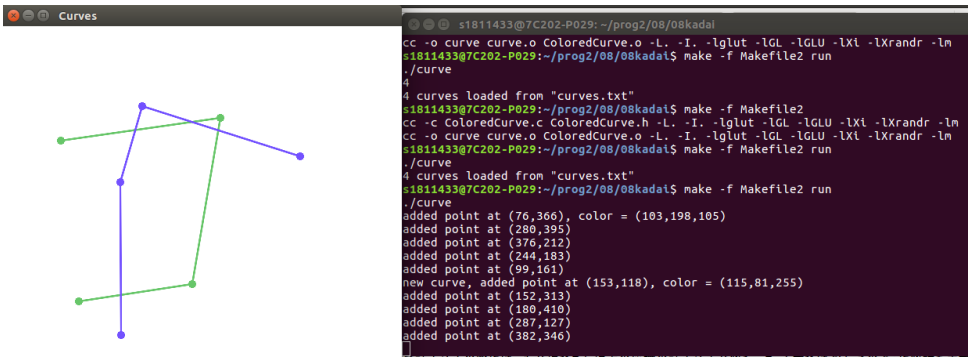

図 3　Shift キー + クリックで頂点を削除

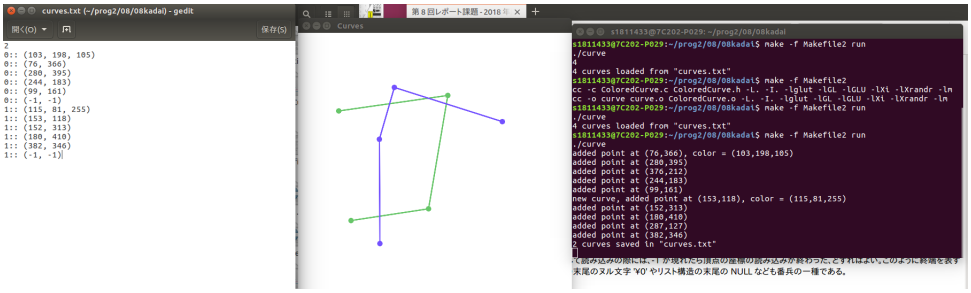

図 4　s キーでファイルにデータを保存

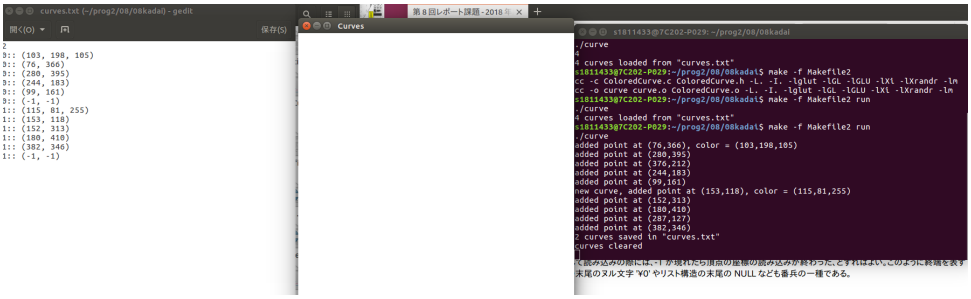

図 5　スペースキーで描画を消去

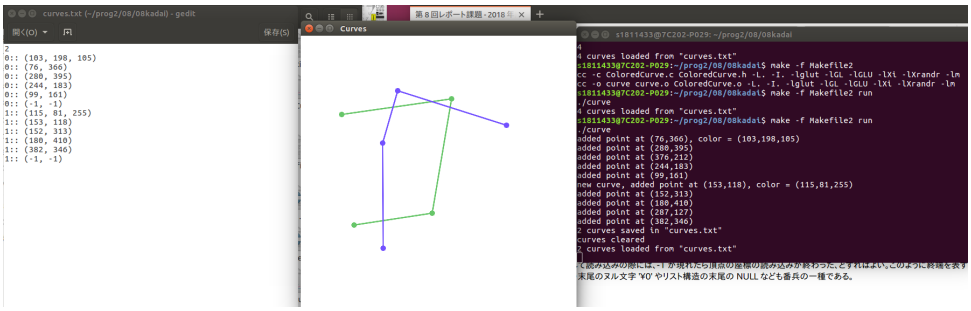

図 6　l キーでファイルからデータを呼び出して描画

3 課題 8-3

3.1 source

this is a program for showing the result of some test data of everyone. I used 2-way structure.

Listing 7　a8-3.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct _data{
  int no;
  char name[20];
  int score;
  struct _data *prev;
  struct _data *next;
} Data;

typedef struct {
  char expname[ 100];
  Data *head;
  Data *tail;
} Experiment;

int main(){
  int i;
  char name[5][10] ={"Atarashi", "Eisaki", "Koroyasu", "Tsukuda", "Magome"};
  int score[5] = {10, 30, 40, 20, 50};
  Data *nowplace = NULL, *tmp = NULL;
  Experiment *expe = NULL;
  char command;

  expe = (Experiment *)malloc(sizeof(Experiment));

  for(i=0; i<5; i++){
    if(i == 0){
      expe->head = (Data *)malloc(sizeof(Data));
      expe->head->prev = NULL;
      nowplace = expe->head;
    }
    else{
      nowplace->next = (Data *)malloc(sizeof(Data));
      nowplace->next->prev = nowplace;
      nowplace = nowplace->next;
    }

    nowplace->no = i+1;
    strcpy(nowplace->name, name[i]);
    nowplace->score = score[i];
    nowplace->next = NULL;
    expe->tail = nowplace;
  }

  expe->tail = nowplace;
```

```
  strcpy(expe->expname, "の中でのテストスコアの比較mast18");

  printf("Experiment name: %s\n", expe->expname);

  nowplace = expe->head;

  //printf("%d", expe->tail->score); //for debugging

  while(1){
    printf("participant no. %d\n name: %s\n score:%d\n", nowplace->no, nowplace->name,
        nowplace->score);

    printf("N: nextdata\nP:previousdata\nQ:quit this program\n\n");
    scanf("%c", &command);

    switch (command){
      case 'N':
      case 'n':
      if(nowplace != expe->tail){
        nowplace = nowplace->next;
      }
      break;
      case 'P':
      case 'p':
      if(nowplace != expe->head){
        nowplace = nowplace->prev;
      }
      break;
      case 'Q':
      case 'q':
      nowplace = expe->head;
      while(nowplace != NULL){
        tmp = nowplace;
        nowplace = nowplace->next;
        free(tmp);
      }
      free(expe);
      return 0;
      break;
      default:
      printf("Not allocated command.\n");
      break;

    }
   scanf("%*c");
 }  // i couldn't debug all of the program(such as when input "pp")



  return 0;
}
```

3.2 result

```
s1811433@ubuntu:~/prog2/08/08kadai$ cc -o a8-3 a8-3.c
s1811433@ubuntu:~/prog2/08/08kadai$ ./a8-3
Experiment name: の中でのテストスコアの比較mast18
participant no. 1
name: Atarashi
score:10
N: nextdata
P:previousdata
Q:quit this program

n
participant no. 2
name: Eisaki
score:30
N: nextdata
P:previousdata
Q:quit this program

n
participant no. 3
name: Koroyasu
score:40
N: nextdata
P:previousdata
Q:quit this program

n
participant no. 4
name: Tsukuda
score:20
N: nextdata
P:previousdata
Q:quit this program

n
participant no. 5
name: Magome
score:50
N: nextdata
P:previousdata
Q:quit this program

n
participant no. 5
name: Magome
score:50
N: nextdata
P:previousdata
Q:quit this program

p
participant no. 4
name: Tsukuda
score:20
N: nextdata
P:previousdata
Q:quit this program
```

```
p
participant no. 3
name: Koroyasu
score:40
N: nextdata
P:previousdata
Q:quit this program

p
participant no. 2
name: Eisaki
score:30
N: nextdata
P:previousdata
Q:quit this program

p
participant no. 1
name: Atarashi
score:10
N: nextdata
P:previousdata
Q:quit this program

p
participant no. 1
name: Atarashi
score:10
N: nextdata
P:previousdata
Q:quit this program

n
participant no. 2
name: Eisaki
score:30
N: nextdata
P:previousdata
Q:quit this program

p
participant no. 1
name: Atarashi
score:10
N: nextdata
P:previousdata
Q:quit this program

q
s1811433@ubuntu:~/prog2/08/08kadai$
```