2024年　7 月　12 日

まず、ホームディレクトリの下にC_PROG/14というディレクトリを作ってその中で作業するC_PROGというディレクトリが既にあれば、

```
% cd C_PROG
% mkdir 14
% cd 14
```

ついで、ここに、データファイルをダウンロード（city_pop.datとcity_pop2.dat、以前のファイルと少し違うので新たにダウンロードしてください）

```
% git clone https://github.com/NGS-maps/PracticeList
```

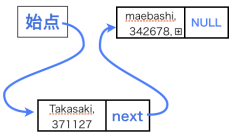
**基本課題１．** 14_1　　###部分を補い、データをリスト構造として市の名前と人口を読み込んでから出力するプログラム（mylist1.c）を作成し、（構造体宣言はヘッダファイルに分けmy_list.hとする。また新しい市の情報を記録する領域（構造体）を確保する関数 new_recordを収めたソースファイルをnew_record.cとする） makefile と ソースおよびヘッダファイルを012yamada14_1 というディレクトリに入れ tar と gzipしてできた012yamada14_1.tar.gzを山田14-1 というタイトルのメールに添付して提出。012, yamada, 山田等は適宜自分の番号や名前に置換. my_list.h, mylist1.cのファイル名は変更しなくて良い

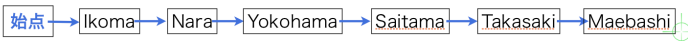
```
====================================== my_ist1.h ============================================
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "my_list.h"

typedef struct city_record                  // リスト構造で市の名前と人口を記録するための構造体宣言
 {
 char name[20];                             // 市の名前　最長20文字と仮定          20byte
 int pop;                                   // 人口                                 4byte
 struct city_record *next;                  // 次の構造体レコードを指すポインタ      8byte
 }city_record;

struct city_record *new_record(void);   // 新しいレコードの記憶領域を確保する関数new_recordのプロトタイプ宣言

=============================================================================================

====================================== mylist1.c ============================================
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "my_list.h"
int main(int argc, char *argv[])
{
city_record *head;                          // 始点のポインタ
city_record *p;                             // 現在いる場所のポインタ
FILE *infile;                               // 入力ファイルのハンドル
char input_file_name[132];
if(argc != 2)                               // アーギュメントの数が２でなければ
 {
 printf(“Wrong number of arguments¥n”);     // メッセージを吐いて
 return 1;                                  // 終わる
 }
else
 {
 strcpy(input_file_name, argv[1]);          // コマンド直後のアーギュメントを入力ファイル名として取り込む
 }

infile = fopen(input_file_name,”r”);        // 入力ファイルオープン
//////////////////////////////////////////////////////////////////////////////////////////
head = NULL;                                // 始点のポインタを空=NULLに、始めは要素無し
p = new_record();                           // 一つ目のレコードを入れる場所を確保
while(fscanf(infile,“%s %d”,p->name,&p->pop) != EOF)  //ファイルから一行ずつ市名人口を最終行まで読む
 {
 p->next = head;                            // 新しいレコードの行く先を今までの先頭（head）に
 head = p;                                  // 始点が新しいレコードを指す様にする
 p = new_record();                          // 次のレコードを入れる場所を確保
 }
//////////////////////////////////////////////////////////////////////////////////////////
p=head;                                     // 現在いる場所を始点に戻す
while(p!= NULL)                             // 現在いる場所がNULL（リストの末尾）に至るまで
 {
 printf(“%20s %10d\n”,###,###);             // 現在いるレコードの名前と人口を出力
 p = ###;                                   // 現在いるレコード p から次のレコードに移動
 }
}
=============================================================================================
```

```
--==================================== newrecord.c ============================================
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "my_list.h"

city_record *new_record(void)     // city_record構造体１つ分のメモリ領域を確保して先頭アドレスを返す
{
return ((city_record *)malloc(sizeof(city_record))); /
}
=============================================================================================
```

```
% make                          : my_list1 をコンパイルするmakefileを作っておく
% ./my_list1 city_pop.dat
          IKOMA     119258
           NARA     364738
       YOKOHAMA    3688624
        SAITAMA    1232577
       TAKASAKI     371127
       MAEBASHI     342678
```

始点に新しいレコードを挿入するやり方なので、最初のレコードが末尾に来る、逆順のリストとなる。




makefileの例を示す。gdbを使えるように-gオプションを付けた。

```
-------------------------------------------------------------
my_list1 : my_list1.o new_record.o
         gcc  -g -o my_list1 my_list1.o
my_list1.o : mylist1.c mylist.h
         gcc -g -c mylist1.c
new_record.o : new_record.c mylist.h
         gcc -g -c new_record.c
clean :
         /bin/rm -f core my_list1 *.o
-------------------------------------------------------------
```

**基本課題２．** 14_2

課題１のプログラムを修正し、最初に入力したデータが最初のレコード、最後に入力したデータが最後のレコードと順方向に並ぶプログラム mylist2.cを作成せよ。mylist1.cの(///////)行で挟まれた部分を以下で置き換え。

```
====================================== mylist2.c（部分） ======================================

city_record *tail;                                      // 今いる場所 p の前の場所 tail へのポインタ
head = new_record();                                    // 一番目のレコードを記録する場所を確保
fscanf(infile,“%s %d”,head->name,&head->pop);           // 一番目のレコードを読み込み記録
tail = head;                                          // 二番目に進むため一番目のレコードを前のレコードtailとする
p = ###;                                                // 二番目のレコードを記録する場所を確保
while(fscanf(infile,“%s %d”,p->name,&p->pop) != EOF) //二番目以降のレコードを読み込み今いる場所に記録
 {
 tail->### = p;                                         // 前のレコードが今いる場所を指す様にする
 tail = ###;                                          // 次のレコードに進むため、今いる場所pを前いた場所tailとする
 p = ###;                                               // 次のレコードを記録する場所を確保
 }
tail->next = ###;                                       // 最後の記録が入っているレコードがNULLを指す様にする
=============================================================================================
% make                                                 // mylist2をコンパイルするmakefileを作っておく
% ./mylist2 city_pop.dat
       MAEBASHI     342678
       TAKASAKI     371127
        SAITAMA    1232577
       YOKOHAMA    3688624
           NARA     364738
          IKOMA     119258
```

作成した mylist2.c と mylist.h、makefile を 012yamada14_2 というディレクトリに入れ、tar, gzip してできた 012yamada14_2.tar.gzを 山田14_2というタイトルのメールに添付して提出。my_list.h, my_list2.cの名前は変更しなくて良い。

**応用課題** 14_3

二分木の考え方を使い、読み込んだリストが、アルファベット順（辞書順）で出力される様なプログラムdlist.cとmylist.h、makefileを 012yamada14_3というディレクトリに入れ tar, gzipしてできた、012yamada14_3.tar.gzを山田14_3というタイトルのメールに添付して提出。