C Programming II
2024 Spring Final

Instructor: Po-Wen Chi

Date: 2024.06.08 PM 2:00-6:00

Policies:

- Online test.
- Do not forget to include your Makefile. TA will only use the command make to build your program. If make fails, you will get zero points and no room for bargaining. **So if you do not know how to solve a problem, please, do not include it in your Makefile**.
- I do not care your source code file names, but the executive binary names should be **fin01, fin02, fin03, fin04**.
- You can ask TA if you do not understand the problems.

1 Motion BMP (30 pts)

Motion JPEG (M-JPEG or MJPEG) is a video format in which each video frame is compressed separately as a JPEG image. Do not worry, I will not make you process JPG files since you know nothing about image compression. I just want to use this idea to build a similar format called **motion BMP**. I will give you a file that contains multiple BMP files, as shown in Fig. 1.

Your program needs to split the input file into multiple BMP files in order. Note that each BMP file may have different resolution. Since this is a video format, you need to **linearly resize** all BMP files into one resolution. If the input file is invalid, just print an error message and terminate your program.

```
$ ./fin01 --help
Usage: fin01 [options] file
  -r, --resolution=widthxheight   Setup the resolution. Default: 1024x768.
  -p, --prefix=str          Setup the file name prefix. Default: output.
  -h, --help              Display this information and exit.
$ ./fin01 -r 800x600 -p anime input.mbmp
$ ls
anime001.bmp  anime002.bmp ... anime786.bmp
```

Note that the numbers in the file names must be filled with minimum zeros, as shown in the example.

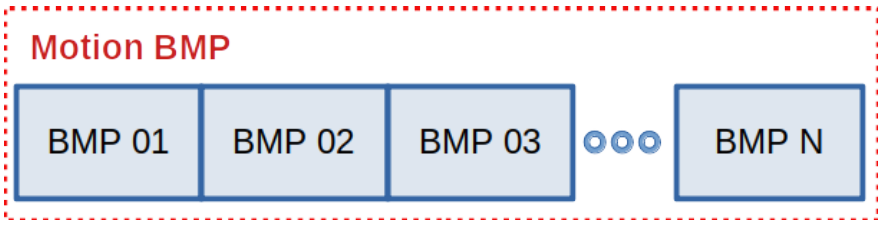

Figure 1: Motion BMP.

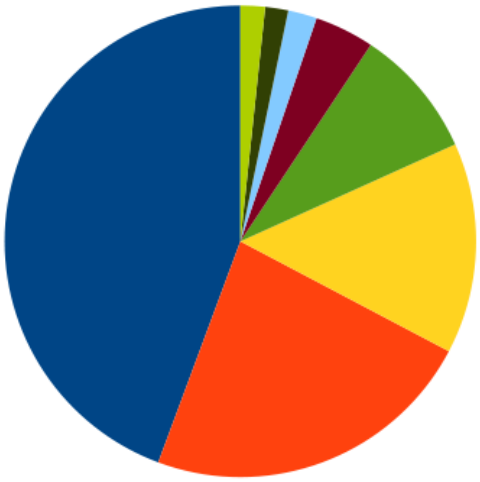

Figure 2: Pie chart for 2024.04.23 summary.

2 CPE (30 pts)

CPE, **Collegiate Programming Examination**, is a programming examination for college students. I guess most of you will take this exam in your university school life. The exam results are published on the following site.

https://cpe.cse.nsysu.edu.tw/history.php

This time, I want you to visualize the exam result with a pie chart, as shown in Fig. 2. Each color represents the percentage of student number for each solved problem count. Since there are seven problems in one exam, there should be eight colors in one pie chart[1]. The color can be defined yourself. For you own good, please use different colors or you may make our TAs annoying.

The program usage is as follows. The output file name is always **output.bmp**. The radius of pie is defined as 300 pixels. For any invalid date, just print an error message and terminate the program.

```
$ ./fin02 --help
Usage: fin02 date
  -h, --help            Display this information and exit.
$ ./fin02 2024-04-23
0: 1183 (44.34%)
1: 612 (22.94%)
```

[1] Undoubtedly, sometimes no one solve all seven problems.

ID	FirstName	LastName	Phone	Salary	Year
1	Alice	Lee	0912111222	40000	23
2	Bob	Chen	0912111223	50000	16
3	Cathy	Wang	0912111333	60000	37
4	David	Tsai	0912777888	70000	45

Figure 3: A simplified database.

```
2: 386 (14.47%)
3: 238 (8.92%)
4: 109 (4.09%)
5: 53 (1.99%)
6: 42 (1.57%)
7: 45 (1.69%)
$ ls
output.bmp
```

For your simplicity, you can assume that **libcurl** has been pre-installed on TAs' computer. The pie char precision is not an issue.

3 Database (40 pts)

You should take **Database** course taught by Dr. Koh in your college school. This time, I want you to build a database program. Just kidding. I want you to develop a very simple database with linked-list.

In this problem, please treat the database as a simple table, like Fig. 3.

How to represent this database? You should prepare two lists. The first list is used to record all labels used in this table, like "ID", "FirstName" and so on. The second list is used to store all records (rows). Each record will also have a list to record all items in the row. You can see Fig. 4 for reference.

Now you are given the following header file **database.h**. Please implement **database.c**. **linuxlist.h** is the same with the one in the example code.

```c
#pragma once

#include <stdint.h>
#include "linuxlist.h"

typedef struct _sLabel
{
  char *pStr;
  struct list_head list;
} sLabel;

```

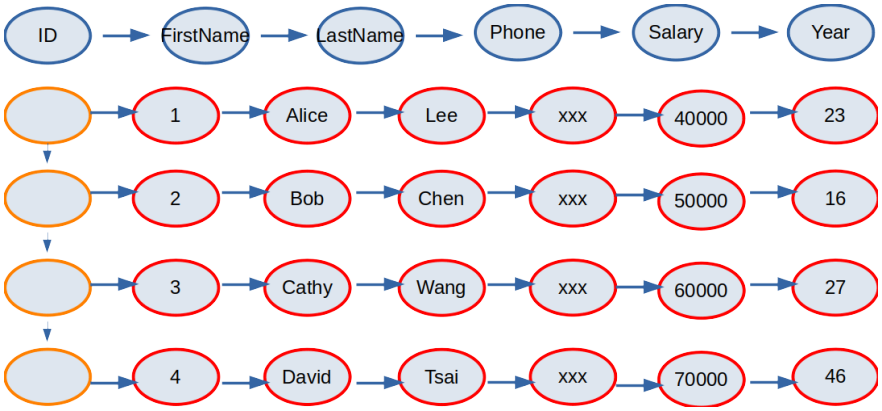

Figure 4: Database data structure.

```
typedef struct _sRecord
{
  struct list_head data;    // Data for each item in this record
  struct list_head list;    // List for the next record
} sRecord;

typedef struct _sItem
{
  char        *pData;
  struct list_head  next;   // Data for each item in this record
} sItem;

// Setup table labels.
// Note that this function must be called before all other functions.
// Input:
//    pLabelList: a list of sLabel.
// Return:
//    0: Success; -1: Error
int32_t setup_table( const struct list_head *pLabelList );

// Add a record.
// Input:
//    pRecordList: record list head.
//    pRecord: the new record;
// Return:
//    0: Success; -1: Error
int32_t add( struct list_head *pRecordList, sRecord *pRecord );

// Get the record list size.
// Input:
//    pRecordList: record list head.
// Return:
//    -1: Error; others, record size.
int32_t get_size( struct list_head *pRecordList );

// Query the database and get the result.
// Input:
```

```
//     pRecordList: record list head.
//     pCmd: the query command.
// Output:
//     pResultList: the queried result. Note that you should put the queried item an ignore all unqueried items.
// Return:
//     The number of query result.
int32_t query( struct list_head *pResultList, struct list_head *pRecordList, char *pCmd );
```

The command string format will be as follows:
SELECT [labels] **WHERE** [label="string"] [and|or] ...

- **SELECT,WHERE** are keywords and case sensitive.
- [labels] are items that should be contained in the result list.
- [label="string"] are given conditions. The target strings will be double quoted. Note that you should support **and,or** operator. For your simplicity, I promise all test commands will only use only one of them.

There is an example.

```
SELECT FirstName,LastName WHERE phone="0912111222" and year="23"
```

This command can gets the all records that satisfy the given phone number and year. The output record will only include FirstName item and LastName item. For your simplicity, I promise the query command will be definitely valid.

As usual, TAs will prepare fin03.c and I promise **database.h** will be included in it. **DO NOT FORGET TO BUILD fin03.c in your Makefile**.

4 Bonus: Your Comments (5 pts)

Your comments about this class. Any comments are welcomed. Do not worry about typos or any grammar errors. However, you will get nothing if you leave this question blank.