CS 392: Homework Assignment 1
Due: September 26, 11:55pm

Collaboration Policy. Homeworks will be done individually: each student must hand in their own answers. It is acceptable for students to collaborate in understanding the material but not in solving the problems or programming. Use of the Internet is allowed, but should not include searching for existing solutions.

Under absolutely no circumstances code can be exchanged between students. If some code was shown in class, it can be used, but it must be obtained from Canvas, the instructor or the TA.

Assignments from previous offerings of the course must not be re-used. Violations will be penalized appropriately.

Late Policy. No late submissions will be allowed without consent from the instructor. If urgent or unusual circumstances prohibit you from submitting a homework assignment in time, please e-mail me.

MovieLens Datasets Data for this assignment should be obtained from: `http://grouplens.org/datasets/movielens/`

You should use the following datasets for all problems:

1. 100,000 ratings from 1000 users on 1700 movies. Released 4/1998.
2. 1 million ratings from 6000 users on 4000 movies. Released 2/2003.
3. 10 million ratings and 100,000 tag applications applied to 10,000 movies by 72,000 users. Released 1/2009.

u.data: Data are stored as strings separated by a tab (\t):
user id item id rating timestamp
For example:

```
196     242     3       881250949
186     302     3       891717742
22      377     1       878887116
244     51      2       880606923
166     346     1       886397596
298     474     4       884182806
115     265     2       881171488
253     465     5       891628467
305     451     3       886324817
6       86      3       883603013
```

Problem 1. text2bin (10 points) Create a program that transforms the data file from text to binary. Your program should be called `text2bin` and take two mandatory command line arguments

```
text2bin <input filename> <output filename>
```

The input filename is the u.data file in the dataset. The output filename is the file to store the binary output of your program. The output should include the same data in the same order but in binary using the following format:

user id	item id	rating	timestamp
2-byte integer	2-byte integer	1-byte integer	8-byte integer

I recommend the following steps:

1. Use any of the character/string/buffer C stream input functions to read the data.
2. Use fwrite() to produce the output.
3. To read the four strings from the input file:
   - write custom code, or
   - try using strtok(), or
   - use strtoul() and friends
4. To convert a string to a number
   - Use atoi() and friends, or
   - use strtoul() and friends
5. Start with something like:

```
int main(int argc, char **argv)
{
        if (argc != 3) {
                fprintf(stderr,
                    "Wrong number of command-line arguments\n");
                usage(argv[0]);
                return -1;
        }
...
```

Problem 2. bin2text (10 points) Create a program that transforms the binary file you created back to text. Your program should be called `bin2text` and take two mandatory command line arguments:

```
bin2text <input filename> <output filename>
```

The input filename is the binary file. The output filename is the file to store the text output of your program. The output should be identical to the original dataset file.

I recommend the following steps:

1. Use any of the character/string/buffer C stream input functions to read the data.
2. Use fprintf() to produce the output.

Problem 3. bin2indexed (20 points) u.data includes item IDs for movies but looking up the actual title of the movie in u.item is slow. Create a program that **replaces the item ID in the binary file with the position (offset) of the corresponding movie item in the u.item file**. Your program should be called `bin2indexed` and take three mandatory command line arguments:

```
   bin2indexed <binary file> <item file> <output filename>
```

The binary file is the file produced by `text2bin`. The item file is u.item from the dataset. The output filename is the file to store the binary output of your program.

The output should include the same data in the same order but in binary using the following format.

user id	item file offset	rating	timestamp
2-byte integer	8-byte integer	1-byte integer	8-byte integer

u.item contains one line per movie. Movies are sorted by item ID. For example:

```
offset

  0   1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-
      exact?Toy%20Story%20(1995)|0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0
124   2|GoldenEye (1995)|01-Jan-1995||http://us.imdb.com/M/title-
      exact?GoldenEye%20(1995)|0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
      3|Four Rooms (1995)|01-Jan-1995||http://us.imdb.com/M/title-
      exact?Four%20Rooms%20(1995)|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
```

I recommend the following steps:

1. Read the text in bold carefully.
2. ftell() can be used to obtain the current offset of a stream within a file. For example: it should be zero right after the file is opened.
3. You may need to use malloc()/free()/realloc() to dynamically allocate memory for the index. Do not assume that you know how many movies there are. If this turns out to be too challenging, peek at the total number of movies for a small penalty.

Time your programs: Use `time p <command + arguments>` to time your programs with the various datasets. How does your program scale with file size?

Expectations.

1. Your program should compile and work correctly. Compiling and performing part of the requirements is better than not compiling.

2. Use comments where necessary to explain what you are doing. No comments or over-commenting are both bad.
3. Use functions! Do not place all your code in main().
4. Do not leak memory. Free all allocated memory and close all opened files.
5. Try to use only material covered in the course this far.

Deliverables. A zip/tar/gz file containing:

1. text2bin.c
2. bin2text.c
3. bin2indexed.c
4. A pdf file with brief explanations of your approach and timing results.