Assignment 1

In this assignment, you are asked to write ***a YouTube content moderation software*** that:

1) Provides two different sets of users:

    a. the user: those who will get videos and rate them.

    b. the creator: those who will generate new videos.

2) Get recommendations of videos based on a keyword and like/dislike ratio

3) Channel score to rank best channels

At this time, I will set the output format flexible. However, the user should have the following options:

1) Get videos

2) Get videos recommendations

3) Trending channels

4) Exit

For creators, the following options should appear:

1) Show channel score

2) Create a video

3) Manage a video

4) Exit

Our mini-project is about how the platform works, and how to find content.

The Classes

Within the folder data in the zip file, you will find files called Person.h , Person.cpp, User.h and Creator.h. These files provide a class to hold details of a person, and declarations for a derived class to hold details of a user and a creator. Such classes would generally be expected to have more data members but only those that are relevant to this assignment have been declared.

The Person class is complete, but you may, if you wish, add extra member functions to the class, e.g. an operator string method; *BUT you should not change any of the existing material, except where noted in this brief.*

The User class and Creator class declaration simply provide declarations for member functions that must be implemented; *you must provide complete implementation for these functions in a file called User.cpp and Creator.cpp, respectively*. The behavior of the functions should be as specified in the comments supplied in the header file. Again, you may, if you wish, add extra member functions to the class (e.g. a method to calculate the average reputation score); if you do so the function bodies should be written in the .cpp file.

The Main

In a separate file, you should create your main function.

Ask the username and identify what type of user he is, and direct to their respective options. Creators and users may be identifiable by a mechanism of your choice (here you can modify the existing one). It can be, as an example, name, or a given registration number. For users and creators, if the name does not match to a previous name, then a new one is created, and a message should be output to the screen. Create a template function login that works for both Creators and Users that attempt to open and read from a text file, whose name should be defined in the program, and in that file contains information for both creators and users. If the info of a particular line of that file is for a User, it will contain the registration number and name; these should be passed as arguments to the User constructor to create a new user object. If the info of a particular line of that file is for a Creator, it will contain a registration number, a creator name, and the channel score. You should create an object of the class Creator that holds who will be the Creator this time.

For users options:

1) Get video – This option will display 20 videos and their respective channels based on a keyword. You need to get videos randomly based on the keyword. Store the data here using a queue. You then ask the user for which video they want to get or if they want to return to the main menu. If they choose to get a video, display the video and ask if they like or dislike the video. You need to store these statistics about the video with a mechanism of your choice.

2) Get video recommendations – Ask the user for a keyword and then display top 10 videos based on their like/dislike ratio. The like/dislike ratio is given by this formula (likes/dislikes). You will need to use an STL algorithm to complete this task. Store the data here using a list. You then ask the user for which video they want to get or if they want to return to the main menu. If they choose to get a video, display the video and ask if they like or dislike the video.

3) Trending channels – Now you display to the user the top 10 channels with the best scores. A channel score is the average of all its videos scores. Use an STL algorithm to complete this task as well. Store the data here using a deque. You then ask the user for which channel they want to explore. Then list the videos of that channel and ask the video they want to get or if they want to return to the main menu. If they choose to get a video, display the video and ask if they like or dislike the video.

4) Exit – Just exit the program

For the creators options:

1. Show channel score – In this option you will calculate the channel score based on the previous formula, and output that to the creator. Again, you need to use an STL algorithm here.

2. Create a video – You ask the user for a name of the video, video duration and store it into the database file.

3. Manage a video – Get all videos into a map or a set. Display all videos to the creator and ask if they wish to:

a) rename a video,

b) reset likes/dislikes and

c) delete the video.

You then implement these options accordingly.

4. Exit – Just exit the program

Video Database

The program should read the sample file provided with assignment brief containing thousand fake video data with the following file structure;

video_id,video_name,video_duration,no_likes,no_dislikes,channel_name

The program should terminate cleanly with an appropriate error message if any of the files cannot be opened using an error exception. Your program must contain a class handling managing this database with a .cpp and a .h file.

NOTE: no class for a video has been created, but feel free to do that.

Challenge

Make the Creator class as a custom iterable container that operates similarly to a linked list. Videos inside the channel class then must be already sorted when accessed inside the main.

Deliverables

You should submit a single zip file containing a single folder. The only file formats acceptable are .zip , .7z or a linux gzipped tar file. If you submit a file in any other format, you will lose 10% of the marks for this assignment.

This folder should contain a MAKE file, necessary for the compilation of the application. Also, you need to include all files necessary to make your program executable.

Marking Scheme

Issues that will may lose marks for efficiency could include unnecessary copying of objects, unnecessary repetition of code. Your comments should say precisely what each function does and what each non-local variable represents; within function bodies you should use only brief comments to say what groups of lines do – you must not say what every statement does.

Marking Criteria	**Marks**
Style/Efficiency/Commenting	25
Implementation of all classes (User and Creator)	10
User and creator options	15
Handling the videos file	5
Get suggestions for videos	15
Implementation of the ranking for videos and channels	5
Requested usage of STL functions	10
MAKE file for compilation	5
Challenge	10
Total	**100**

I must compile the code using the MAKE file and execute the application for testing.