# 

# CSE 240: Course Project

*Video Game Recommendation List*

**3 Team members are required for this project. In the extreme case.**

# Project Objective:

The objective of this project is to create a video game recommendation system. The recommendation system will use a collaborative paradigm to compile a list of recommended video games for each gamer base on their preferences.

# Recommendation System Specification:

The recommendation system will automatically take in 2 comma-delimited files (game.csv and scoring.csv) during the startup phase of the application. Once all the files are loaded, the system will allow gamers to choose up to 2 genres (types) to generate their recommended list.

The recommendation system will use a list of criteria (specify later) to determine the rank for each game. Each genre will have its own computation to determine the overall score. (More details will be provided in the genre section.) Because the scoring.csv file will have multiple scores for the same game, your application will need to average out these scores.

A gamer will be allowed to select up to 2 genres to include in their recommendation list. Once selected, the system will display the list of top picked games in the following format:

Infamous: Second Son  
Console: PS4  
Genre: Role Playing  
Graphic: 10 Gameplay: 08 Storyline: 05   
Overall Score: 7.67

Mario Kart 8  
Console: Wii U  
Genre: Action  
Graphic: 10 Gameplay: 05 Storyline: 05   
Overall Score: 6.67

*(This is just a sample display; scores and computation may not be correct.)*

*\*\* The application must display the title of the game, the gaming console, the game genre, scoring criteria, and lastly the overall score. All scores must be rounded to 2 decimal places.*

## Game Genre

Video games can be classified into the following genres:

* Action
* Role Playing
* Strategy
* Sports

Gamers may pick from the list of genres to generate their recommendation list. (A gamer can only select up to 2 genres at one time to create their recommendation list.)

## 

## Scoring Criteria

The popularity of a game can be determined by the following criteria:

* Graphic
* Gameplay
* Storyline

Using a ranking system, we can score each criteria (with a value) to create a ranking. (Each criteria will be scored a value between 1 and 10, where the lowest value is awful and the highest value is the best.) The overall score will be determined through a list of computation provided below.

Each genre uses a different methodology to compute their overall score. Please follow the guideline below:

* ***Action games*** focuses on gameplay rather than storyline. Hence, we place heavier emphasis on gameplay than graphic and storyline. Please use this formula to compute the overall score:

* ***Role playing games*** focuses on storyline and gameplay. Please use this formula to compute the overall score:
* ***Strategy games*** focuses on gameplay only. We do include storyline in the calculation. Please use this formula to compute the overall score:
* ***Sport games*** focuses on graphic and gameplay. There is no storyline in sport games that we need to account for. Please use this formula to compute the overall score:

# Programming Requirements:

* C++ must be used to program this project. Visual Studios is highly recommended for this task.
* **Classes** must be usedin the project. ***(struct is forbidden!!!)***
* Demonstrate the use of **polymorphism** and **inheritance** when creating the different video games as classes.
* **Linked list** must be used to manage the video games.
* **Enumeration** and **type definition** must be implemented.
* Proper use of **memory management** must be demonstrated. (For every *new* called there must be a corresponding *delete* to free it from the heap memory.)
* Establish proper error handling and use of exceptions using throw and catch.

# Project Deliverables and Due Dates

This project has three (3) deliverables. For each deliverable, there are two submissions.

Each team member must submit the individual part into the folder “Individual Submission”. For this part, you must remove other member’s code and the joint code. You submit “my-code-only” in this part. If you do not have your own code, you will not receive credit. It is OK if the code does not work after removing other code. We will test your code in the joint submission.

One of the team members must submit the joint (entire) deliverable into the “Team Submission”. Only one submission per team is allowed for this part. In the joint work, each team member must identify their parts to receive credit. Please make that visible in the code as part of documentation.

The followings are the three deliverables.

1. First deliverable is due Saturday, **Oct 25th**. For milestone 1, we expect to see the following tasks completed:

* Team member 1: Create a Game class to store all the information from game.csv and scoring.csv. (Assume there is only one set of score for each game.) Do **NOT** create a variable within the class to store the overall score. You must create a function within the class to calculate the results. For the first deliverable, please average out the scores. For example, if a game scored 8 (gameplay), 8 (graphics), and 7(storyline); then the overall score will be 7.67.   
    
  Team member 1 must also create a linked list to manage all of the video games. (Hint: You will need to create a container class to store each of the video game and link them together).

Finally, create the main function to call ReadGameInfo() and ReadScore() in that same order.

* Team member 2: Create a ReadGameInfo() function to extract the information from game.csv. To demonstrate the function is working properly, we request that you print the name, genre, and console (for each game) to the screen.
* Team member 3: Create a ReadScore() function to extract the information from scoring.csv. To demonstrate your function is working correctly, we ask that you print out the name and the scores (for each game) to the console.

1. The second deliverable is due Saturday, **Nov 1st**. In addition to the content from the first deliverable, the second deliverable must include:

* Team member 1: Modify the existing Game class to use linked list to store multiple set of scores. Create the derive classes for each of the genre.

Create a Display() function to show each of the game in the linked list. The function must display all the content in the following format:

Mario Kart 8  
Console: Wii U  
Genre: Action  
Graphic: 10 Gameplay: 05 Storyline: 05   
Overall Score: 6.67

Finally, modify the main function and insert Display() function after ReadGameInfo() and ReadScore().

* Team member 2: Create an Insert() function to insert a new game into the linked list. If the game already exist in the linked list, do not duplicate it.

Update ReadGameInfo() function to use the insert function.

* Team member 3: Create a Search(char \*name) function to lookup an existing game in the linked list. If found, it should return a pointer that points to that Game object; otherwise, return null.

Update ReadScore() to utilize the Search function. You may need to create a new function inside Game class to keep track of the scores and compute the average score for each criteria. (Remember, there might be multiple entries for the same game.)

1. The last deliverable is due Saturday, **Nov 15th**. You must work as a team to complete this deliverable. We expect the following to be completed as part of this deliverable:

* Apply polymorphism to the derived classes – modify each class to have its own Compute() function.
* Modify Display() function to print the recommended list. Please allow gamers to select up-to 2 genres to create the list. Rank (by descending order) the games base on overall score.
* Implement error handlers.
* Any dynamic memory allocations must also be de-allocated. (We strongly recommend that you use destructor for this exercise.)