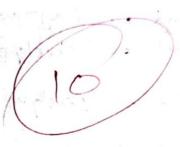
Q1 You have been provided with Developer and GameGenre classes to aid in a video game management system. Developers and genres can exist independently of any specific video game. You need to establish an aggregation relationship between the VideoGame class and the Developer and GameGenre classes.

```
class Developer (
private:
string name;
string location;
public:
Developer(string name, string location) {
this->name = name;
this->location = location;
string getDevName();
string getLocation();
class GameGenre {
private:
string name;
string description;
public:
GameGenre(string name, string description) {
this->name = name;
this->description = description;
string 'getGenreName();
string getDescription();
```



1. You need to code for the VideoGame class, which will demonstrate an aggregation relationship between the VideoGame class and the Developer and GameGenre classes. The VideoGame class should have title and playtime (in hours) as private attributes. You will need to write a parameterized constructor, destructor, and a display function to showcase the details of a video game, including its title, developer's name, location, genre, description, and playtime. (6)

2. Demonstrate its use in the main function and display the video game details using the display function. (4)

#include ciostream>

ming namespace std;

11 considering above other cl

class Video Game {

stoing title; int playtime;

Developer developer; Carregence game Gence 010220

Object Oriented Programming SE102T-A-Spring 2024

22nd April 2024

```
publici
  violeo Game (const stoingh -tit, const inth -pt) Developer devel,
                                          Come Gense , gone Gen) {
      +itle = -tit;
     playtime = -pt;
     developer = devel;
     game Gente = gane Ben;
 ~ Video Game = default;
  vil display() const {
     cont cc "Title: " cc title cc endl;
cont cc "Playtime: " ac playtime exendl;
     cout ec "Developer Nome: " ec developer set Der Nome () ec
    cont ce "Developer Location: " « c developer -> get Location() « endl;
     cont ec "Genre: " ec game Genre -> getGenre Nomec)
     cont ce "Description: " Le generalise -> get Doscorption()
                                                         ec endl;
int main () }
     Developed developes ("HD", "PC");
   Gome Genser game Gense ("PH", "This is Gense");
Wideogane game ("GTA", 20, &developet, EnganeGense);
      game. display();
     cout ec "This is somple implementation."
return 0;
```

CS CamScanner