# Section 1—C++ [20 Marks]

1. C++ is an Object-Oriented Programming language. Give a short explanation of what OOP is

and at least one benefit of OOP in game programming.

What is it:

OOP is a programming language model organized around objects rather than “actions” and data rather than logic.

Advantages:

OOP helps create maintainable code that is understandable, adaptable and extendable.

OOP allows for Code Reuse and Recycling.

[2 Marks]

1. Give short explanations for:
   1. Class – Class is a program-code-template for creating objects, providing initial values for state and implementation of behavior.
   2. Object – Objects refers to an instance of a class, objects can be a combination of variables, functions, and data structures.
   3. Inheritance – Inheritance allows new objects to take on the properties of existing objects.
   4. Polymorphism – Polymorphism refers to a programming languages ability to process objects differently depending on their data type or class; redefine methods for derived classes.
   5. Abstraction – Abstraction is a process where the programmer hides all but the relevant data about an object to reduce complexity and increase efficiency.
   6. Encapsulation – Encapsulation is the idea of bundling data and methods, and it’s also often used to hide the internal representation or state of an object from the outside.

[6 Marks]

1. Explain the difference(s) between class and header files in C++.

A header files contain any arbitrary C++ code and doesn’t typically contains any function definitions as they are usually intended to be used by more than one C++ module.

Class files contain the “implementations” which is where the header files functions are defined.

[2 Marks]

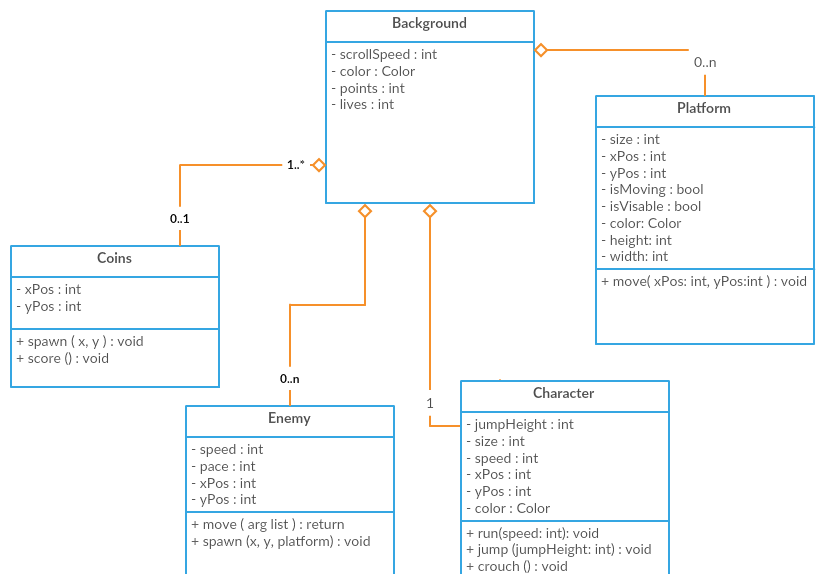
1. Give a brief explanation of the model, view and controller in MVC architecture for game development.

In MVC, the Model is used to define what data should be included in the application. The Model will usually notify the view and sometimes the controller, so they can be updated.

The View defines how the application’s data should be displayed. The controller contains logic that updates the model and/or view in response to input from the users of the application.

[3 Marks]

1. Design a possible class diagram for a simple platformer game. The game has a playable character that can move left, right, jump and crouch. The in-game objects include platforms that the player can walk on, coins and enemies. The enemies are placed randomly on platforms in the game world and can only move within the platform. Your diagram should include possible data members and member functions.



[7 Marks]

# Section 5—Problem Solving Capability [20 Marks]

1. Write approximately 500 words, describe the major game development problems you have overcome in your 2D sprite project. Please describe your solutions to these problems.

A major game development problem we faced in the sprite project was allowing the user to the input commands to the ‘sacks’ for this we developed the sacklang which is similar to lisp and asm.

Sacklang allowed for users to send commands to a specified sack, or BRAND a script into a sack which would be performed indefinity or until completed. It is the main way to interact with the game, though it did cause issues on OS’s that didn’t work well with (or was difficult to setup) named pipes.

Another major problem was issues with it running on windows since it used named pipes to send commands from the user to the game. By using named pipes we had inadvertently restricted the game to unix based system, which was a major issue since it was going to be marked on a windows os. For this we somewhat fixed it using the Bash subsystem for linux which is avalaible in windows 10 to install ubuntu (or another linux distro) and running it from there. This did also come with it’s fair share of issues but we managed to get it to run.

Sacklang was sort of an ongoing development problem, as with any other langauge we later found better ways to perform certain tasks have to then refactor the project, and so on. This kept happening as Sacklang was a core part of the game and so it became priority number one, whenever some new was added or something was changed. Solutions to Sacklang issues were in essence the same, find a more robust way to do the specified task.

Probably the biggest game development problem was overestimating our ability and time and chosing a scope that was outside our range. This lead to us, having to scrap or not being able to fully realize the idea we had wanted. This was a mistake we had made because we had started planning and done some development before we actually saw the Assignment Instruction and by the time we noticed the scope was too big we couldn’t chose some easier and we had to start scrapping different section. Though this didn’t stop just from making a rudamentory version of the game but it didn’t quite meet our own expectation for the game.

Otherwise, since I was in a group for the project, I didn’t really face any major problems. Josh (my group partner) used a similar structure for all the gameobjects from his first project (snake game) which made it a lot easier to keep track of things. For the sections I mainly worked on we didn’t face any major issues which halted other modules.

[20 Marks]

This is an individual assignment. You must work on the individual tasks by yourself and all work on these tasks must be your own.

Please sign the statement below to declare that this assignment submission is your own work and hand in the signed statement with your assignment. Failure to sign and include this statement may mean your assignment is not marked.

|  |
| --- |
| **ISCG6442—Game Programming**  Take Home Test  I declare that the individual part of this assignment submission is my own work. Where I have incorporated work by other people, I have correctly acknowledged the source in my assignment.  Student Name …Shivneel Achari………. Student ID …1463570.…  Date: …25/06/18... |