



# ASSIGNMENT BRIEFING

ITP4507 CTSE

# ASSIGNMENT AT A GLANCE

Friday 14 November, 2025



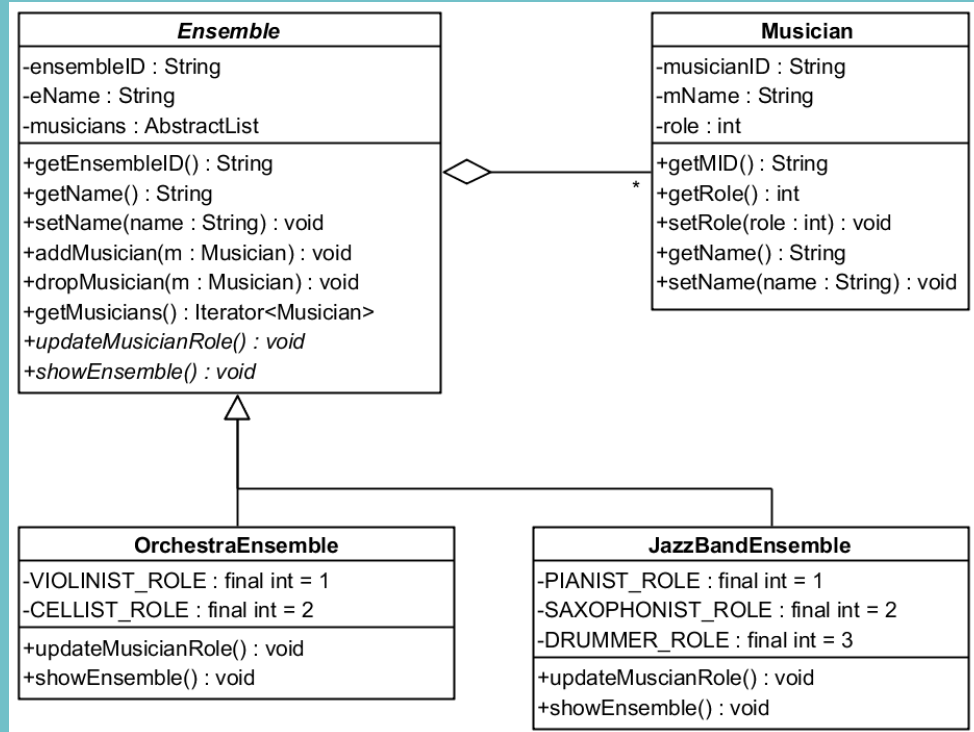
Source Code and  
Short Report

66.67% of the EA

On or before 4:30 PM to  
Moodle

# SCENARIO

- Ensemble is an abstract class (so as its abstract functions:  
updateMusicianRole(),  
showEnsemble())
- you may use Vector or LinkedList for AbstractList in Ensemble.
- not to change the design of the existing classes:  
Ensemble,  
OrchestraEnsemble,  
JazzBandEnsemble, and  
Musician.



# SCENARIO

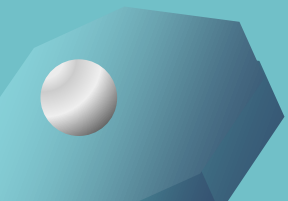
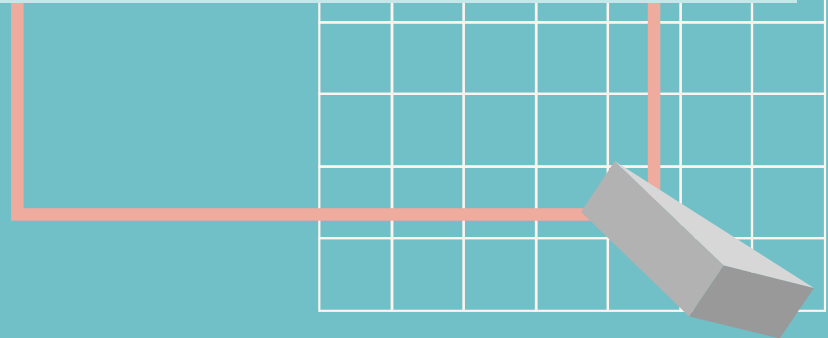


Music Ensembles Management System (MEMS)

c = create ensemble, s = set current ensemble, a = add musician, m = modify musician's instrument,  
d = delete musician, se = show ensemble, sa = display all ensembles, cn = change ensemble's name,  
u = undo, r = redo, l = list undo/redo, x = exit system

Please enter command [ c | s | a | m | d | se | sa | cn | u | r | l | x ] :- c

MEMS has 12 functions



# TASKS

Your system design should conform to the Open Closed Principle and easily be extended to support new music groups

## Command

Use Command Pattern to Implement all 11 command objects (except Exit System) (e.g. setEnsemble object)

## Factory

Use Factory pattern or Abstract Factory Pattern to create:

- Command objects
- Ensemble/Musician objects  
(e.g., JazzBandEnsemble object, OrchestraEnsemble object, Musician object)

## Memento

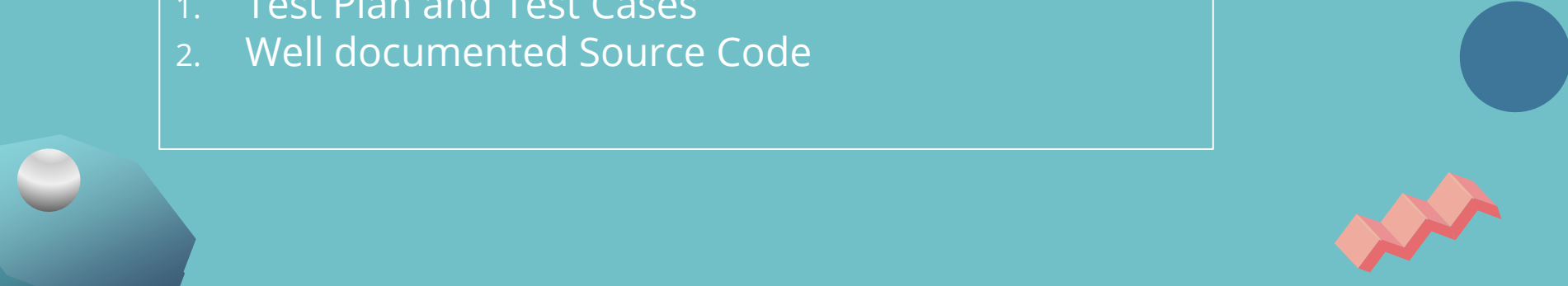

Use Memento pattern to provide "Undo" and "Redo" functions on:

- "modify musician's instrument"
- "change ensemble's name" functions"

# TASKS



## Assignment Report (Short)

1. Assumptions regarding the problem context
  2. Application design with **class diagram**
  3. Discussion and explanation (how) on each of the design patterns applied to the application
  1. Test Plan and Test Cases
  2. Well documented Source Code
- 
- 

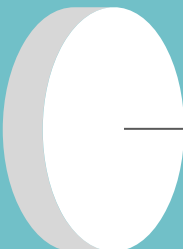

# GRADING INFORMATION

Estimation of your work		
For passing	For merit	For distinction
2 Patterns implemented correctly	3 Patterns implemented correctly	3 Patterns implemented correctly
Correct design with class diagram	Good design with clear class diagram	Elegant design with detailed class diagram

# CHECKPOINT: WHAT YOU NEED TO ANSWER



How will your design cope with **new Musician**, **new Ensemble**, or **new Command**?



The available roles of musician depends on the kind of Ensemble, however, unlike `updateMusicianRole()` in Ensemble, there is **no `createMusician()`** (you cannot add it), how do your design create new role of musician?