

Project Deliverable 1

SYST17796

Fundamentals of Software Design and Development

Submitted to Professor Paul Bonenfant

Submitted by

Group 4

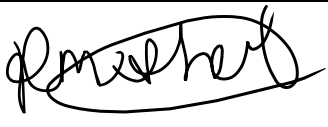

Jean/Tran Hoang

Robin Methot

Alexander Matthew

June 21, 2020

Team contract

Team Member Names (Please Print)	Signatures	Student ID
Project Leader: Robin Methot		991358325
Jean/Tran Hoang		991580595
Alexander Matthew	AM	991475460

Scenario	Accepted initials	We agree to do the following
Team member does not deliver component on time due to severe illness or extreme personal problem	JH AM RM	Team shifts target date if possible
Team member cannot deliver component on time due to lack of ability	JH AM RM	Team helps member
Team member does not deliver component on time due to lack of effort	JH AM	Team absorbs workload

Scenario	Accepted initials	We agree to do the following
	RM	
Team member does not attend team meeting	JH AM RM	Team proceeds without him/her and will assign work to the absent member
An unforeseen constraint occurs after the deliverable has been allocated and scheduled (a surprise test or assignment)	JH AM RM	Team will cope with constraint
Team cannot achieve consensus leaving one member feeling "railroaded", "ignored", or "frustrated" with a decision which affects all parties	JH AM RM	Team agrees to abide by majority vote
Team members do not share expectations for grade desired	JH AM RM	Team will elect one person as "standards-bearer" who has the right to ask that work be redone
Team member behaves in an unprofessional manner by being rude or uncooperative	JH AM RM	Team agrees to avoid use of all vocabulary inappropriate to the business setting
Team member assumes or requests that his/her name be signed to a submission but has not participated in production of the deliverable	JH AM RM	Team agrees that this is cheating and is unethical

Scenario	Accepted initials	We agree to do the following
There is a dominant team member who is content to make all decisions on the team's behalf leaving some team members feeling like subordinates rather than equal members	JH AM RM	Team will express subordination feelings and attempt to resolve issue
Team has a member who refuses to participate in decision making but complains to others that s/he wasn't consulted	JH AM RM	Team routinely checks with each other about perceived roles

Table of Contents

TEAM CONTRACT.....	2
UML DIAGRAM	6
I. PROJECT BACKGROUND & DESCRIPTION	6
A. DESCRIPTION.....	6
B. STARTING BASE CODE	7
II. PROJECT SCOPE	7
A. TEAM MANAGEMENT.....	7
B. TECHNICAL SCOPE	7
III. HIGH LEVEL REQUIREMENTS.....	7
IV. IMPLEMENTATION PLAN	8
A. PROJECT SETUP	8
C. ADDITIONAL USES.....	8
V. DESIGN CONSIDERATIONS.....	8

UML DIAGRAM

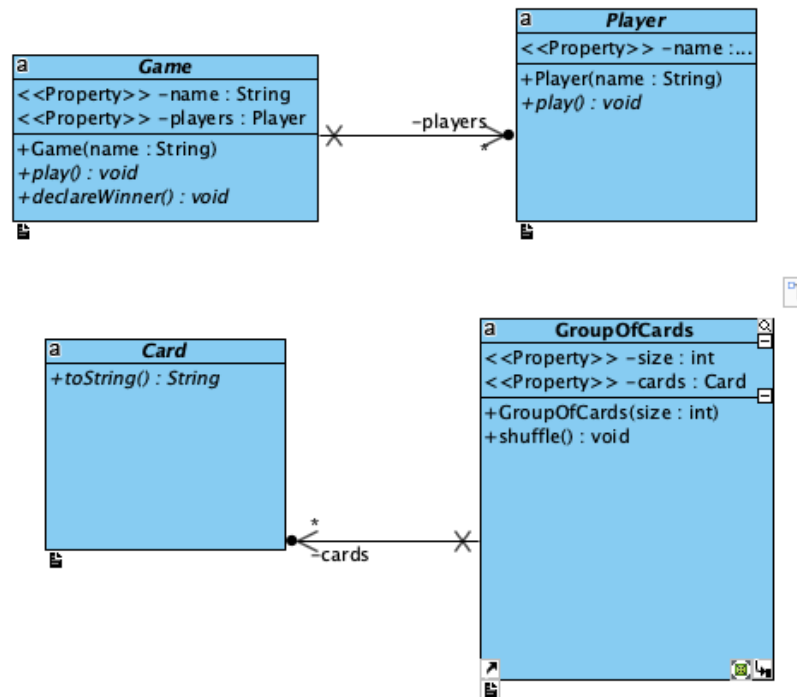


Figure 1 UML Diagram for BlackJack starter code

I. Project background & Description

Vietnamese BlackJack (Xi dách)

A. Description

1. Winning condition

To win the Vietnamese BlackJack, the player has to earn a point that's as close as possible to 21 but not exceeding 21.

At the start of every game, the dealer and the players will receive 2 cards each, the remainders are kept facing down.

2. Point System

In the deck cards rank from 2-10 and each cards value (King, Queen, Jack, etc.) becomes the amount of points gained when drawn. Jack, King, Queen carry the value of 10. Ace can carry either a point value of 1 or 10 or 11.

3. Game play

First round - Checking the values of the 2 pre-dealt cards

- If the 2 cards dealt at the beginning are a combination of either Jack, King, Queen or Ace, the player automatically wins.
- Player/dealer can stop drawing cards if they're happy with their assigned cards. If players/dealer wish to draw more card, they then proceed to second round.

Second round - Drawing more cards

- During this phase, players can lose and cannot withdraw more cards if their card total values exceed 21 points.

B. Starting base code

There are 4 classes used for the project's starting base code: Card, GroupOfCards, Game and player.

1. Card

Used enumerations to store the constant Suits and Values of a standard card deck of 52 cards (without Jokers).

2. GroupOfCard

Hold group/combination of cards for a game, which is stored into an ArrayList. This class also implements a method to shuffle the cards and get the size of the card combination.

3. Game

This is an abstract class that models the entire gameplay.

4. Player

This is an abstract class that models each player in the game. Where each player is assigned a unique identifier.

II. Project scope

A. Team management

The three members in Group 3 are: Robin Methot – Project Leader, Jean/Tran and Alexander Matthew. By creating our project's scope, we were able to determine and documented a list of specific project goals, deliverables, functions, tasks, and deadlines to keep us on track and up to date. The workload and roles were evenly split between our three members. Each team member has completed a section of code predetermined with the help of our project's scope. Along with an equal share of proofreading, planning, and filling out the report.

A to-do list is constructed with assigned team-members along with deadlines are posted weekly through Discord. Ideally group-meetings are held during the weekends where team members do not have exams or class interferences.

B. Technical Scope

For this project, Java Standard Edition is used. However, graphical user interface will not be used. The chosen IDE for this project is NetBeans. The entire project will be input/output through the NetBean's console log.

The project is completed once it's playable and outputs desired gameplay as described in the project background and description.

III. High Level Requirements

[Will construct a Use Case Diagram later for this part]

Players

Login: lets user enter their desired username

Game

The gameplay will always be the dealer

A system that determines if players win or lose

[Potential]

Point system

Displaying Card ASCII symbols

IV. Implementation plan

A. Project setup

Version control: GitHub

IDE: Netbeans

CASE tool: Visual Paradigm

GitHub URL: <https://github.com/jeanhoang/SYST17796-group3-deliverables.git>

Each team member should create a branch and consult with the team before merging with the repo's master branch.

Text directory is created for text files and documents.

Diagrams directory is created for UML/Use case diagrams for the project.

B. Coding standards

Each team members are required to follow the coding standards in which:

- Codes are format and indented properly.
- Codes should follow industry standards as learnt from Java 1, in which it is easy to: read, maintain, debug, robust and consistent.
- All codes should be documented properly using Javadoc

The outcome of this project should follows the Object Oriented Design Principle where it should be highly cohesive and loosely coupled.

C. Additional uses

Communication method: Discord/Slate

Text/Report: Google Word Doc Online

V. Design Considerations

Encapsulation

The interface should show the BlackJack gameplay while hiding the implementations from the users. In this case, all data members should be set to private and only methods/constructors needed for the console will be set to public.

Delegation

High cohesion and **loose coupling** are the main goals for the outcome for this project. Every classes are focused on ideally one logical entity. Classes in this project should also be independent to one another so that the project holds **Flexibility and Maintainability**. By making sure the project is highly cohesive and low-coupled, it will be easier to maintain since future needed changes only affect ideally few classes. This project also aims to be as simple as possible to reduce complexity with fewer operations. The program also looks into **extensibility** in which special case of winning and more implementations could be added easily in the future.