SCHOOL OF ENGINEERING AND TECHNOLOGY

COURSEWORK FOR THE
BSC (HONS) INFORMATION TECHNOLOGY; YEAR 1
BSC (HONS) COMPUTER SCIENCE; YEAR 1
BSC (HONS) INFORMATION TECHNOLOGY (COMPUTER NETWORKING AND SECURITY); YEAR 1
BSC (HONS) SOFTWARE ENGINEERING; YEAR 1

ACADEMIC SESSION 2023; SEMESTER 2,3,4

PRG1203: OBJECT ORIENTED PROGRAMMING FUNDAMENTALS

DEADLINE: 18 DECEMBER 2023 11:59PM (Monday)

INSTRUCTIONS TO CANDIDATES

- This assignment will contribute 20% to your final grade.
- This is a group (maximum 5 students) assignment

IMPORTANT

The University requires students to adhere to submission deadlines for any form of assessment. Penalties are applied in relation to unauthorized late submission of work.

Any work submitted after the deadline, or after any period of extension granted shall be marked as a Fail or awarded a zero.

Academic Honesty Acknowledgement

"I ....., ..., ..., ..., ... (student name). verify that this paper contains entirely my own work. I have not consulted with any outside person or materials other than what was specified (an interviewee, for example) in the assignment or the syllabus requirements. Further, I have not copied or inadvertently copied ideas, sentences, or paragraphs from another student. I realize the penalties *(refer student handbook undergraduate programme)* for any kind of copying or collaboration on any assignment."

....., ..., ..., ….., …... (Student's signature / Date)

Group Number: _____

Team Members:

No	Name	Student ID	Contribution %
1			
2			
3			
4			
5			

Marking Scheme

Criteria	Reference Marks		Marks	Remarks
Design (10%) Implement good object-oriented design in solving the problem, with high modularity, maintainability and reusability. Able to identify appropriate classes and their relationships, complete the classes with appropriate attributes and methods. Correct application of the inheritance and polymorphism concepts. The design is well presented in UML class and class relationship diagrams, and the coding is aligned to the design presented in UML.	**10**	**Excellent**		
	7-9	**Good**		
	4-6	**Average**		
	1-3	**Poor**		
Coding (5%) Fulfil all the functionalities, follow the best programming practices, such as naming convention, indenting, code structure, optimisation, with appropriate exception handling. Good user-friendliness.	**5**	**Excellent**		
	4	**Good**		
	2	**Average**		
	1	**Poor**		
Add-on Feature (5%) Implement at least one additional enhancement or feature to your program.	**5**	**Excellent**		
	4	**Good**		
	2	**Average**		
	1	**Poor**		
TOTAL	**20**			

Guideline

Getting Started: When embarking an object-oriented programming project, it's essential to follow these steps, prioritizing design before coding:

1. Identify the classes and their attributes, e.g. the model class and controller class.
2. Determine the class relationships, e.g. composition, aggregation, and inheritance.
3. Create a UML diagram to visualize your design.
4. Initiate the coding phase.

Evaluating Object-Oriented Design: To ensure that your program exhibits a strong object-oriented design, always verify your design against the following criteria:

Criteria of Quality Object-Oriented Design and Code

- **Modularity**: The program is divided into classes and objects, with each class having a clear and distinct responsibility. This promotes code reusability and maintainability.

- **Encapsulation**: Classes encapsulate their data (attributes) and behavior (methods), restricting direct access to internal state. Access to data is controlled through getter and setter methods.

- **Inheritance**: Inheritance is used when a class shares common attributes and behavior with another class. It allows for code reuse and the creation of specialized classes based on a common base.

- **Polymorphism**: Polymorphism enables objects of different classes to be treated as objects of a common superclass. It allows for flexibility and dynamic method dispatch.

- **Loose Coupling**: Objects are loosely coupled when they interact with each other through well-defined interfaces. This reduces the interdependence of components, making the system more flexible and maintainable.

- **High Cohesion**: Classes and modules should have a single, well-defined purpose, with methods and attributes that are closely related. This minimizes the need for excessive dependencies between classes.

- **Single Responsibility Principle (SRP)**: Each class should have only one reason to change, meaning it should have a single responsibility. This principle leads to more maintainable and understandable code.

- **Code Reusability**: Reusable components (classes, libraries, and modules) are designed to minimize redundancy and promote efficient code reuse.

- **Consistent Naming and Coding Conventions**: Adherence to consistent naming conventions and coding styles improves the readability and maintainability of the codebase.

- **Flexibility and Scalability**: Object-oriented design should allow the program to evolve and scale over time by accommodating new features and changes without major overhauls.

Deliverables

Checklist for the items to submit:

1. Report that include:
    a. Cover page with the team group ID, team members name and student ID.
    b. UML diagram (class and class relationship diagram) that includes all the classes and class relationships of the system.
    c. Reflection that includes:

        i. A description of the object-oriented concepts you have applied in the assignment (e.g. design of classes, inheritance, polymorphism, etc.).
        ii. A description of the add-on feature(s) you have implemented in the assignment.

    d. Video URL for item (2).

2. A short video (not more than 3 minutes) that describe and show the program is working for the basic features and any add-on feature(s) you have implemented. Upload the video to YouTube and attach the URL in the report.

3. Project/solution files, including all the source code (*.java) and application program. (Note: zip the whole project folder)

You are required to upload the report and source code to the eLearn. Only the team leader needs to do the submission.

Question

In this assignment, you are required to build a Java console program of the Pokémon Ga-Ole game. You are required to demonstrate the ability of applying the object-oriented knowledge in designing the system, identify the classes and the class relationships, and implement the inheritance and polymorphism in the solution.

Your program should include the **basic features** of 'Battle and Catch' mode as shown below. And implement at least an add-on features to enhance the game. To get more information about the game, you can visit: https://world.pokemongaole.com/ma/howtoplay/

■ About "Battle and Catch" mode

"Battle and Catch" is a mode in which
you battle with Wild Pokémon to catch them!
Choose a stage where you'll meet with the Pokémon
you want to catch, and depart for battle!

Battle controls are simple!
Just mash the buttons to move further ahead than your opponent!

Can you catch powerful Pokémon!?

1.Start "Battle and Catch"!

Once you insert coins and choose "Battle and Catch", select the stage you want to take on!

The Pokémon shown in the stage screen are more likely to appear in the stage! Each stage features different Pokémon you can meet!
*They are not guaranteed to appear.

Basic features: Start the game by generating a random set of Pokémon that will likely appear in the game.

2.Catch Time!

Before the battle begins, throw balls to catch Pokémon!
You can catch three Pokémon, so choose the Pokémon you want to put on a disk!
The Pokémon you've chosen will appear on a disk!
*With added coins, you can also have the two remaining Pokémon dispensed on a disk.

Basic features: Allow player to catch and collect one out of 3 Pokémon.

3.Depart for battle!

When you depart for battle, two Wild Pokémon will appear!
Look at the Pokémon types and affinities, then insert your Ga-Olé disks!

Players who don't have a Ga-Olé disk can still do battle with "Rental Pokémon"!

Basic features: Display the details of two wild Pokémon and allow player to send two of their Pokémon to the battle.

4.Let's go! Battle!

Battle controls are simple
Just mash the buttons to move further ahead than your opponent!

Once your Pokémon HP reaches zero, you can insert another Ga-Olé disk to continue the battle!

There are all kinds of chances that can be activated during a battle!
Use them wisely to get the advantage in moving your battle forward!

Basic features: The player's and opponent's Pokémon will take turn to attack. When the Pokémon is attacked, the HP will go down. When it reaches zero, the Pokémon is considered defeated. The effectiveness of the attack, will be decided by the factors of the defender type and move type. Each Pokémon should have one defender type and one move type. Your game should cater for at least three different defender and move types (refer to the chart below). You can refer to the full battle details at https://world.pokemongaole.com/ma/howtoplay/about_battle.html

Additional features which you can consider to add:

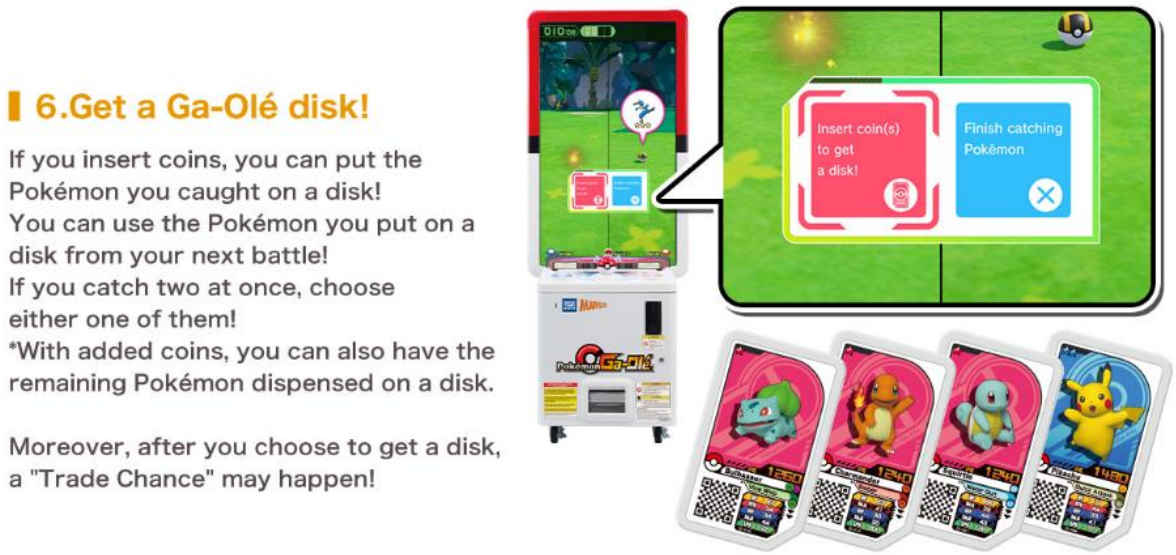
- There may be other chances that come out during battle: https://world.pokemongaole.com/ma/howtoplay/chances_battle.html
- And new "Double Rush" and "Rush Combo" feature: https://world.pokemongaole.com/ma/information/article/21757

Type Chart

Defender type \ Attacker's move type	Normal	Fighting	Poison	Ground	Flying	Bug	Rock	Ghost	Steel	Fire	Water	Electric	Grass	Ice	Psychic	Dragon	Dark	Fairy
Normal		●						×										
Fighting					●	▲	▲								●		▲	●
Poison		▲	▲	●		▲							▲		●			▲
Ground			▲				▲				●	×	●	●				
Flying		▲		×		▲	●					●	▲	●				
Bug		▲		▲	●		●			●			▲					
Rock	▲	●	▲	●	▲				●	▲	●		●					
Ghost	×	×	▲			▲		●									●	
Steel	▲	●	×	●	▲	▲	▲		▲	●			▲	▲	▲	▲		▲
Fire				●		▲	●		▲	▲	●		▲	▲				▲
Water									▲	▲	▲	●	●	▲				
Electric				●	▲				▲			▲						
Grass			●	▲	●	●				●	▲	▲	▲	●				
Ice		●					●		●	●				▲				
Psychic		▲				●		●							▲		●	
Dragon										▲	▲	▲	▲	●		●		●
Dark		●				●		▲							×		▲	●
Fairy		▲	●			▲			●							×	▲	

●: Super effective ▲: Not very effective ×: Almost no effect No mark: Normal effect

Basic features: When a Pokémon is defeated, players can attempt to catch it using Poke Balls. Stronger Pokémon may require the use of more powerful Poke Balls for a successful capture.



Basic features: No need to include this. However, you can consider adding the ability to save the Pokémon as your add-on feature.

7.Extra Battle

After "Catch Time" ends, there's a chance that an "Extra Battle" will occur! If an "Extra Battle" comes up, you can continue battling!

*Extra Battles happen at random.

Basic features: No need to include this.

8.Check your results!

Once the last battle is over, you can check the disks you got this time and your battle score on the results screen.

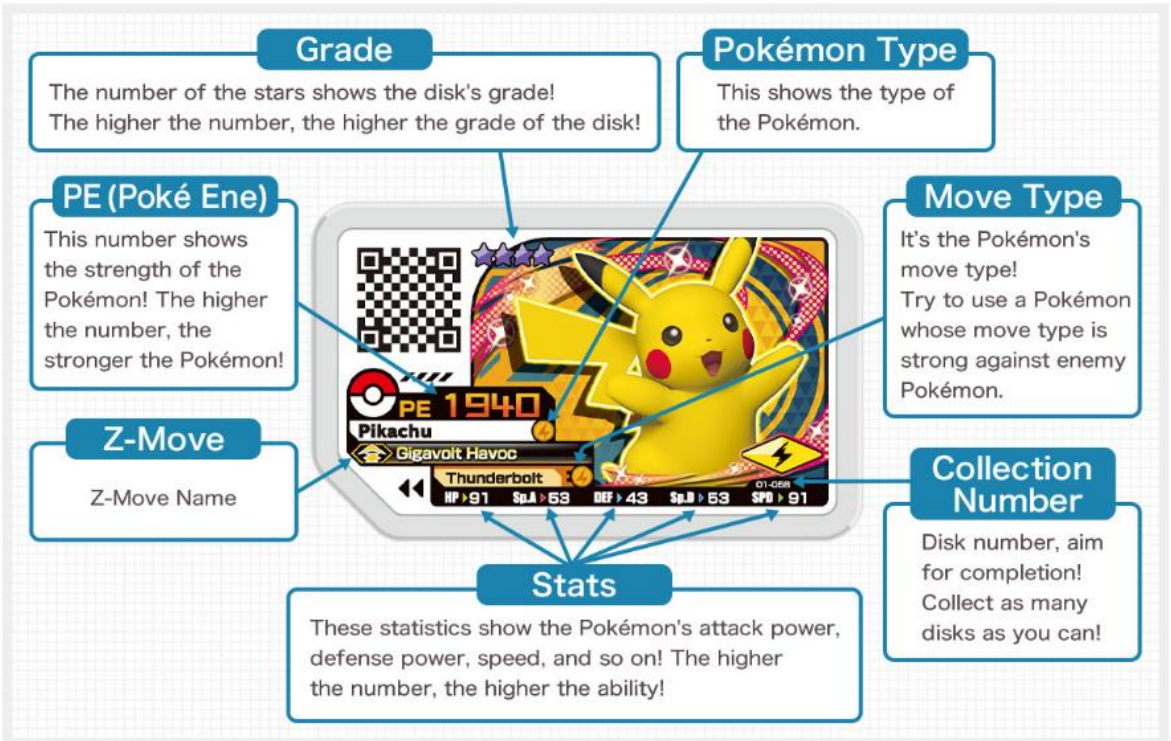
Basic features: Calculate and show the battle score. Allow saving the top 5 score in the top score list (in a file). And, display the top score list.

9.Ga-Olé Medals

Pokémon you caught in battle, "Golden Chip" and "Golden Sand" that Pokémon dropped are converted into Ga-Olé Medals, which accumulate in the machine! Once a certain amount accumulates, a good thing can happen!

Basic features: No need to include this. You can consider adding this as your add-on feature.

About Ga-Olé disks



The information of each Pokémon is shown as above.
To learn more about the disk: https://world.pokemongaole.com/ma/howtoplay/about_disks.html

Sample How to Play Video

Video 1:

https://youtu.be/eXXQgBy2rC8

Video 2:
https://sunwayedu.ap.panopto.com/Panopto/Pages/Viewer.aspx?id=c470da9a-a546-4f18-8edf-b0b40099108b