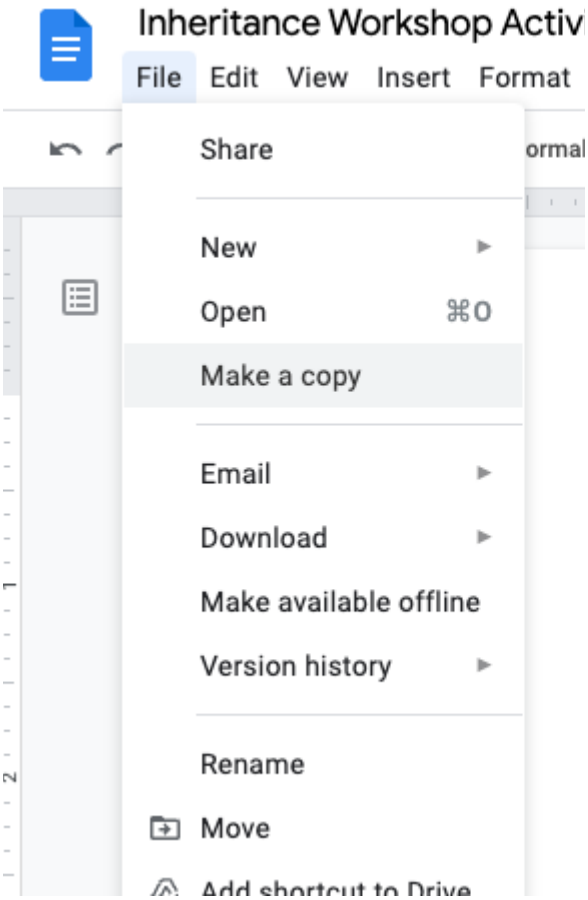
To start this activity, have one team member make a copy of this document and share with the rest of the team members present in the workshop.
This will give your team a copy to edit as you are working through the activity steps.

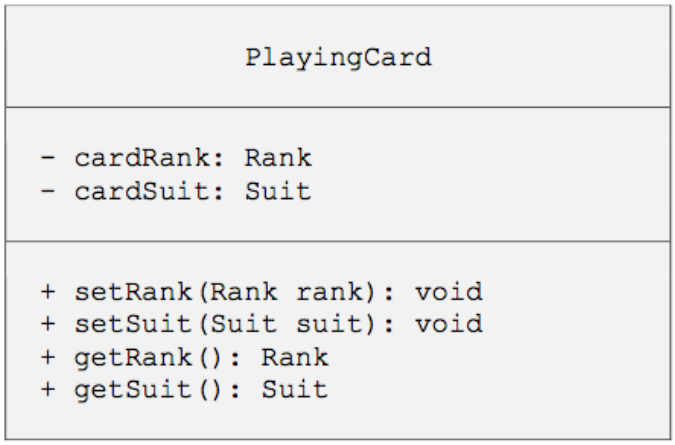


List team members present and participating in this activity:

Team Members
Ash Zahabiuon

"Is-a" versus "Has-a"

A large program will often be built out of multiple classes. Often, these classes will have a "has a" relationship. For example, a game program might include three classes: PlayingCard, Suit, and Rank. A PlayingCard object *has a* Suit and *has a* Rank, which are handled in a Java program as instance variables.



Less often, you might have two classes which have an "is a" relationship, where one class is a more general class, while the second class is a more specialized version of the original class. A blackjack program might have a BlackJackCard class which *is a* PlayingCard. Notice the relationship is not symmetrical: *every* BlackJackCard *is a* PlayingCard, but not every PlayingCard *is a* BlackJackCard.

Critical Thinking Questions

1. **Consider the PlayingCard, Rank and Suit classes.**
    a. **Which two classes are instance variables in the third class?**

- The Rank and Suit classes are instance variables in the PlayingCard class.

    b. **Are instance variables an *is-a* or *has-a* relationship?**

- Instance variables represent a "has-a" relationship.

2. **Consider the BlackJackCard and PlayingCard classes.**
    a. **Do these two classes have an *is-a* or *has-a* relationship?**

- These two classes have an "is-a" relationship.

    b. **Which class is the more general class?**

- The more general class is the PlayingCard class.

c. Which class is the more specialized class?

- The more specialized class is the BlackJackCard class.

d. Would the four methods from the general class be appropriate in the specialized class? Explain your answer.

- Yes, the four methods from the general class would be appropriate in the specialized class because a BlackJackCard is a type of PlayingCard and should have all the behaviors of a PlayingCard.

e. Give one example of a method that might appear in the specialized class but not the more general class.

- An example of a method that might appear in the specialized class but not the more general class could be calculateBlackJackValue(), which calculates the value of the card in the context of a game of Blackjack.

3. Explain why the relationship between the PlayingCard, Rank and Suit classes is not an *is-a* relationship.

- The relationship between the PlayingCard, Rank, and Suit classes is not an "is-a" relationship because a PlayingCard is not a type of Rank or Suit. Instead, a PlayingCard "has-a" Rank and "has-a" Suit.

4. Consider a Person class (which keeps track of a Person's name) and a Student class (which keeps track of a Student's name and student ID number). State the relationship between the two classes as either an *is-a* or a *has-a* relationship (similar to the last sentence of each paragraph in the above Model).

- The relationship between a Person class and a Student class is an "is-a" relationship. A Student is a type of Person.

5. As a group, come up with another example of a *has-a* relationship.

a. Which class is the instance variable?

- An example of a "has-a" relationship could be a Car and Engine.
- The Engine class is the instance variable.

b. What is the other class?

- The other class is the Car.

c. State the relationship between your two classes using the phrase "has a".

- The relationship between the two classes is that a Car "has-a" Engine.

6. As a group, come up with an example of an *is-a* relationship.

a. Which class is the more general class?

- An example of an "is-a" relationship could be a Dog and Animal.
- The more general class is the Animal.

b. Which class is the more specialized class?

- The more specialized class is the Dog.

c. State the relationship between your two classes using the phrase "every ____ is a ____".

- The relationship between the two classes is that "every Dog is an Animal".

7. Consider the BlackJackCard and PlayingCard classes again, especially your answer to question 2e above

a. How could this extra method be implemented by adding an instance variable?

- The extra method could be implemented by adding an instance variable like blackJackValue.

b. How could this extra method be implemented by adding code instead of an instance variable?

- The extra method could be implemented by adding code that calculates the value based on the rank of the card.

c. Which class would you add the instance variable or new code to?

- The instance variable or new code would be added to the BlackJackCard class, as it is the more specialized class.

Inheritance

The more general class is typically known as the **parent** class, **super** class, or **base** class. The more specific class is typically known as the **child** class, **sub** class, or **derived** class.

The Java keyword extends allows you to indicate that one class should be the child class of the original parent class. The child will then *inherit* everything from the parent class, including variables and methods.

Group Reflection:

Explain how an *is-a* relationship differs from a *has-a* relationship.

- An "is-a" relationship represents inheritance, where one class is a more specialized version of another class. A "has-a" relationship represents composition, where one class contains an instance of another class as a field. For example, a Car "has-a" Engine, and a Student "is-a" Person.