**How did you use classes and objects in the completed program?**

In the Hilo game program, I used two class objects: class Card and class Director. I separated the program into separate files, each object in its own python file and I was able to call and reuse the methods to allow the player to keep playing if they choose to continue playing. The game starts in the \_\_main\_\_.py file which imports the Director object to start the game. The Director imports the object Card from the Hilo.py file using a while loop for all true conditions of play on, else the game is terminated when the player chooses no or has zero points. The Director updates the player’s score by comparing the card number (1 to 13) with their guess (higher or lower), 100 points are awarded for correct guess and 75 subtracted for an incorrect guess.

**How did you apply abstraction in your program's design?**

In the program’s design, I divided the program code design into separate files, each performing a group of related functions in order to apply the principle of abstraction. There are two classes in the program: class Director and class Card. The Director imports the object Class using a while loop for all true of play on, otherwise, the game ends. The Director then updates the player’s score accordingly. The relationship of the program objects is summarized below:

Game entry point:

Imports class Director to initiate the game

Class Director:

Def \_init\_(self)

Def start\_game()

While loop: for when play is true (i.e. points > zero and play on = yes)

Def get\_inputs()

Gets inputs from player if they want to play

Def do\_updates()

Updates player score by generating a random card and comparing it to the player’s guess.

Def do\_output()

Displays the player’s score and checks if they can play again.

Class Card:

Def \_init\_(self):

Def play():

Checks if player can play on.