Homework #8

nim.py contains five functions, printScore, checkPlayer, computerTurn, checkComputer, and main. The main function prints out a welcoming statement for the Nim game.

While the player wants to play the game (variable playAgain is equal to ‘y’):

The main function generates a random number from 9 to 21 and set the random number equal to the variable currentSize. While the current size of the pile (currentSize) is not equal to 0, the main function will:

Print the current size of the pile and prompts the user for the number of items the user wants to remove from the pile or to enter ‘q’ if they want to quit the game.

If the user has entered ‘q’, the main function calls the printScore function. The printScore function takes two parameters, playerCount and computerCount. PlayerCount and computerCount hold the values of the number of times the user and computer have won the game, and is originally set to 0 at the start of the game. The printScore function prints out the final score by printing the value of playerCount (the number of times the user has won the game) and computerCount (the number of times the computer has won the game). After printing the final score, the main function will then exit the game.

If the user has entered a 1, 2, or 3, the main function will then see if the entered value (1,2,or 3) is greater than or equal to the current pile size. If the entered value is greater than or equal to the current pile size, the main function will print out that the user has entered an invalid number and the main function will prompt the user for a new input. If the entered value is less than the current pile size, the main function will calculate the new pile size (currentSize) by subtracting the user’s entered value from the value of currentSize (previous current pile size). The main function will then check to see if the player has won the game. The main function calls the checkPlayer function, which takes three parameters, currentSize, playerCount, and computerCount. If the value of newCurrent (pile size after subtracting the user’s entered value from the previous pile size) is equal to 1, the checkPlayer function adds one to the value of playerCount, prints out that the user has won, and calls the printScore function to print out the current score of the game. The checkPlayer function returns the value of playerCount. If the returned value of playerCount is larger than the value of playerCount before the main function called the checkPlayer function (previous value of playerCount), the main function adds one to the current value of playerCount within the main function and the user will be asked if they would like to play again. If the returned value of playerCount is not larger than the previous value of playerCount, the player has not won and the main function will then call the computerTurn function to allow the computer to take a turn. The computerTurn function takes one parameter, currentSize. If currentSize is less than or equal to four, the computerTurn function will subtract one from the value of currentSize and assign it to compChoice, the computer’s choice of the number of items to remove from the current pile. If the current pile size is not less than or equal to four, the computerTurn function will generate a random integer from 1 to 3 and assign it to compChoice. If the different between the values of newCurrent and compChoice is less than or equal to zero, the computer has chosen an invalid number and the computerTurn function will continue to generate a new random value until the value of compChoice is not greater than or equal to the value of newCurrent. The computerTurn function will then calculate the new current pile size (newCurrent) by subtracting the value of compChoice from newCurrent (previous pile size). The computerTurn function will print out

the number of items the computer has removed from the pile, return the value of newCurrent, and the main function will assign the value of newCurrent to the variable currentSize. The main function will then check to see if the computer has won the game. The main function calls the checkComputer function, which takes three parameters, currentSize, playerCount, and computerCount. If the value of newCurrent (pile size after subtracting the user's entered value from the previous pile size) is equal to 1, the checkComputer function adds one to the value of computerCount, prints out that the computer has won, and calls the printScore function to print out the current score of the game. The checkPlayer function returns the value of computerCount. If the returned value of computerCount is larger than the value of computerCount before the main function called the checkComputer function (previous value of computerCount), the main function adds one to the current value of computerCount within the main function and the user will be asked if they would like to play again.

If the user did not enter a valid number (1,2,or 3) or 'q', the main function prints out that the user has not entered a valid choice and prompts the user for a new input.

When the game has ended (the user or computer has won, or the user entered 'q'), the main function will ask the user if they would like to play again. If the user enters 'y', a new game will begin (condition for the while loop is still true). If the user enters 'n', the main function will print out the final score and the game will end.

The program should be run as:
python3 nim.py