Assignment SU3: Implementing a Hangman Game with the MyArrayList class

Objective:

Develop a Java Hangman game using the class MyArrayList. The program should read a list of words from a text file, sort the words alphabetically, and then allow the user to play the game with one of the words chosen at random.

Instructions:

1. Use the provided text file named `words.txt`.

2. Use the provided `MyArrayListHM` class to implement the generic MyArrayList functionalities.

3. Implement the Hangman game logic using the `MyHangman` class.

4. Write the 3 methods in the `MyHangman` class:

   1. Write the method loadWords() to read the words from the "words.txt" file, store them in the "words" arraylist and display the words. Use the sortlist() method of the `MyArrayListHM` class to sort the list and display the sorted list of words.
   2. Write the method startNewGame() to initialize a new game of Hangman. This method should randomly select a word from the list of words, set up the secret word and guessed word arrays, and initialize the number of guesses left.
   3. Write the method playGame() that handles user input, updates the game state, and provides feedback to the player. The method manages the gameplay loop to continue the game as long as there are guesses left and the word has not been completely guessed. (Hint – Use the updateGuessedWord() and isWordGuessed() methods discussed below)

5. Write a method updateGuessedWord() in the MyHangman class. The method receives the correct guessed letter as parameter and updates the guessed word with the correct guessed letter. (Hint - Use the remove and the add methods of the MyArrayListHM class to replace the '_' character with the correct guessed letter).

6. Write a method isWordGuessed() in the MyArrayListHM class to check if the entire word has been correctly guessed by the player. The method should return a boolean value indicating whether the word has been correctly guessed. (Hint - Use the contains method of the MyArrayListHM class to check if the guessed word still contains an underscore character '_')

Example Output:

```
The words: [Apple, Bread, Chair, Earth, House, Italy, Juice, Knife,
Lemon, Mouse, Snake, Table, Venus, Zebra, Eagle, Horse, Onion,
Pizza, Robin, Tiger]

The sorted words: [Apple, Bread, Chair, Eagle, Earth, Horse, House,
Italy, Juice, Knife, Lemon, Mouse, Onion, Pizza, Robin, Snake,
Table, Tiger, Venus, Zebra]

Welcome to Hangman!
```

```
Guessed word: [_, _, _, _, _]
Guesses left: 7
Enter a letter: a
Incorrect guess.
Guessed word: [_, _, _, _, _]
Guesses left: 6
Enter a letter: e
Correct guess!
Guessed word: [_, E, _, _, _]
Guesses left: 6
Enter a letter: r
Incorrect guess.
Guessed word: [_, E, _, _, _]
Guesses left: 5
Enter a letter: o
Correct guess!
Guessed word: [_, E, _, O, _]
Guesses left: 5
Enter a letter: s
Incorrect guess.
Guessed word: [_, E, _, O, _]
Guesses left: 4
Enter a letter: p
Incorrect guess.
Guessed word: [_, E, _, O, _]
Guesses left: 3
Enter a letter: m
Correct guess!
Guessed word: [_, E, M, O, _]
Guesses left: 3
Enter a letter: l
Correct guess!
Guessed word: [L, E, M, O, _]
Guesses left: 3
Enter a letter: n
Correct guess!
Congratulations! You guessed the word: [L, E, M, O, N]
```

Instructions to Complete the Assignment:

Use the two Java files: `MyArrayListHM.java` and `MyHangman.java` and complete the code.

Compile and run `MyHangman.java` to ensure the game works as expected.

Test the program with different inputs and scenarios to ensure proper handling of edge cases, such as invalid input or repeated letters.

Submit the Java files (`MyArrayListHM.java` and `MyHangman.java`).