Hangman - Word Game

In this homework, you should create a word game - Hangman. Initial template of the code is provided and it should be extended as described bellow.

The Game

Hangman is a famous word game. Given the hidden word, players should try and guess it. The player says the character and if it is part of the word, all occurrences of it should be revealed, otherwise, a piece of a hanged man is drawn.

Rules of Hangman:

```
 Try to guess the hidden word one letter at a
time. The number of dashes are equivalent to
the number of letters in the word. If a player
suggests a letter that occurs in the word,
blank places containing this character will be
filled with that letter. If the word does not
contain the suggested letter, one new element
of a hangman's gallow is painted. As the game
progresses, a segment of a victim is added for
every suggested letter not in the word. Goal is
to guess the word before the man hangs!
```

You can find more about the game in the web.

Structure of the gameplay

```
def hangman():
    while True:
        displayIntro()
        result = play()
        displayEnd(result)
        # TODO

if __name__ == "__main__":
    hangman()
```

The game starts from `hangman()` function. Currently, the game loop (`while True:`) is infinite. At the end of the loop, you should ask the user yes/no question - if he/she wants to play again and if not you should end the loop with `break` or `return`. Everything else must remain unchanged in this function. In while loop game is divided in 3 functions: `displayIntro`, `play` and `displayEnd`. You should implement these 3 main functions and several helper functions described below.

Display Intro

`displayIntro` is a simple function that should print the welcome screen and game rules in the terminal. You can use ASCII art and rules from examples and `hangman-ascii.txt` or you can come up with your own.

Display End

`displayEnd` is another simple function that takes one argument `result` and if it is true function should print winning text, if not, print that player lost the game and reveal the hidden word.

Helper Functions

Before we continue to 3rd main function, we need to implement several helper functions.

displayHangman function gets one argument - number of lives left. The Player has 5 lives at the beginning and it decreases if the player says the wrong letter. `displayHangman` function should draw the gallow and hanging man. If a user has all 5 lives only gallows should be drawn. If a player has 4 lives left draw gallows and the head of the man. On 3 lives add neck of the man and so on. When the player has zero lives left draw gallows and full man.

getWord This function should choose the word, which the user has to guess and return it. At first write a function that always returns the same word, 'moon' for example. This is not worth any points, but might be useful in the beginning. After that, you can improve function, so that it returns random word from an already predefined list of words: Create a list of words(Ex: `['moon','dog','horse','sun','river']`), select a random index and return the word which is on this index in the list. To select a random integer you can use python `randint` function. (Ex. `randint(1, 5)` randomly returns integer numbers from 1 to 5. *https://www.w3schools.com/python/ref_random_randint.asp*). Now `getWord` is a bit less boring, but we can make it even better. Instead of choosing a word from an already predefined list, choose words from `hangman-words.txt`. Each line contains one word, so you can read the file line by line and store it in the list. After that, you can select words randomly exactly like before.

valid function gets one argument - user's input. It should return true if the input is valid and false otherwise. Input is valid only if it is a single English lowercase letter.

Play

The main part of the gameplay is written in `play` function, but already created helper functions will make it a bit easier. It should act in the following way:

1. At the beginning of the game, you should choose the a random word from the `hangman-words.txt` to play with. `getWord` helper function should be used for this.
2. After that, the game starts. The player begins with 5 lives. It means that a player can make at most 5 mistakes until he/she losses the game.
    1. At each step, you should draw a hanged man, based on the number of lives left. When the player loses, the drawing should be finished. `displayHangman` should be used for drawing.
    2. At each step, you should display the current word. Guessed characters should be visible, while others should be hidden and replaced by '_','#','*' or something else.
    3. You should ask the player to enter the character.
        1. Input should be validated. If it is not a single lower case English character, ask again for it. Use proper helper function.
        2. If a given character appears in the word reveal hidden letters and display during the next turn.
        3. If the given character does not appear in the word, the player loses a life and ASCII art should be updated.
    4. If the player guesses the word or if all lives are lost, the game should end. `play` function should return True if the game was won or False otherwise.

Examples

- Welcome screen

```
_______________________________________________
 _
| |
| |__   __ _ _ __   __ _ _ __ ___   __ _ _ __
| '_ \ / _` | '_ \ / _` | '_ ` _ \ / _` | '_ \
| | | | (_| | | | | (_| | | | | | | (_| | | | |
|_| |_|\__,_|_| |_|\__, |_| |_| |_|\__,_|_| |_|
                    __/ |
                   |___/
_______________________________________________
_____________________Rules_____________________
Try to guess the hidden word one letter at a
time. The number of dashes are equivalent to
the number of letters in the word. If a player
suggests a letter that occurs in the word,
blank places containing this character will be
filled with that letter. If the word does not
contain the suggested letter, one new element
of a hangman’s gallow is painted. As the game
progresses, a segment of a victim is added for
every suggested letter not in the word. Goal is
to guess the word before the man hangs!
_______________________________________________
```

- In case of wrong guess a piece of man is drawn.

```


     ._______.
     |/      |
     |      (_)
     |
     |
     |
     |
 ____|___

Guess the word:  ____
Enter the letter:
> c


     ._______.
     |/      |
     |      (_)
     |       |
     |       |
     |
     |
 ____|___

Guess the word:  ____
Enter the letter:
```

- In case of correct guess appropriate letters are revealed.

```
     ._______.
     |/
     |
     |
     |
     |
     |
 ____|___

Guess the word:  m___
Enter the letter:
> o

     ._______.
     |/
     |
     |
     |
     |
     |
 ____|___

Guess the word:  moo_
Enter the letter:
```

- Winning screen

```

     ._______.
     |/      |
     |
     |
     |
     |
     |
 ____|___

Guess the word:  moo_
Enter the letter:
> n
Hidden word was:  moon
________________________________________________________________________
          _                                  _
         (_)                                (_)
__      ___ _ __  _ __   ___ _ __  __      ___ _ __  _ __   ___ _ __
\ \ /\ / / | '_ \| '_ \ / _ \ '__| \ \ /\ / / | '_ \| '_ \ / _ \ '__|
 \ V  V /| | | | | | | |  __/ |     \ V  V /| | | | | | | |  __/ |
  \_/\_/ |_|_| |_|_| |_|\___|_|      \_/\_/ |_|_| |_|_| |_|\___|_|
           | |   (_)    | |                  | (_)
        ___| |__  _  ___| | _____ _ __    __| |_ _ __  _ __   ___ _ __
       / __| '_ \| |/ __| |/ / _ \ '_ \  / _` | | '_ \| '_ \ / _ \ '__|
      | (__| | | | | (__|   <  __/ | | | | (_| | | | | | | | |  __/ |
       \___|_| |_|_|\___|_|\_\___|_| |_|  \__,_|_|_| |_|_| |_|\___|_|
________________________________________________________________________
Do you want to play again? (yes/no)
```

- Loosing screen

```
     ._______.
     |/      |
     |      (_)
     |      \|/
     |       |
     |      / \
     |
 ____|___

Hidden word was:  fort

 __     __           _           _   _
 \ \   / /          | |         | | | |
  \ \_/ /__  _   _  | | ___  ___| |_| |
   \   / _ \| | | | | |/ _ \/ __| __| |
    | | (_) | |_| | | | (_) \__ \ |_|_|
    |_|\___/ \__,_| |_|\___/|___/\__(_)

        _______ _                                        _ _          _ _
       |__   __| |                                      | (_)        | | |
          | |  | |__   ___   _ __ ___   __ _ _ __     __| |_  ___  __| | |
          | |  | '_ \ / _ \ | '_ ` _ \ / _` | '_ \   / _` | |/ _ \/ _` | |
          | |  | | | |  __/ | | | | | | (_| | | | | | (_| | |  __/ (_| |_|
          |_|  |_| |_|\___| |_| |_| |_|\__,_|_| |_|  \__,_|_|\___|\__,_(_)
__________________________________________________________________________
Do you want to play again? (yes/no)
```

NOTE: '>' denotes the input.\ NOTE: `hangman-ascii.txt` contains the complete example.\ NOTE: You can find ASCII drawings in the example file and copy it if you do not want to spend time on drawing or searching web.

Grading

This assignment is worth 10 points in total.

- displayIntro - 0.5 point if it works properly
- displayEnd - 0.5 point if it works properly
- displayHangman - 0.5 point if it works properly
- getWord - 0 point if it returns always the same word, 1.5 point if it returns random word from the list of words, 2.5 points if it returns random word from hangman-words.txt
- valid - 0.5 point if it works properly
- hangman - 0.5 point if the end condition was added properly
- play - 5 points

If the code does not compile and run, your code **will be graded with 0 points**.