Hangman Game

Making Game with Python (1)

Zhihong (John) Zeng & Andrew Zeng

Class 1

- Review and test
- Check a letter in a string
- Hangman game flowchart
- choose_word
- is_word_guessed
- get_guessed_word
- guess_loop
- Hangman game

Review

- import time module
- Escape character and multiline string
- User-defined function
- While loop
- Boolean operators: and, or , not

Test

```
# find the bugs
A = ‘It’s a test’

# what will be printed
print(‘Welcome\nToday is Sunday’)

def addition(x, y=0):
        ans = x + y
        return ans

print(addition(1, 2))
print(addition(1))
```

Test

```
# find the bugs
A = 'It's a test'

# what will be printed
print('Welcome\nToday is Sunday')

def addition(x, y=0):
        ans = x + y
        return ans

print(addition(1, 2))
print(addition(1))
```

```
# correction
A = 'It\'s a test'
A = "It's a test"

Welcome
Today is Sunday





3
1
```

Test

```
# what will be printed

x = 5
while x > 0:
        print(x)
        x -= 1  # x = x-1



x = 5
while x:  # while x != 0
        print(x)
        x -= 1
```

Test

```
# what will be printed

x = 5
while x > 0:
        print(x)
        x -= 1



x = 5
while x:
        print(x)
        x -= 1
```

0==False

```
# what will be printed

5
4
3
2
1

5
4
3
2
1
```

Check a item in a string

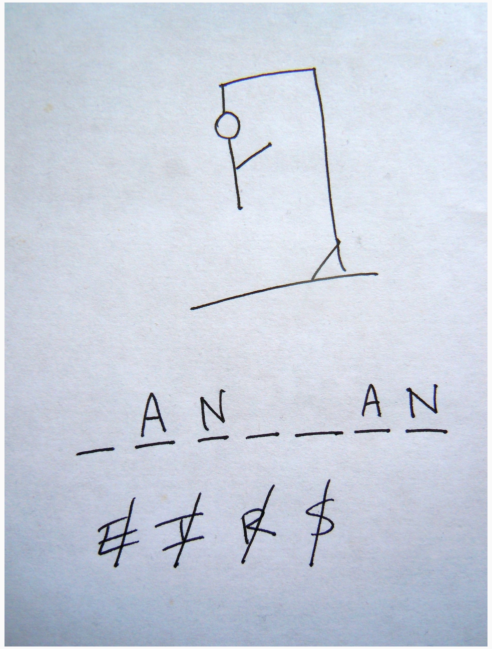
A = ‘abcd’

print(‘c’ in A) $\longrightarrow$ True

print(‘e’ in A) $\longrightarrow$ False

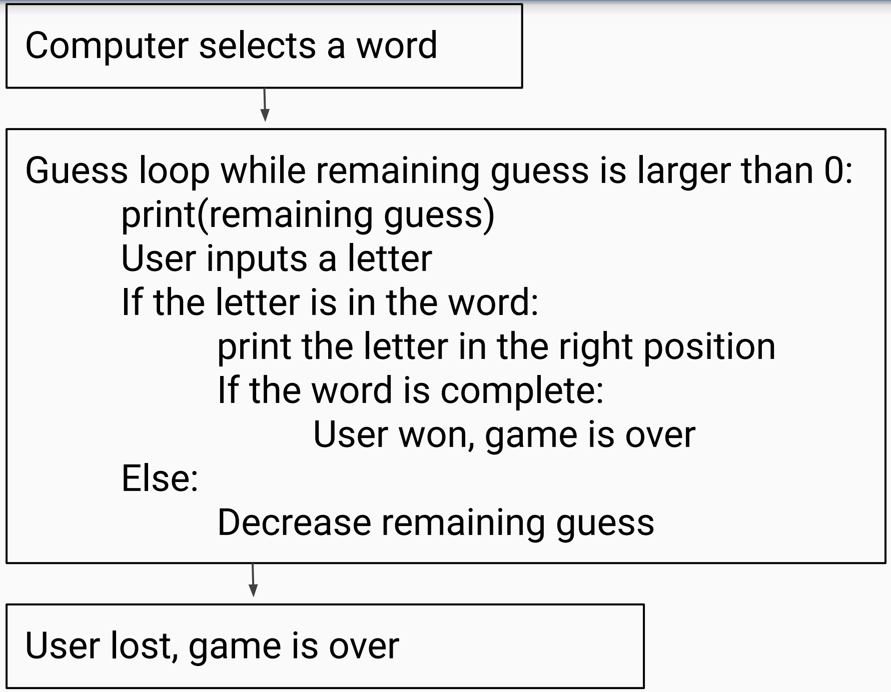
print(‘e’ not in A) $\longrightarrow$ True

Hangman Game



demo

Hangman Game Flowchart

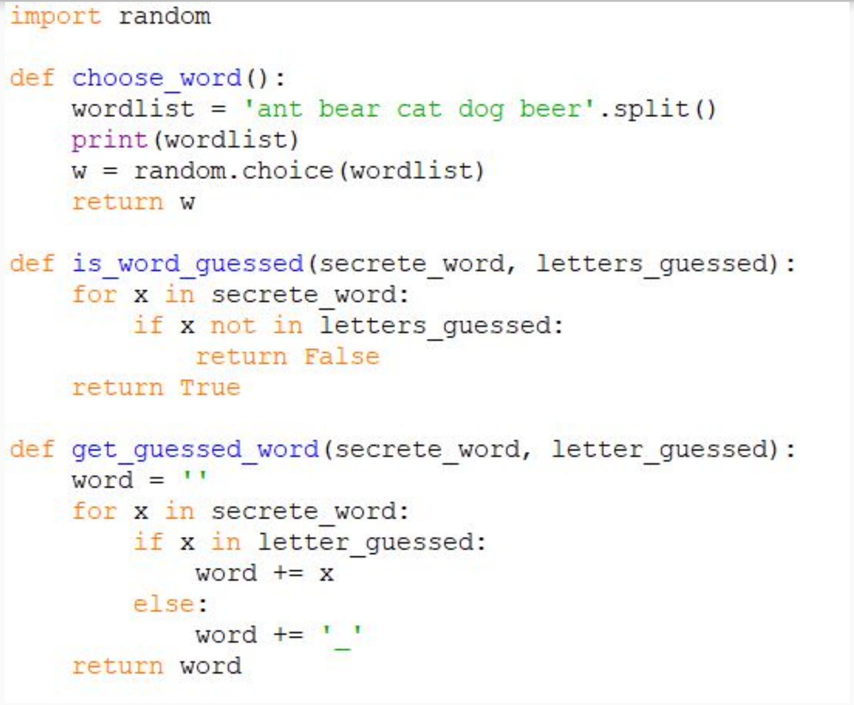


```
Computer selects a word

Guess loop while remaining guess is larger than 0:
        print(remaining guess)
        User inputs a letter
        If the letter is in the word:
                print the letter in the right position
                If the word is complete:
                        User won, game is over
        Else:
                Decrease remaining guess

User lost, game is over
```

User-defined functions



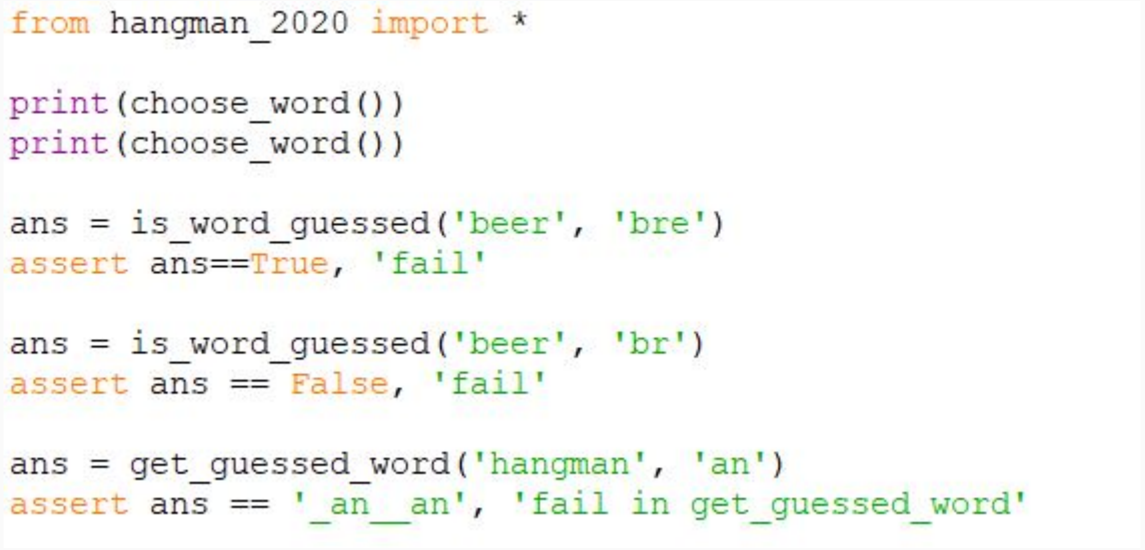
```
import random

def choose_word():
    wordlist = 'ant bear cat dog beer'.split()
    print(wordlist)
    w = random.choice(wordlist)
    return w

def is_word_guessed(secrete_word, letters_guessed):
    for x in secrete_word:
        if x not in letters_guessed:
            return False
    return True

def get_guessed_word(secrete_word, letter_guessed):
    word = ''
    for x in secrete_word:
        if x in letter_guessed:
            word += x
        else:
            word += '_'
    return word
```

Function test



```python
from hangman_2020 import *

print(choose_word())
print(choose_word())

ans = is_word_guessed('beer', 'bre')
assert ans==True, 'fail'

ans = is_word_guessed('beer', 'br')
assert ans == False, 'fail'

ans = get_guessed_word('hangman', 'an')
assert ans == '_an__an', 'fail in get_guessed_word'
```

Guess loop

```
def guess_loop(secrete_word, max_guess):
    remaining_guess = max_guess
    guessed = ''
    guessed_word = get_guessed_word(secrete_word, guessed)
    while remaining_guess > 0:
        print(f'You have {remaining_guess} guesses left')
        letter = input('Please guess a letter: ')
        letter = letter.lower()
        if letter in secrete_word:
            guessed += letter
            guessed_word = get_guessed_word(secrete_word, guessed)
            print(f'Good guess: {guessed_word}')
            if is_word_guessed(secrete_word, guessed):
                print('Congratulations, You won!\n')
                return
        else:
            print(f'Oops! That letter is not in my word: {guessed_word}')
            remaining_guess -= 1

    print(f'Sorry, you ran out of guesses. The word was {secrete_word}\n')

guess_loop('bear', 3)
```

Hangman game

```
def hangman(max_guess):
    secrete_word = choose_word()

    print(f"'"Welcome to the game Hangman!
    I am thinking of a word that is {len(secrete_word)} letters long.'"")

    guess_loop(secrete_word, max_guess)

if __name__ == '__main__':
    hangman(4)
```

Additional exercise: 1

```
import string
def get_available_letters(letter_guessed):
    """get available lower case alphabet letters
    Arguments:
        letter_guessed {string} -- guessed letters
    Returns:
        string -- available lower case alphabet letters excluding guessed letter
    """
    letters = string.ascii_lowercase
    remaining = ''
    for x in letters:
        if x not in letter_guessed:
            remaining += x
    return remaining

print(get_available_letters(‘bear’))
```

Guess loop

```
def guess_loop(secrete_word, max_guess):
    remaining_guess = max_guess
    guessed = ''
    while remaining_guess > 0:
        print('You have {} guesses left'.format(remaining_guess))
        print('Available letters: {}'.format(get_available_letters(guessed)))
        letter = input('Please guess a letter: ')
        letter = letter.lower()
        if letter in secrete_word:
            guessed += letter
            print('Good guess: {}'.format(get_guessed_word(secrete_word, guessed)))
            if is_word_guessed(secrete_word, guessed):
                print('Congratulations, You won!\n')
                return
        else:
            print('Oops! That letter is not in my word: {}'.format(get_guessed_word(secrete_word, guessed)))
            remaining_guess -= 1

    print('Sorry, you ran out of guesses. The word was {}\n'.format(secrete_word))

guess_loop('bear', 3)
```

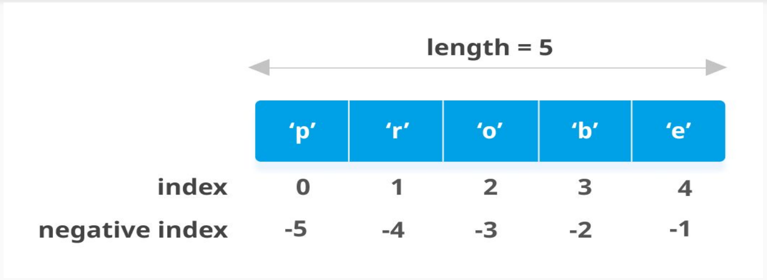
Additional exercise: 2

- List
- HANGMANPICS
- Hangman game with HANGMANPICS

list

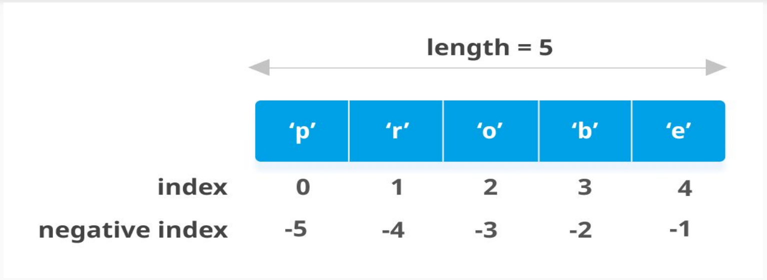
- A list is a collection which is ordered and changeable
- Syntax:
    - Square brackets: […]
- Create a list
    - mylist = [1, 2, 5, 6]
    - mylist = [‘a’, ‘b’, ‘apple’]
    - mylist = [‘a’, 1, ‘apple’]
    - mylist = [[1,2], [3,4]]
- Get the size of list
    - len(mylist)

list: access items



- mylist = [‘p’, ‘r’, ‘o’, ‘b’, ‘e’]
- print(mylist[0]) ‘p’
- print(mylist[2]) ‘o’
- print(mylist[-1]) ‘e’
- print(mylist[0:2]) [‘p’, ‘r’]

list: change item



- mylist = ['p', 'r', 'o', 'b', 'e']
- mylist[0] = 'a'
- print(mylist) ➡ ['a', 'r', 'o', 'b', 'e']
- mylist.append('t')
- print(mylist) ➡ ['a', 'r', 'o', 'b', 'e', 't']

list: for loop

```
a = list(range(6))

print(a)

for x in a:

    print(x)
```

list and string

- str1 = ‘probe’
- list1 = list(str1)
- str1_1 = ”.join(list1)

$\rightarrow$

[‘p’, ‘r’, ‘o’, ‘b’, ‘e’]
‘probe’

- str2 = ‘apple orange’
- list2 = str2.split(‘ ‘)
- str2_0 = ‘,’.join(list2)

$\rightarrow$

[‘apple’, ‘orange’]
‘apple,orange’

Check a item in a list or string

A = [1, 2, 3, 4]

print(2 in A) $\longrightarrow$ True

A=[‘a’, ‘b’, ‘c’, ‘d’]

print(‘c’ in A) $\longrightarrow$ True

A = ‘abcd’

print(‘c’ in A) $\longrightarrow$ True

HANGMANPICS

- HANGMANPICS is a list of multi-line string

  HANGMANPICS = [‘’’

  …’’’, ‘’’

  ….’’’]

(github.com/zhihongzeng2002/pythongame/tree/master/1: hangman_2019_3.py)

Game change

```
def guess_loop(secrete_word, max_guess):
    remaining_guess = max_guess
    guessed = ''
    while remaining_guess > 0:
        print('You have {} guesses left'.format(remaining_guess))
        print('Available letters: {}'.format(get_available_letters(guessed)))
        print(HANGMANPICS[max_guess-remaining_guess])
        letter = input('Please guess a letter: ')
        letter = letter.lower()
        if letter in secrete_word:
            guessed += letter
            print('Good guess: {}'.format(get_guessed_word(secrete_word, guessed)))
            if is_word_guessed(secrete_word, guessed):
                print('Congratulations, You won!\n')
                return
        else:
            print('Oops! That letter is not in my word: {}'.format(get_guessed_word(secrete_word, guessed)))
            remaining_guess -= 1

    print(HANGMANPICS[-1])
    print('Sorry, you ran out of guesses. The word was {}\n'.format(secrete_word))

hangman(len(HANGMANPICS)-1)
```