AI Lab\BFS.py

```python
from collections import deque

class Graph:
    def __init__(self, directed=False):
        self.graph = {}
        self.directed = directed

    def add_edge(self, u, v):
        if u not in self.graph:
            self.graph[u] = []
        self.graph[u].append(v)

        if not self.directed:
            if v not in self.graph:
                self.graph[v] = []
            self.graph[v].append(u)

    def bfs(self, start_vertex):
        visited = set()
        queue = deque([start_vertex])
        visited.add(start_vertex)

        traversal = []

        while queue:
            vertex = queue.popleft()
            traversal.append(vertex)

            for neighbor in self.graph.get(vertex, []):
                if neighbor not in visited:
                    visited.add(neighbor)
                    queue.append(neighbor)

        return traversal

if __name__ == "__main__":
    g = Graph()
    g.add_edge(0, 1)
    g.add_edge(0, 2)
    g.add_edge(1, 2)
    g.add_edge(2, 0)
    g.add_edge(2, 3)
    g.add_edge(3, 3)

    print("BFS traversal starting from vertex 2:")
    print(g.bfs(2))

```


>_pwsh
Python_LocalVC
master ≡ ?1
2ms
>> python -u "d:\SelfRepoClone\Python_LocalVC\AI Lab\BFS.py"
BFS traversal starting from vertex 2:
[2, 0, 1, 3]

AI Lab\Water_Jug_Problem.py

```python
from collections import deque

x = int(input("Enter capacity of Jug 1: "))
y = int(input("Enter capacity of Jug 2: "))
target = int(input("Enter Target to achieve: "))

def gcd(a, b):
    while b:
        a, b = b, a % b
    return a

if target > max(x, y):
    print("No solution possible: Target exceeds capacity of both jugs")
elif target % gcd(x, y) != 0:
    print("No solution possible: Target cannot be measured with these jug sizes")
else:

    visited = set()
    queue = deque([(0, 0, [])])
    solution_found = False

    while queue and not solution_found:
        jug1, jug2, steps = queue.popleft()

        if jug1 == target or jug2 == target:
            print("Solution found:")
            for i, step in enumerate(steps, 1):
                print(f"{i}. {step}")
            solution_found = True

        if (jug1, jug2) in visited:
            continue

        visited.add((jug1, jug2))

        if jug1 < x:
            queue.append((x, jug2, steps + [f"Fill jug 1: ({x}, {jug2})"]))

        if jug2 < y:
            queue.append((jug1, y, steps + [f"Fill jug 2: ({jug1}, {y})"]))

        if jug1 > 0:
            queue.append((0, jug2, steps + [f"Empty jug 1: (0, {jug2})"]))

        if jug2 > 0:
            queue.append((jug1, 0, steps + [f"Empty jug 2: ({jug1}, 0)"]))

        if jug1 > 0 and jug2 < y:
            pour = min(jug1, y - jug2)
            queue.append((jug1 - pour, jug2 + pour, steps + [f"Pour jug 1 to jug 2: ({jug1 - pour}, {jug2 + pour})"]))

```

```
        if jug2 > 0 and jug1 < x:
            pour = min(jug2, x - jug1)
            queue.append((jug1 + pour, jug2 - pour, steps + [f"Pour jug 2 to jug 1: ({jug1 + pour}, {jug2 - pour})"]))

    if not solution_found:
        print("No solution found")
```

```
>_pwsh  Python_LocalVC  master ≡ ?1  56s 844ms
>> python -u "d:\SelfRepoClone\Python_LocalVC\AI Lab\Water_Jug_Problem.py"
Enter capacity of Jug 1: 5
Enter capacity of Jug 2: 3
Enter Target to achieve: 4
Solution found:
1. Fill jug 1: (5, 0)
2. Pour jug 1 to jug 2: (2, 3)
3. Empty jug 2: (2, 0)
4. Pour jug 1 to jug 2: (0, 2)
5. Fill jug 1: (5, 2)
6. Pour jug 1 to jug 2: (4, 3)
```

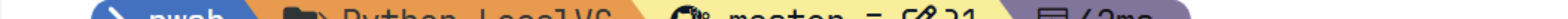
AI Lab\PunctuationRemoval.py

```python
import string

def remove_punctuation(input_string):
    translator = str.maketrans('', '', string.punctuation)
    return input_string.translate(translator)


sample_text = "Hi, I am Under the water! Here its too much raining :("
clean_text = remove_punctuation(sample_text)
print(f"Original: {sample_text}")
print(f"Cleaned: {clean_text}")

```




```
>> python -u "d:\SelfRepoClone\Python_LocalVC\AI Lab\PunctuationRemoval.py"
Original: Hi, I am Under the water! Here its too much raining :(
Cleaned: Hi I am Under the water Here its too much raining
```

AI Lab\sentenceOrder.py

```python
def sort_sentence(sentence):
    words = sentence.split()
    words.sort(key=str.lower)
    sorted_sentence = ' '.join(words)
    return sorted_sentence

input_sentence = input("Enter a sentence: ")
result = sort_sentence(input_sentence)
print("Sorted sentence:", result)
```


>_pwsh
Python_LocalVC
master ≡ ?1
49ms
>> python -u "d:\SelfRepoClone\Python_LocalVC\AI Lab\sentenceOrder.py"
Enter a sentence: The Fool that doesn't belong to this era;
Sorted sentence: belong doesn't era; Fool that The this to

AI Lab\Hangman Game\hangman.py

```python
import random

words = ['python', 'hangman', 'programming', 'computer', 'algorithm', 'database']
word = random.choice(words)
guessed = set()
attempts = 6

while attempts > 0:
    display = ''.join([letter if letter in guessed else '_' for letter in word])
    print(f"\nWord: {display}")
    print(f"Attempts left: {attempts}")
    print(f"Guessed: {', '.join(sorted(guessed))}")

    if display == word:
        print("\nYou won!")
        break

    guess = input("Guess a letter: ").lower()

    if len(guess) != 1 or not guess.isalpha():
        print("Please enter a single letter")
        continue

    if guess in guessed:
        print("Already guessed")
        continue

    guessed.add(guess)

    if guess not in word:
        attempts -= 1
        print("Wrong!")
else:
    print(f"\nGame over! The word was: {word}")
```

```
>_pwsh  Python_LocalVC  master ≡ ?5 ~2  67ms
>> python -u "d:\SelfRepoClone\Python_LocalVC\AI Lab\Hangman Game\hangman.py"

Word: ________
Attempts left: 6
Guessed:
Guess a letter: m
Wrong!

Word: ________
Attempts left: 5
Guessed: m
Guess a letter: e

Word: _______e
Attempts left: 5
Guessed: e, m
Guess a letter: d

Word: d______e
Attempts left: 5
Guessed: d, e, m
Guess a letter: a

Word: da_a_a_e
Attempts left: 5
Guessed: a, d, e, m
Guess a letter: t

Word: data_a_e
Attempts left: 5
Guessed: a, d, e, m, t
Guess a letter: b

Word: databa_e
Attempts left: 5
Guessed: a, b, d, e, m, t
Guess a letter: s

Word: database
Attempts left: 5
Guessed: a, b, d, e, m, s, t

You won!
```

AI Lab\tic_tac_toe.py

```

WIN_LINES = [
    (0, 1, 2), (3, 4, 5), (6, 7, 8),
    (0, 3, 6), (1, 4, 7), (2, 5, 8),
    (0, 1, 2), (2, 4, 6), (0, 4, 8)
]

def print_board(b):
    rows = [b[0:3], b[3:6], b[6:9]]
    print("\n  " + " | ".join(c if c != " " else str(i+1) for i, c in enumerate(b[:3])))
    print(" ---+---+---")
    print("  " + " | ".join(c if c != " " else str(i+1) for i, c in enumerate(b[3:6], start=3)))
    print(" ---+---+---")
    print("  " + " | ".join(c if c != " " else str(i+1) for i, c in enumerate(b[6:9], start=6)))
    print()

def winner(b):
    for a, c, d in {(0,1,2),(3,4,5),(6,7,8),(0,3,6),(1,4,7),(2,5,8),(0,4,8),(2,4,6)}:
        if b[a] != " " and b[a] == b[c] == b[d]:
            return b[a]
    if all(x != " " for x in b):
        return "D"
    return None

def get_move(b, player):
    while True:
        try:
            m = input(f"Player {player}, enter 1-9: ").strip()
            if m.lower() in {"q", "quit", "exit"}:
                return -1
            n = int(m)
            if 1 <= n <= 9 and b[n-1] == " ":
                return n-1
            print("Invalid move.")
        except ValueError:
            print("Enter a number 1-9.")

def game():
    b = [" "] * 9
    turn = "X"
    print_board(b)
    while True:
        idx = get_move(b, turn)
        if idx == -1:
            print("Game aborted.")
            return
        b[idx] = turn
        print_board(b)
```

```
        w = winner(b)
        if w == "X" or w == "O":
            print(f"Player {w} wins!")
            break
        if w == "D":
            print("Draw.")
            break
        turn = "O" if turn == "X" else "X"

def main():
    while True:
        game()
        again = input("Play again? (y/n): ").strip().lower()
        if again not in {"y", "yes"}:
            break

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

```
>_pwsh  Python_LocalVC  master ≡ ?1  29ms
>> python -u "d:\SelfRepoClone\Python_LocalVC\AI Lab\tic_tac_toe.py"

 1 | 2 | 3
---+---+---
 4 | 5 | 6
---+---+---
 7 | 8 | 9

Player X, enter 1-9: 1

 X | 2 | 3
---+---+---
 4 | 5 | 6
---+---+---
 7 | 8 | 9

Player O, enter 1-9: 9

 X | 2 | 3
---+---+---
 4 | 5 | 6
---+---+---
 7 | 8 | O

Player X, enter 1-9: 3

 X | 2 | X
---+---+---
 4 | 5 | 6
---+---+---
 7 | 8 | O

Player O, enter 1-9: 5

 X | 2 | X
---+---+---
 4 | O | 6
---+---+---
 7 | 8 | O

Player X, enter 1-9: 2

 X | X | X
---+---+---
 4 | O | 6
---+---+---
 7 | 8 | O

Player X wins!
Play again? (y/n): n
```

AI Lab\exp_07\stop_words.py

```python
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

nltk.download('punkt', quiet=True)
nltk.download('stopwords', quiet=True)

with open('D:/SelfRepoClone/Python_LocalVC/AI Lab/exp_07/input.txt', 'r') as file:
    text = file.read()

words = word_tokenize(text)
stop_words = set(stopwords.words('english'))

filtered_words = [word for word in words if word.lower() not in stop_words]

filtered_text = ' '.join(filtered_words)

print("Original text:")
print(text)
print("\nFiltered text:")
print(filtered_text)

with open('output.txt', 'w') as file:
    file.write(filtered_text)
```

```
>_pwsh  Python_LocalVC  master ≡ ?2 ~1  2ms                    13,23:19
>> C:\Users\adtya\AppData\Local\Programs\Python\Python312\python.exe -u "d:\SelfRepoClone\Python_LocalVC\AI Lab\exp_07\stop_words.py"
Original text:
The quick brown fox jumps over the lazy dog near the old wooden bridge by the river. In the morning, when the sun rises above the mountains, all the birds begin to sing their beautiful songs. The children were playing with their toys under the big tree while their parents sat on the bench and watched them carefully. There are many books on the shelf in the library where students come to study for their exams. This is a test passage that contains numerous common stop words such as the, is, are, in, on, at, to, for, of, and, a, an, with, by, from, about, between, during, before, after, while, because, although, since, until, unless, when, where, which, who, whom, whose, what, how, why, if, then, else, or, nor, but, yet, so, as, than, through, into, onto, upon, within, without, throughout, and many more.

Filtered text:
quick brown fox jumps lazy dog near old wooden bridge river . morning , sun rises mountains , birds begin sing beautiful songs . children playing toys big tree parents sat bench watched carefully . many books shelf library students come study exams . test passage contains numerous common stop words , , , , , , , , , , , , , , , , , , , , , , , , , , although , since , , unless , , , , , , , whose , , , , , , , else , , , , yet , , , , , , onto , upon , within , without , throughout , many .
```