# Mini-Project3

## Hangman-game

June 21th

Yoji

# Agenda

1. Overreview
2. Conding
    i. Word list
    ii. Validation
    iii. Game logic
    iv. Additonal function - Ranking
3. Demonstration
4. Conclusion

# Overreview

## Objective

Create simple hangman-game.

## Features

User can input 1-letter.
Game has to show

- What letters user has guessed
- What letters is correct or not
- How many guess count user still has

# Coding - Word list

## How did I get word list

I got word list which some made on GitHub
https://github.com/Xethron/Hangman/blob/master/words.txt

## How did I read word list

When initialize hangman-class, read word list

```
class hangman:
    def __init__(self) -> None:
        with open("words.txt", "r") as file:
            self.words = file.read().splitlines()
```

# Coding - Validation

After user type answer, it has validation.

- Only accept 1-letter alphabet.
- letter has not been guessed

```
if not letter.isalpha() or len(letter) != 1:
    print("Invalid input. Please enter a single letter.")
elif letter in lstguessed:
    letter = ""
    print("Already guessed!!")
```

# Coding - Game logic

- Make sure letter in the correct word
  - Update guessed
  - If guessed doesn't have "_", game would end
- Update used_words word list

```
elif letter not in word:
    tries -= 1
    print(f"Sorry, '{letter}' is not in the word.")
    lstguessed.append(letter)
else:
    print(f"Good job! '{letter}' is in the word.")
    lstguessed.append(letter)
    for i in range(len(word)):
        if letter == word[i]:
            guessed[i] = letter

if "_" not in guessed:
    print(f"Congratulations! You guessed the word!: {word}\n")
    break
```

# Additonal function - Ranking

When initialize hangman-class, read Ranking list

```
if os.path.isfile("ranking.json"):
    with open("ranking.json", "r") as file:
        self.ranking = json.load(file)
else:
    self.ranking = list()
```

When user win game, game ask user to want to save

```
if input("Do you want to save your score? (yes/no): ").lower() == "yes":
    name = input("Please enter your name: ")
    self.ranking.append({"name": name, "tries": tries})
    self.ranking = sorted(self.ranking, key=lambda k: k["tries"], reverse=True)
    self.ranking = self.ranking[:5]

    print("\nTop 5 ranking:")
    for i, player in enumerate(self.ranking):
        print(f"{i+1}. {player['name']} with {player['tries']} tries left.")
```

When user close game (destruct hangman-class), ranking-data is saved as json file

```
def __del__(self):
    if self.ranking:
        with open("ranking.json", "w") as file:
            json.dump(self.ranking, file)
```

# Demonstration

# Conclusion

### What was the challenges and impressions

Mini-Project3 was easier than Mini-Prject1. It didn't require new function, library and so on. Additionaly, it was solo project so that I didn't have to use GitHub with some branch strategy. I only use main branch.