Example

가위 바위 보 게임

1. 게임 소개:
전통적인 가위 바위 보 게임을 사용자와 컴퓨터가 대결하여 승자를 결정하도록 구현

2. 게임의 주요 구성요소:
- 사용자의 선택: 사용자는 가위, 바위, 보 중 하나를 선택
- 컴퓨터의 선택: 컴퓨터는 무작위로 가위, 바위, 보 중 하나를 선택
- 결과 표시: 두 선택을 바탕으로 승, 패, 또는 비김을 결정하고 결과를 화면에 표시

3. 게임 시작:
게임이 시작되면 사용자에게 가위, 바위, 보 중 하나를 선택하도록 안내

4. 게임 로직:
- 사용자의 선택: get_user_choice() 함수를 사용하여 사용자로부터 입력을 받음
- 컴퓨터의 선택: get_computer_choice() 함수를 사용하여 1~3 사이의 숫자를 무작위로 생성하여 컴퓨터의 선택으로 활용
- 승자 결정: determine_winner() 함수를 통해 사용자와 컴퓨터의 선택을 비교하여 승자를 결정
- 결과 표시: 선택과 결과는 print_choice() 함수를 통해 그래픽 형태로 화면에 표시

가위 바위 보 게임

5. 게임 반복:
사용자에게 게임을 계속할지 여부를 물어보며, 사용자가 종료를 원할 때까지 이를 반복

6. 게임 종료:
게임이 종료되면, 사용자의 승, 패, 비김 횟수를 화면에 표시하며 게임을 마침

7. 그래픽 디스플레이:
가위, 바위, 보의 각 선택은 특별한 그래픽 형태로 화면에 표시됩니다. 이를 위해 `print_choice()` 함수가 사용

가위 바위 보 게임 구조

- main() 함수: 게임의 주 로직을 관리
  - 승, 패, 비김 횟수 초기화
  - 반복 게임 루프
    - get_user_choice(): 사용자 선택 가져오기
    - get_computer_choice(): 컴퓨터 선택 가져오기
    - print_choice(): 선택한 옵션 그래픽 출력
    - determine_winner(): 승자 결정
  - 최종 결과 출력
- print_choice(choice): 선택한 옵션에 따른 그래픽 출력
- get_user_choice(): 사용자로부터 선택을 입력받아 반환
- get_computer_choice(): 1~3 사이의 임의의 숫자를 반환하여 컴퓨터의 선택으로 사용
- determine_winner(user_choice, computer_choice): 사용자와 컴퓨터의 선택을 비교하여 승자 결정

가위 바위 보 게임 (알고리즘)

```
함수 정의:
    print_choice(choice):
        만약 choice가 1이면 '가위' 그래픽 출력
        만약 choice가 2이면 '바위' 그래픽 출력
        그 외의 경우 '보' 그래픽 출력

    get_user_choice():
        사용자에게 가위, 바위, 보 중 어떤 것을 선택할 것인지 물어보
기
        사용자의 선택을 반환

    get_computer_choice():
        1에서 3 사이의 임의의 숫자(가위, 바위, 보를 나타냄)를 선택
        선택된 숫자 반환

    determine_winner(user_choice, computer_choice):
        만약 사용자와 컴퓨터의 선택이 같으면 0 반환 (비김)
        만약 사용자가 이기는 조합이면 1 반환 (사용자 승)
        그 외의 경우 -1 반환 (컴퓨터 승)
```

가위 바위 보 게임 (알고리즘)

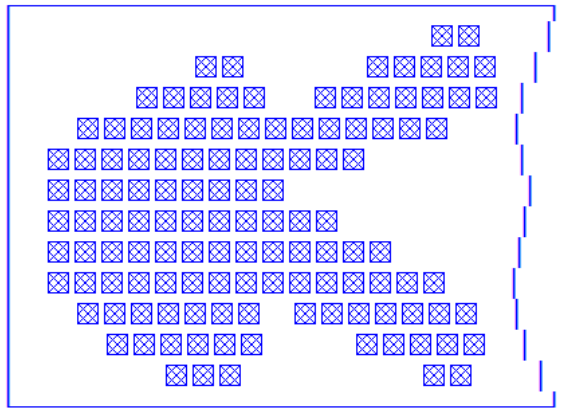
```
메인 함수:
    승, 패, 비김 횟수를 0으로 초기화

    무한 루프:
        사용자의 선택을 가져옴
        컴퓨터의 선택을 무작위로 결정
        두 선택을 화면에 그래픽으로 표시
        승자 결정 함수를 사용하여 승자를 판별

        승, 패, 비김 횟수 업데이트
        사용자에게 계속할 것인지 물어봄
        만약 사용자가 계속하지 않기로 결정하면 루프 종료

    게임 종료 메시지 출력 (최종 승, 패, 비김 횟수 포함)

메인 함수 실행
```

```python
import random

def print_choice(choice):
    if choice == 1:  # 가위
        print("┌────────────────────────────────────┐")
        print("│                            ▩▩    │")
        print("│            ▩▩        ▩▩▩▩▩  │")
        print("│        ▩▩▩▩▩   ▩▩▩▩▩▩▩ │")
        print("│    ▩▩▩▩▩▩▩▩▩▩▩▩▩▩    │")
        print("│  ▩▩▩▩▩▩▩▩▩▩▩▩          │")
        print("│  ▩▩▩▩▩▩▩▩▩                │")
        print("│  ▩▩▩▩▩▩▩▩▩▩▩            │")
        print("│  ▩▩▩▩▩▩▩▩▩▩▩▩▩        │")
        print("│  ▩▩▩▩▩▩▩▩▩▩▩▩▩▩▩    │")
        print("│    ▩▩▩▩▩▩▩  ▩▩▩▩▩▩▩  │")
        print("│      ▩▩▩▩▩▩      ▩▩▩▩▩  │")
        print("│          ▩▩▩            ▩▩    │")
        print("└────────────────────────────────────┘")

    elif choice == 2:  # 바위
        print("┌────────────────────────────────────┐")
        print("│                                    │")
        print("│        ▩▩▩▩▩                  │")
        print("│      ▩▩▩▩▩▩▩▩▩            │")
        print("│    ▩▩▩▩▩▩▩▩▩▩▩          │")
        print("│  ▩▩▩▩▩▩▩▩▩▩▩▩          │")
        print("│  ▩▩▩▩▩▩▩▩▩▩▩▩▩        │")
        print("│  ▩▩▩▩▩▩▩▩▩▩▩▩▩        │")
        print("│  ▩▩▩▩▩▩▩▩▩▩▩▩▩        │")
        print("│  ▩▩▩▩▩▩▩▩▩▩▩▩          │")
        print("│    ▩▩▩▩▩▩▩▩▩▩            │")
        print("│      ▩▩▩▩▩▩▩                │")
        print("│                                    │")
        print("└────────────────────────────────────┘")
```

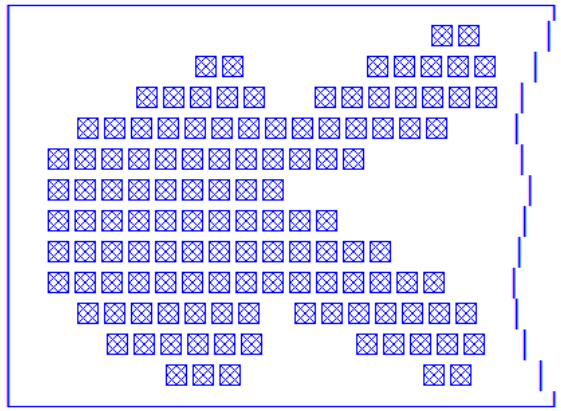
```python
    else:  # 보
        print(" ┌──────────────────────────────────────┐ ")
        print(" │                                      │ ")
        print(" │        ⊠⊠⊠⊠⊠                       │")
        print(" │      ⊠⊠⊠                            │")
        print(" │    ⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠          │")
        print(" │  ⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠                   │")
        print(" │  ⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠        │")
        print(" │  ⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠                   │")
        print(" │  ⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠          │")
        print(" │  ⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠                   │")
        print(" │    ⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠              │")
        print(" │      ⊠⊠⊠⊠⊠                      │")
        print(" │                                      │ ")
        print(" └──────────────────────────────────────┘ ")

def get_user_choice():
    print("가위, 바위, 보 게임!")
    print("1: 가위, 2: 바위, 3: 보")
    choice = int(input("당신의 선택은? "))
    return choice

def get_computer_choice():
    return random.randint(1, 3)

def determine_winner(user_choice, computer_choice):
    if user_choice == computer_choice:
        return 0  # 비김
    elif (user_choice == 1 and computer_choice == 3) or \
         (user_choice == 2 and computer_choice == 1) or \
         (user_choice == 3 and computer_choice == 2):
        return 1  # 사용자 승리
    else:
        return -1  # 컴퓨터 승리
```

```python
def main():
    wins, losses, draws = 0, 0, 0

    while True:
        user_choice = get_user_choice()
        computer_choice = get_computer_choice()

        print("\n당신의 선택:")
        print_choice(user_choice)
        print("\n컴퓨터의 선택:")
        print_choice(computer_choice)

        result = determine_winner(user_choice, computer_choice)
        if result == 0:
            print("비겼습니다!")
            draws += 1
        elif result == 1:
            print("당신이 이겼습니다!")
            wins += 1
        else:
            print("컴퓨터가 이겼습니다!")
            losses += 1

        print(f"\n현재 승리: {wins}, 패배: {losses}, 비김: {draws}")
        continue_game = input("\n계속하시겠습니까? (Y/N) ").lower()
        if continue_game != 'y':
            break

    print(f"\n게임을 종료합니다. 최종 승리: {wins}, 패배: {losses}, 비김: {draws}")

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

```
가위, 바위, 보 게임!
1: 가위, 2: 바위, 3: 보
당신의 선택은? 1

당신의 선택:
```

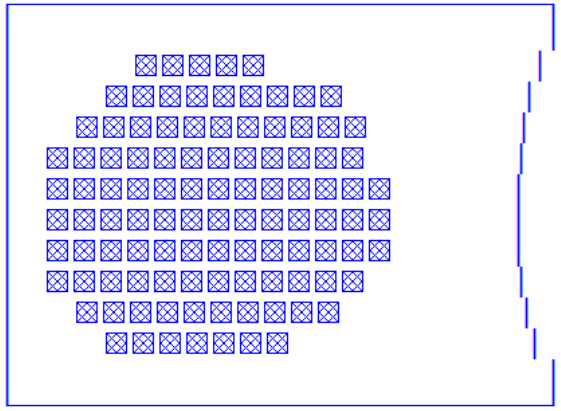


```
컴퓨터의 선택:
```

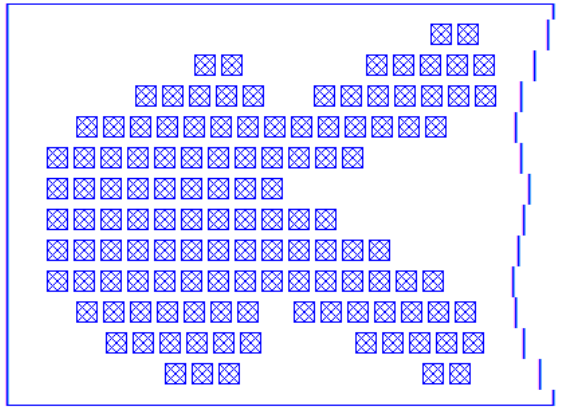


```
비겼습니다!

현재 승리: 0, 패배: 0, 비김: 1

계속하시겠습니까? (Y/N)
```

```
가위, 바위, 보 게임!
1: 가위, 2: 바위, 3: 보
당신의 선택은? 2

당신의 선택:
```

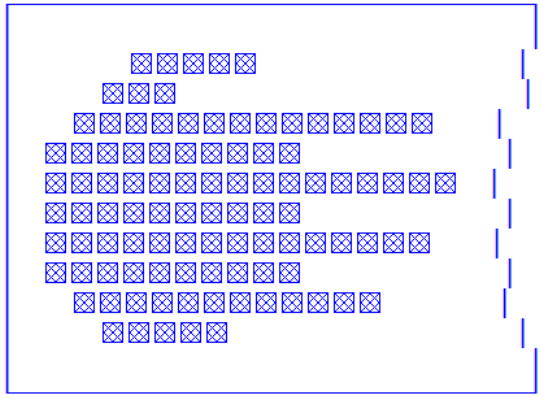


```
컴퓨터의 선택:
```

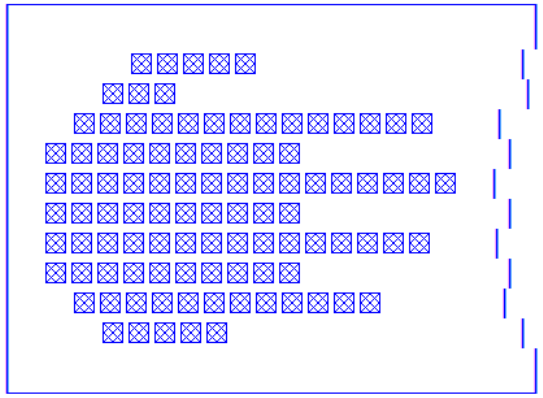


```
당신이 이겼습니다!

현재 승리: 1, 패배: 0, 비김: 1

계속하시겠습니까? (Y/N)
```

```
1: 가위, 2: 바위, 3: 보
당신의 선택은? 3

당신의 선택:
```



```
컴퓨터의 선택:
```



```
비겼습니다!

현재 승리: 1, 패배: 0, 비김: 2

계속하시겠습니까? (Y/N) n

게임을 종료합니다. 최종 승리: 1, 패배: 0, 비김: 2
```

- `if __name__ == "__main__":`
- `__name__` 변수

- 파이썬에서 모든 모듈은 내부적으로 `__name__`이라는 내장 변수를 가짐
- 파이썬 파일(모듈)이 직접 실행될 때, `__name__` 변수는 `"__main__"`으로 설정
- 만약 파이썬 파일이 다른 파일에 의해 임포트될 때, `__name__` 변수는 그 모듈의 이름으로 설정

- `if __name__ == "__main__":`

1. 스크립트와 모듈로의 사용성:
- 파이썬 파일은 스크립트로 직접 실행되거나 다른 스크립트에 모듈로서 임포트될 수 있음
- `if __name__ == "__main__":`을 사용하면, 파일이 직접 실행될 때만 특정 코드를 실행할 수 있게 하며 다른 스크립트에서 이 파일을 임포트할 때는 해당 코드가 실행되지 않음

2. 코드의 조직과 가독성:
   이 패턴을 사용하면 스크립트의 시작점을 명확하게 알 수 있으며 코드의 구조와 가독성을 향상시킴

```python
# example.py


def function_a():
    print("Function A")

def function_b():
    print("Function B")

if __name__ == "__main__":
    function_a()
    function_b()
```

1. `example.py`를 직접 실행하면, `__name__`은 `"__main__"`이 되어 `function_a()`와 `function_b()` 모두 실행됨
2. 다른 스크립트에서 `import example`를 사용해 `example.py`를 임포트하면, `function_a()`와 `function_b()`는 실행되지 않으며, `example.function_a()`와 같은 방식으로 함수를 호출할 수 있음

- f-string

- Python 3.6 이후 도입된 문자열 포매팅 방법
- 표현식 내부의 값을 직접 문자열에 삽입 가능
- 코드의 가독성 향상

- f-string의 기본 구조
- 문자열 앞에 `f` 또는 `F` 접두사 사용
- 중괄호 `{}` 안에 변수 또는 표현식 삽입
- 예제: `name = "John"; print(f"Hello, {name}")`

- 복잡한 표현식 또한 중괄호 내에 사용 가능
- 예제:
  - `x = 10`
  - `print(f"Value: {x}, Doubled: {x * 2}")`
  - 결과: `Value: 10, Doubled: 20`

- f-string

- 숫자 형식, 정렬, 패딩 등 다양한 포맷 옵션 가능
  - 예제:
    - `number = 3.14159`
    - `print(f"Pi: {number:.2f}")`
    - 결과: `Pi: 3.14`