자바 프로그래밍으로 구현한
카운터 스트라이크 글로벌 오펜시브

컴퓨터공학과 | 2학년 | 202111525 | 유승헌

목차

게임 소개

3

카운터 스트라이크 글로벌 오펜시브란 ?

카운터 스트라이크: 글로벌 오펜시브(CS:GO)는 Valve가 개발한 인기 1인칭 슈팅(FPS) 게임입니다. 테러리스트와 대테러리스트 두 팀으로 나뉘어 목표를 달성하거나 상대 팀을 전멸시키는 것이 주된 목표입니다. 폭탄 설치/해체 등을 진행하며 , 팀 전략과 협동이 중요합니다. 각 라운드에서 획득한 돈으로 무기를 구매할 수 있으며, 정확한 조준과 빠른 반응이 요구됩니다.

요구 사항으로는 맵을 무작위로 선택하는 기능, 플레이어 10명을 생성하고 팀을 나누고 각플레이어가 무기를 구매할 수 있는기능, 플레이어간 공격 및 폭탄 설치/해체 하는 기능, 라운드의 시작과 종료하는 기능/ 테러리스트와 대 테러리스트의 승리조건을 나누는 기능이 요구됩니다.

게임 최종 목표 및 승리 요건

4

테러리스트의 승리 요건

테러리스트의 최종목표

대 테러리스트를 전멸시키거나 대 테러리스트 의 방어구역을 폭발시켜 라운드의 승리를 가져갑니다.

테러리스트의 승리요건

1. 대 테러리스트 전멸시키기
2. 방어구역 폭탄 설치 후 폭발 시키기

대 테러리스트의 승리 요건

대 테러리스트의 최종목표

테러리스트를 전멸시키거나 , 테러리스트로 부터 방어 구역을 지켜냅니다.

대 테러리스트의 승리요건

1. 테러리스트 전멸시키기
2. 테러리스트의 폭탄 설치 막기
3. 설치가 되더라도 폭발 전까지 폭탄 해체하기

게임 코드 및 클래스 소개 – 1


```java
// Attack 클래스: 공격 기능을 담당하는 클래스
class Attack {
    public static void perform(Player attacker, Player target, boolean isHeadshot, BombPlanting bombPlanting, Round round) {
        if (attacker.getTeam() == target.getTeam()) {
            System.out.println("같은 팀을 공격할 수 없습니다!");
            return;
        }

        if (target.getHealth() <= 0) {
            System.out.println("이미 사망한 플레이어입니다!");
            return;
        }

        Weapon weapon = attacker.getWeapon();
        int damage = isHeadshot ? weapon.getHeadshotDamage() : weapon.getDamage();

        target.setHealth(target.getHealth() - damage);
        System.out.println(attacker.getName() + "이(가) " + target.getName() + "을(를) " + weapon.getName() + "(으)로 " + damage + " 데미지로 공격했습니다!");

        // 공격 시 타이머 단축
        if (bombPlanting.isBombPlanted()) {
            Random random = new Random();
            int reducedTime = 15 + random.nextInt(6); // 15 ~ 20초 랜덤 단축
            bombPlanting.reduceBombTime(reducedTime * 1000);
            System.out.println("폭탄 타이머가 " + reducedTime + "초 단축되었습니다!");
        }

        if (target.getHealth() <= 0) {
            System.out.println(target.getName() + "이(가) 죽었습니다!");
            attacker.getMoney().add(300); // 자금 추가
            attacker.setScore(attacker.getScore() + 1); // 점수 추가
            System.out.println(attacker.getName() + "이(가) 자금 $300원과 점수 1점을 획득했습니다!");

            if (round.checkRoundEnd()) {
                round.endRound();
            }
        }
    }
}
```

Attack 클래스

공격자가 대상 플레이어를 공격하는 상황을 다룹니다.
공격자와 대상이 같은팀인 경우 메시지를 출력하고 공격이 되지 않습니다.
대상플레이어가 이미 사망한 경우 메시지를 출력하고 중복적인 처치 및 사망한 플레이어를 재공격,처치, 처치로 인한 자금획득을 방지합니다.

공격자가 사용하는 무기의 데미지를 계산하고 헤드샷 여부에 따라 데미지를 추가합니다.
대상플레이어의 체력을 감소시키고 공격결과를 출력합니다.
사망 메시지를 출력하고 공격자에게 보상을 줍니다.
라운드 종료 조건을 체그하고 라운드를 종료합니다.

게임 코드 및 클래스 소개 – 2

6

```java
// Map 클래스: 맵 정보를 관리하는 클래스
class Map {
    private static final String[] maps = {"de_dust2", "overpass", "mirage"};
    private String currentMap;

    public Map() {
        Random random = new Random();
        currentMap = maps[random.nextInt(maps.length)];
    }

    public String getCurrentMap() {
        return currentMap;
    }
}
```

Map 클래스
FPS 에서 전장을 담당하는 클래스입니다.
랜덤하게 3개중에 한가지를 반환받아 게임을 진행합니다.

```java
// Money 클래스: 플레이어의 자금을 관리하는 클래스
class Money {
    private int balance;

    public Money(int balance) {
        this.balance = balance;
    }

    public int getBalance() {
        return balance;
    }

    public void add(int amount) {
        balance += amount;
    }

    public void subtract(int amount) {
        balance -= amount;
    }
}
```

Money 클래스

플레이어 개인의 자금을 관리하는 클래스 입니다

각각 플레이어들이 갖고 있는 자금의 양에 따라 라운드에서 사용할 수 있는 무기가 제한이 되기 때문에 게임에 미치는 중요한 부분의 클래스입니다.

게임 코드 및 클래스 소개 – 3


```java
// BombPlanting 클래스: 폭탄 설치 및 해체를 담당하는 클래스
class BombPlanting {
    private boolean bombPlanted;
    private boolean bombDefused;
    private Timer bombTimer;
    private long bombTimeRemaining;
    private Round round;

    public BombPlanting(Round round) {
        this.bombPlanted = false;
        this.bombDefused = false;
        this.bombTimeRemaining = 0;
        this.round = round;
    }

    public void plantBomb() {
        if (!bombPlanted) {
            bombPlanted = true;
            bombDefused = false;
            bombTimeRemaining = 2 * 60 * 1000; // 2분 설정
            System.out.println("폭탄이 설치되었습니다!");

            bombTimer = new Timer();
            bombTimer.schedule(new TimerTask() {
                @Override
                public void run() {
                    bombTimeRemaining -= 1000;
                    if (bombTimeRemaining <= 0) {
                        bombTimer.cancel();
                        bombTimer.purge();
                        if (!bombDefused) {
                            System.out.println("폭탄이 폭발했습니다! 테러리스트 승리!");
                            round.distributeRewards('T');
                            round.endRound();
                        }
                    }
                }
            }, 0, 1000);
        } else {
            System.out.println("이미 폭탄이 설치되었습니다!");
        }
    }
```

```java
    public void defuseBomb() {
        if (bombPlanted && !bombDefused) {
            bombDefused = true;
            bombTimer.cancel();
            bombTimer.purge();
            System.out.println("폭탄이 해체되었습니다! 대테러리스트 승리!");
            round.distributeRewards('C');
            round.endRound();
        } else {
            System.out.println("설치된 폭탄이 없습니다!");
        }
    }

    public void reduceBombTime(long time) {
        bombTimeRemaining -= time;
        if (bombTimeRemaining < 0) {
            bombTimeRemaining = 0;
        }
    }

    public boolean isBombPlanted() {
        return bombPlanted;
    }

    public boolean isBombDefused() {
        return bombDefused;
    }

    public long getBombTimeRemaining() {
        return bombTimeRemaining;
    }
}
```

BombPlanting 클래스

폭탄이 설치, 해체 여부/ 폭탄 타이머 관리를 합니다.

각팀에 승리요건을 달성하기 위해 반드시 필요한 클래스입니다. 테러리스트는 폭탄을 설치로 승리에 도달 해야하고, 대 테러리스트는 폭탄 설치를 저지해야합니다.

게임 코드 및 클래스 소개 – 4

8

```java
class Player {
    private String name;
    private char team;
    private int health;
    private int score;
    private Money money; // 플레이어의 자금
    private Weapon weapon; // 플레이어의 무기

    public Player(String name, char team) {
        this.name = name;
        this.team = team;
        this.health = 100; // 기본 체력 설정
        this.score = 0; // 초기 점수
        this.money = new Money(5000); // 초기 자금 설정
    }

    // 게터 및 세터 메서드
    public String getName() {
        return name;
    }

    public char getTeam() {
        return team;
    }

    public int getHealth() {
        return health;
    }

    public void setHealth(int health) {
        this.health = health;
    }

    public int getScore() {
        return score;
    }
```

```java
    public void setScore(int score) {
        this.score = score;
    }

    public Money getMoney() {
        return money;
    }

    public Weapon getWeapon() {
        return weapon;
    }

    public void setWeapon(Weapon weapon) {
        this.weapon = weapon;
    }

    // 플레이어 정보 출력 메서드
    public void displayInfo() {
        System.out.println("이름: " + name);
        System.out.println("팀: " + (team == 'T' ? "테러리스트" : "대테러리스트"));
        System.out.println("체력: " + health);
        System.out.println("점수: " + score);
        System.out.println("자금: $" + money.getBalance());
        if (weapon != null) {
            System.out.println("무기: " + weapon.getName());
        } else {
            System.out.println("무기: 없음");
        }
    }
}
```

Player 클래스

플레이어 이름, 팀, 체력, 점수, 자금, 무기 등을 저장합니다

게임에 참여하는 플레이어들의 기본적인 상황들을 저장하는 클래스입니다.

게임 코드 및 클래스 소개 – 5

9

```java
// Round 클래스: 라운드 진행을 담당하는 클래스
class Round {
    private Player[] terrorists;
    private Player[] counterTerrorists;
    private BombPlanting bombPlanting;

    public Round(Player[] terrorists, Player[] counterTerrorists) {
        this.terrorists = terrorists;
        this.counterTerrorists = counterTerrorists;
        this.bombPlanting = new BombPlanting(this);
    }

    public void startRound() {
        System.out.println("라운드가 시작되었습니다!");
    }

    public void distributeRewards(char winningTeam) {
        Player[] winningPlayers = (winningTeam == 'T') ? terrorists : counterTerrorists;
        Player[] losingPlayers = (winningTeam == 'T') ? counterTerrorists : terrorists;

        for (Player player : winningPlayers) {
            player.getMoney().add(3000); // 승자팀 자금 추가
        }

        for (Player player : losingPlayers) {
            player.getMoney().add(1900); // 패자팀 자금 추가
        }

        System.out.println((winningTeam == 'T' ? "테러리스트" : "대테러리스트") + " 팀이 라운드에서 승리하였습니다!");
        System.out.println("패자팀에게는 자금 $1900원이 지급되었습니다.");
        System.out.println("승자팀에게는 자금 $3000원이 지급되었습니다.");
    }
```

```java
    public boolean checkRoundEnd() {
        int remainingTerrorists = 0;
        int remainingCounterTerrorists = 0;

        for (Player player : terrorists) {
            if (player.getHealth() > 0) {
                remainingTerrorists++;
            }
        }

        for (Player player : counterTerrorists) {
            if (player.getHealth() > 0) {
                remainingCounterTerrorists++;
            }
        }

        if (bombPlanting.isBombPlanted() && remainingCounterTerrorists == 0) {
            System.out.println("대테러리스트가 전멸하였습니다. 폭탄이 폭발합니다!");
            bombPlanting.reduceBombTime(bombPlanting.getBombTimeRemaining());
        } else if (!bombPlanting.isBombPlanted() && remainingTerrorists == 0) {
            System.out.println("테러리스트가 전멸하였습니다. 대테러리스트 승리!");
            distributeRewards('C');
            return true;
        } else if (!bombPlanting.isBombPlanted() && remainingCounterTerrorists == 0) {
            System.out.println("대테러리스트가 전멸하였습니다. 테러리스트 승리!");
            distributeRewards('T');
            return true;
        }

        return false;
    }

    public void endRound() {
        System.out.println("라운드가 종료되었습니다!");
        System.exit(0);
    }

    public BombPlanting getBombPlanting() {
        return bombPlanting;
    }
}
```

Round 클래스

게임의 각 라운드를 관리하는 역할을 합니다. 이 클래슨느 테러리스트와 대 테러리스팀을 설정, 라운드 시작 및 종료, 승패 판정 과 보상지급 기능을 설정합니다. 폭탄 설치와 해체 상태를 관리하며 라운드 종료 조건을 판단합니다.

게임 코드 및 클래스 소개 – 6


```java
class Weapon {
    private String name;
    private int damage;
    private int headshotDamage;
    private int price;

    public Weapon(String name, int damage, int headshotDamage, int price) {
        this.name = name;
        this.damage = damage;
        this.headshotDamage = headshotDamage;
        this.price = price;
    }

    // 게터 메서드
    public String getName() {
        return name;
    }

    public int getDamage() {
        return damage;
    }

    public int getHeadshotDamage() {
        return headshotDamage;
    }

    public int getPrice() {
        return price;
    }
}
```

Weapon 클래스

게임 내 무기의 속성을 관리합니다. 무기의 이름과 기본데미지를 저장하고, 헤드샷데미지와 가격또한 저장을 진행하는 Weapon 클래스입니다.

게임 코드 및 클래스 소개 – 7

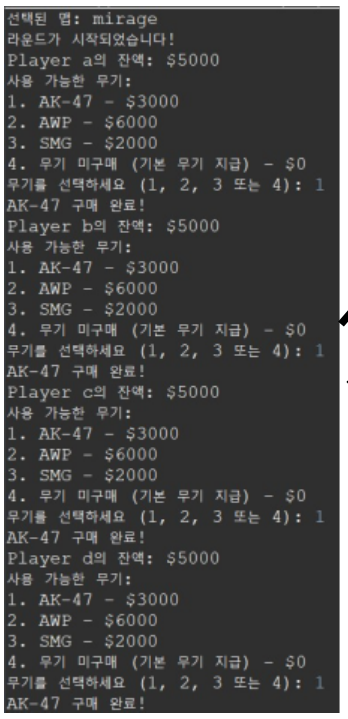
11

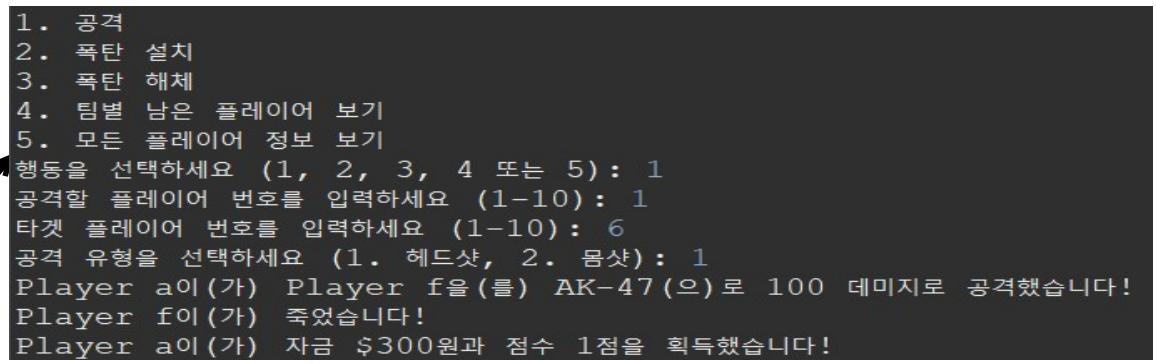
```java
// Main 클래스: 프로그램의 시작점
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // 맵 선택
        Map map = new Map();
        System.out.println("선택된 맵: " + map.getCurrentMap());

        // 플레이어 10명 생성 및 팀 분류
        Player[] players = new Player[10];
        Player[] terrorists = new Player[5];
        Player[] counterTerrorists = new Player[5];

        for (int i = 0; i < 10; i++) {
            char team = i < 5 ? 'T' : 'C';
            players[i] = new Player("Player " + (char) ('a' + i), team);
            if (i < 5) {
                terrorists[i] = players[i];
            } else {
                counterTerrorists[i - 5] = players[i];
            }
        }

        // 무기 생성
        Weapon ak47 = new Weapon("AK-47", 40, 100, 3000); // 체스상 데미지 100으로 수정
        Weapon awp = new Weapon("AWP", 80, 150, 6000);
        Weapon smg = new Weapon("SMG", 25, 70, 2000); // SMG 무기 추가
        Weapon glock = new Weapon("Glock", 20, 50, 0); // 테러리스트 기본 무기
        Weapon usp = new Weapon("USP", 25, 55, 0); // 대테러리스트 기본 무기

        // 게임 생성 및 시작
        Round round = new Round(terrorists, counterTerrorists);
        round.startRound();

        // 플레이어별 무기 구매
        for (Player player : players) {
            while (true) {
                System.out.println(player.getName() + "의 잔액: $" + player.getMoney().getBalance());
                System.out.println("무기 구매 목록:");
                System.out.println("1. AK-47 - $3000");
                System.out.println("2. AWP - $6000");
                System.out.println("3. SMG - $2000"); // SMG 무기 추가
                System.out.println("4. 무기 미구매 (기본 무기 지급) - $0");
                System.out.print("구매할 선택하세요 (1, 2, 3 또는 4): ");
                int choice = scanner.nextInt();
                if (choice == 1) {
                    if (player.getMoney().getBalance() >= ak47.getPrice()) {
                        player.setWeapon(ak47);
                        player.getMoney().subtract(ak47.getPrice());
                        System.out.println("AK-47 구매 완료!");
                        break;
                    } else {
                        System.out.println("AK-47을 구매할 자금이 부족합니다!");
                    }
                } else if (choice == 2) {
                    if (player.getMoney().getBalance() >= awp.getPrice()) {
                        player.setWeapon(awp);
                        player.getMoney().subtract(awp.getPrice());
                        System.out.println("AWP 구매 완료!");
                        break;
                    } else {
                        System.out.println("AWP를 구매할 자금이 부족합니다!");
                    }
                } else if (choice == 3) {
                    if (player.getMoney().getBalance() >= smg.getPrice()) {
                        player.setWeapon(smg);
                        player.getMoney().subtract(smg.getPrice());
                        System.out.println("SMG 구매 완료!");
                        break;
                    } else {
                        System.out.println("SMG를 구매할 자금이 부족합니다!");
                    }
                } else if (choice == 4) {
                    if (player.getTeam() == 'T') {
                        player.setWeapon(glock);
                        System.out.println("Glock 지급 완료!");
                    } else {
                        player.setWeapon(usp);
                        System.out.println("USP 지급 완료!");
                    }
                    break;
                } else {
                    System.out.println("잘못된 입력입니다. 다시 입력하세요.");
                }
            }
        }

        // 라운드 진행
        while (true) {
            System.out.println("1. 공격");
            System.out.println("2. 폭탄 설치");
            System.out.println("3. 폭탄 해체");
            System.out.println("4. 팀별 남은 플레이어 보기");
            System.out.println("5. 모든 플레이어 정보 보기");
            System.out.print("행동을 선택하세요 (1, 2, 3, 4 또는 5): ");
            int action = scanner.nextInt();
            if (action == 1) {
                // 공격 로직
                System.out.print("공격할 플레이어 번호를 입력하세요 (1-10): ");
                int attackerIndex = scanner.nextInt() - 1;
                System.out.print("대상 플레이어 번호를 입력하세요 (1-10): ");
                int targetIndex = scanner.nextInt() - 1;
                System.out.print("공격 유형을 선택하세요 (1. 헤드샷, 2. 바디): ");
                int attackType = scanner.nextInt();
                boolean isHeadshot = (attackType == 1);
                Attack.perform(players[attackerIndex], players[targetIndex], isHeadshot, round.getBombPlanting(), round);
            } else if (action == 2) {
                // 폭탄 설치 로직
                round.getBombPlanting().plantBomb();
            } else if (action == 3) {
                // 폭탄 해체 로직
                round.getBombPlanting().defuseBomb();
            } else if (action == 4) {
                // 팀별 남은 플레이어 목록 보기
                System.out.print("테러리스트 팀 남은 플레이어: ");
                for (Player player : terrorists) {
                    if (player.getHealth() > 0) {
                        System.out.print(player.getName() + " ");
                    }
                }
                System.out.println();

                System.out.print("대테러리스트 팀 남은 플레이어: ");
                for (Player player : counterTerrorists) {
                    if (player.getHealth() > 0) {
                        System.out.print(player.getName() + " ");
                    }
                }
                System.out.println();
            } else if (action == 5) {
                // 모든 플레이어 정보 보기
                for (Player player : players) {
                    player.displayInfo();
                    System.out.println();
                }
            }

            // 폭탄이 설치된 경우 남은 시간 표시
            if (round.getBombPlanting().isBombPlanted()) {
                long timeRemaining = round.getBombPlanting().getBombTimeRemaining();
                long seconds = (timeRemaining / 1000) % 60;
                long minutes = (timeRemaining / (1000 * 60)) % 60;
                System.out.println("폭탄이 폭발까지 남은 시간: " + minutes + "분 " + seconds + "초");
            }

            // 라운드 종료 조건 체크
            if (round.checkRoundEnd()) {
                round.endRound();
                break;
            }
        }
    }
}
```

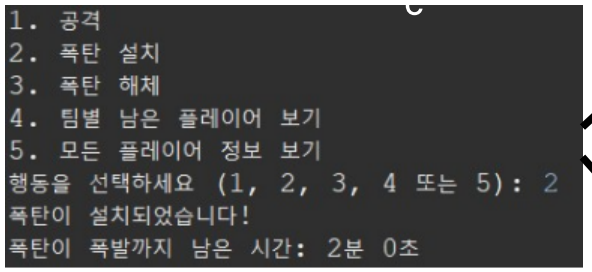
Main 클래스 (Game)

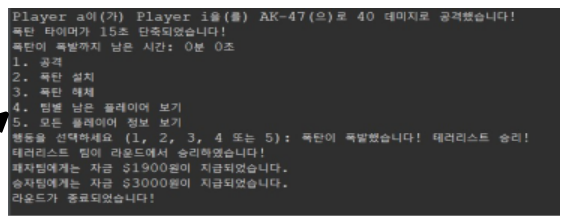
맵을 선택하고, 플레이어를 생성 및 팀 분류를 진행합니다. 무기 생성을 하고 각각의 속성을 설정합니다. 라운드를 시작하는 메소드는 입니다. 이것은 기존에 있던 코드와 동일합니다. 플레이어별 무기 구매 목록을 추가하였습니다. 플레이어는 주어진 초기 자금을 통해 무기 구매를 진행하고 그 무기가 갖고 있는 가격에 대해 플레이어가 갖고 있는 자금이 충분한지 확인하고 무기 구매를 승인합니다. 또한 새로 생긴 라운드 진행 부분입니다. 사용자가 공격, 폭탄 설치, 폭탄 해체, 팀별 남은 플레이어보기, 모든 플레이어 정보등을 콘솔로 입력받아 게임을 진행합니다. 또한 폭탄이 설치된 경우 남은 시간도 표시하고 라운드 종료 조건을 체크해 전반적인 게임 진행 상황을 관리합니다.

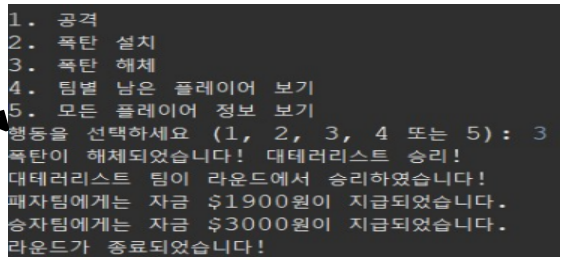
게임 진행 (콘솔)


선택된 맵: mirage
라운드가 시작되었습니다!
Player a의 잔액: $5000
사용 가능한 무기:
1. AK-47 - $3000
2. AWP - $6000
3. SMG - $2000
4. 무기 미구매 (기본 무기 지급) - $0
무기를 선택하세요 (1, 2, 3 또는 4): 1
AK-47 구매 완료!
Player b의 잔액: $5000
사용 가능한 무기:
1. AK-47 - $3000
2. AWP - $6000
3. SMG - $2000
4. 무기 미구매 (기본 무기 지급) - $0
무기를 선택하세요 (1, 2, 3 또는 4): 1
AK-47 구매 완료!
Player c의 잔액: $5000
사용 가능한 무기:
1. AK-47 - $3000
2. AWP - $6000
3. SMG - $2000
4. 무기 미구매 (기본 무기 지급) - $0
무기를 선택하세요 (1, 2, 3 또는 4): 1
AK-47 구매 완료!
Player d의 잔액: $5000
사용 가능한 무기:
1. AK-47 - $3000
2. AWP - $6000
3. SMG - $2000
4. 무기 미구매 (기본 무기 지급) - $0
무기를 선택하세요 (1, 2, 3 또는 4): 1
AK-47 구매 완료!


1. 공격
2. 폭탄 설치
3. 폭탄 해체
4. 팀별 남은 플레이어 보기
5. 모든 플레이어 정보 보기
행동을 선택하세요 (1, 2, 3, 4 또는 5): 1
공격할 플레이어 번호를 입력하세요 (1-10): 1
타겟 플레이어 번호를 입력하세요 (1-10): 6
공격 유형을 선택하세요 (1. 헤드샷, 2. 몸샷): 1
Player a이(가) Player f을(를) AK-47(으)로 100 데미지로 공격했습니다!
Player f이(가) 죽었습니다!
Player a이(가) 자금 $300원과 점수 1점을 획득했습니다!


1. 공격
2. 폭탄 설치
3. 폭탄 해체
4. 팀별 남은 플레이어 보기
5. 모든 플레이어 정보 보기
행동을 선택하세요 (1, 2, 3, 4 또는 5): 2
폭탄이 설치되었습니다!
폭탄이 폭발까지 남은 시간: 2분 0초


Player a이(가) Player i을(를) AK-47(으)로 40 데미지로 공격했습니다!
폭탄 타이머가 15초 단축되었습니다!
폭탄이 폭발까지 남은 시간: 0분 0초
1. 공격
2. 폭탄 설치
3. 폭탄 해체
4. 팀별 남은 플레이어 보기
5. 모든 플레이어 정보 보기
행동을 선택하세요 (1, 2, 3, 4 또는 5): 폭탄이 폭발했습니다! 테러리스트 승리!
테러리스트 팀이 라운드에서 승리하였습니다!
패자팀에게는 자금 $1900원이 지급되었습니다.
승자팀에게는 자금 $3000원이 지급되었습니다.
라운드가 종료되었습니다!


1. 공격
2. 폭탄 설치
3. 폭탄 해체
4. 팀별 남은 플레이어 보기
5. 모든 플레이어 정보 보기
행동을 선택하세요 (1, 2, 3, 4 또는 5): 3
폭탄이 해체되었습니다! 대테러리스트 승리!
대테러리스트 팀이 라운드에서 승리하였습니다!
패자팀에게는 자금 $1900원이 지급되었습니다.
승자팀에게는 자금 $3000원이 지급되었습니다.
라운드가 종료되었습니다!

게임 진행 (콘솔) -2

전원 처치로 인한 라운드 종료

```
1. 공격
2. 폭탄 설치
3. 폭탄 해체
4. 팀별 남은 플레이어 보기
5. 모든 플레이어 정보 보기
행동을 선택하세요 (1, 2, 3, 4 또는 5): 1
공격할 플레이어 번호를 입력하세요 (1-10): 1
타겟 플레이어 번호를 입력하세요 (1-10): 6
공격 유형을 선택하세요 (1. 헤드샷, 2. 몸샷): 1
Player a이(가) Player f을(를) AK-47(으)로 100 데미지로 공격했습니다!
Player f이(가) 죽었습니다!
Player a이(가) 자금 $300원과 점수 1점을 획득했습니다!
```

→

```
1. 공격
2. 폭탄 설치
3. 폭탄 해체
4. 팀별 남은 플레이어 보기
5. 모든 플레이어 정보 보기
행동을 선택하세요 (1, 2, 3, 4 또는 5): 1
공격할 플레이어 번호를 입력하세요 (1-10): 1
타겟 플레이어 번호를 입력하세요 (1-10): 10
공격 유형을 선택하세요 (1. 헤드샷, 2. 몸샷): 1
Player a이(가) Player j을(를) AK-47(으)로 100 데미지로 공격했습니다!
Player j이(가) 죽었습니다!
Player a이(가) 자금 $300원과 점수 1점을 획득했습니다!
대테러리스트가 전멸하였습니다. 테러리스트 승리!
테러리스트 팀이 라운드에서 승리하였습니다!
패자팀에게는 자금 $1900원이 지급되었습니다.
승자팀에게는 자금 $3000원이 지급되었습니다.
라운드가 종료되었습니다!
```

질문