다마고치 게임 구현 보고서

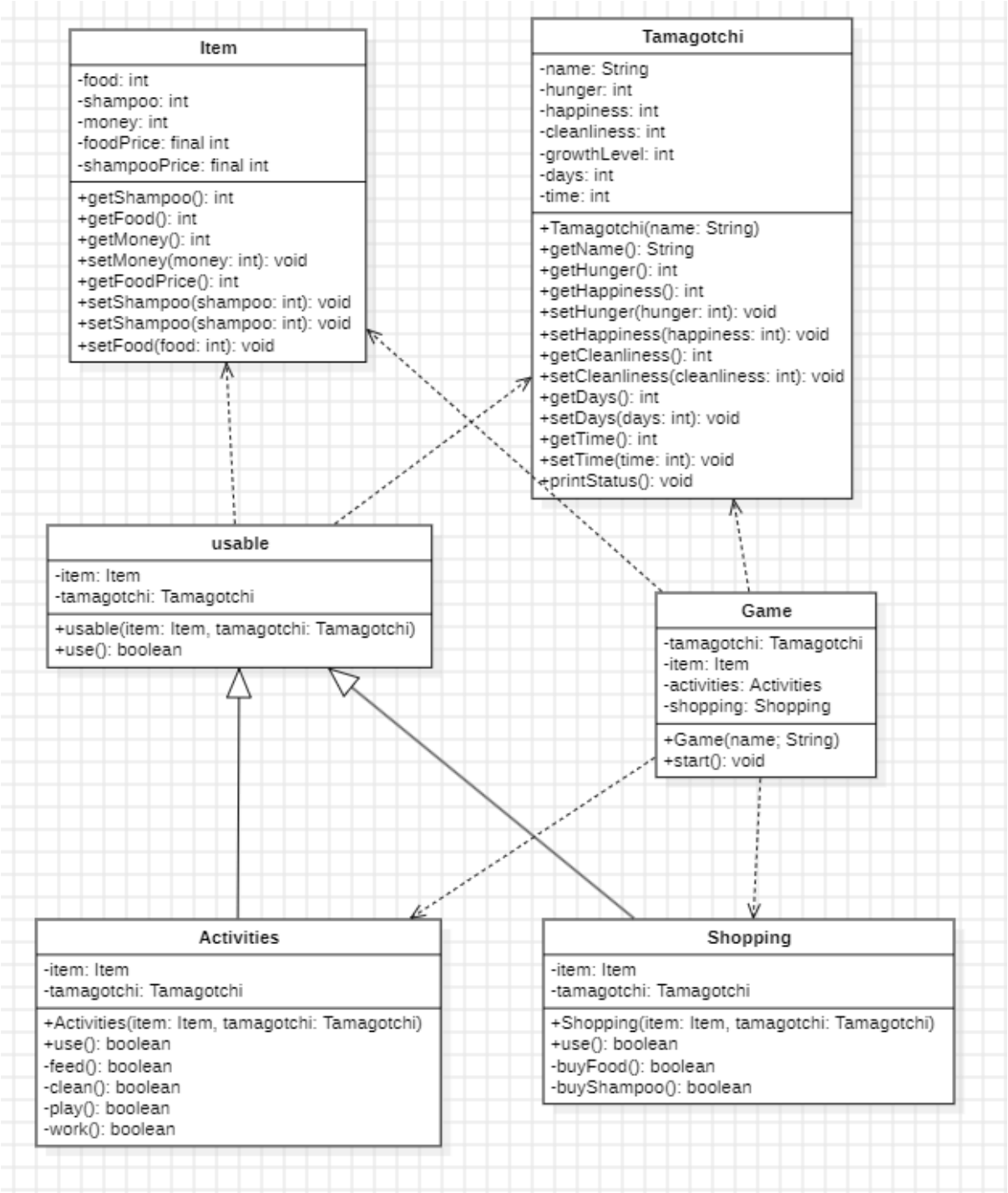
202111481 김찬영

옛날에 재밌게 했었던 '다마고치' 게임을 구현해 보기로 했다. '다마고치'는 가상의 애완 동물을 키우는 시뮬레이션 게임으로, 사용자는 다마고치의 상태를 관리하고 먹이를 주거나 놀아주는 등의 동작을 수행하여 다마고치를 키우는 게임이다. 내가 구현해 볼 게임은 다마고치가 성장하여 레벨이 올라가면서를 행복도나 배고픔, 청결도 수치가 일정 이하로 떨어지지 않고 (게임오버) 최종 성장 단계 (레벨 5)에 도달했을 때 우승 메세지 출력하며 게임을 종료하는 프로그램을 만들기로 했다. 사용자는 다마고치가 최종 성장을 할 때까지 사용자가 다마고치의 상태를 확인하고 관리해줘야 한다.

다마고치 게임에 필요한 요구사항들은 다음과 같이 나타낼 수 있다.
1. 다마고치 상태 관리: 게임은 사용자가 키우는 다마고치의 상태를 관리해야 한다. 상태는 배고픔, 행복도, 청결도, 성장 레벨 등을 포함한다.
2. 먹이 주기: 사용자는 다마고치에게 먹이를 주어서 배고픔, 행복도 상태를 관리할 수 있어야 한다.
3. 놀아주기: 사용자는 다마고치와 놀아주기로 행복도를 증가시킬 수 있어야 한다.
4. 탐색하기: 사용자는 다마고치를 관리하는 물품을 구매하기 위해 다마고치와 탐색이라는 것을 하여 랜덤하게 돈을 벌 수 있어야 한다.
5. 청결 상태 관리: 사용자는 다마고치를 목욕시켜 청결도와 행복도 상태를 관리할 수 있어야 한다.
6. 쇼핑하기: 다마고치의 먹이를 사거나 청결도 관리를 위한 물품을 살 수 있어야 한다.
7. 게임 종료 조건: 게임은 다마고치의 상태가 일정 수준 이하로 떨어졌을 때 또는 다마고치가 최종 성장을 마쳤을 때 게임이 종료되어야 한다.
8. 상호 작용: 사용자는 콘솔에서 키 입력을 통해 다마고치와 상호 작용할 수 있어야 한다. 키를 입력하여 먹이를 주거나 놀아줄 수 있어야 한다.
9. 시간 경과: 게임은 시간이 경과함에 따라 다마고치의 상태가 변화해야 한다. 예를 들어, 일정 시간이 지남에 따라 배고픔이 증가하거나 청결도가 감소할 수 있다.

이 요구사항을 토대로 Activities 클래스를 만들어 feed, play, clean, work의 기능이 다 포함된 클래스를 만들고, usable 추상 클래스를 만들어 Activities, Shopping 클래스가 상속받게 하였다. 또한 물품, 돈을 관리할 수 있는 Item 클래스를 만들어 보았다.

클래스 다이어그램으로 나타내면 이렇다.



1.Game 클래스는 Tamagotchi, Item, Activities, Shopping 클래스를 참조하므로 의존 관계
2.usable 클래스는 Tamagotchi, Item 클래스를 참조하므로 의존 관계
3.Activities, Shopping 클래스는 usable 클래스를 상속받으므로 일반화 관계

1.Tamagotchi 클래스

```
package tamagotchiGame;

public class Tamagotchi {
   private String name;
   private int hunger;
   private int happiness;
   private int cleanLiness;
   private int growthLevel;
   private int days; // 날짜
   private int time; // 시간

   public Tamagotchi(String name) { // 생성자, 이름을 인자로 받아 다마고치 객체
생성
       this.name = name;
       this.hunger = 50;
       this.happiness = 50;
       this.cleanLiness = 50;
       this.growthLevel = 1;
       this.days = 0;
       this.time = 1;
   }

   public String getName() {
       return name;
   }

   public int getHunger() {
       return hunger;
   }

   public int getHappiness() {
       return happiness;
   }

   public void setHunger(int hunger) {
       this.hunger = hunger;
   }

   public void setHappiness(int happiness) {
       this.happiness = happiness;
   }

   public int getCleanLiness() {
       return cleanLiness;
   }

   public void setCleanLiness(int cleanliness) {
       this.cleanLiness = cleanliness;
```

```
    }

    public int getDays() {
        return days;
    }

    public void setDays(int days) {
        this.days = days;
    }

    public int getTime() {
        return time;
    }

    public void setTime(int time) {
        this.time = time;
    }

    public int getGrowthLevel() {
        return growthLevel;
    }

    public void setGrowthLevel(int growthLevel) {
        this.growthLevel = growthLevel;
    }

    public void printStatus(){
        System.out.println(name + "의 상태");
        System.out.println("--------------------");
        System.out.println("배고픔: " + hunger + " 행복도: " + happiness + "
청결도: " + cleanLiness + " 레벨: " + growthLevel);
    }

}
```

다마고치의 상태 (이름, 배고픔, 행복도, 청결도, 성장 레벨)를 관리하기 위한 변수와 게임 내 시간을 조절하는 변수들을 추가했다. 생성자는 다마고치의 객체를 초기화하고, 이름을 인자로 받아 초기값으로 배고픔, 행복도, 청결도, 성장 레벨, 시간들을 기본 값으로 설정한다. 또한 getter와 setter 메서드를 통해 외부에서 private 변수에 접근할 수 있게끔 하고, printStatus() 메서드를 통해 Game 클래스에서 다마고치의 상태를 출력하게끔 만들었다.

2.Item 클래스

```
package tamagotchiGame;

public class Item {
    private int food = 5;
    private int shampoo = 1;
    private int money = 0;
    private final int foodPrice = 1000;
```

```
    private final int shampooPrice = 2000;

    public int getShampoo() {
        return shampoo;
    }

    public int getFood() {
        return food;
    }

    public int getMoney() {
        return money;
    }

    public void setMoney(int money){
        this.money = money;
    }

    public int getFoodPrice() {
        return foodPrice;
    }

    public int getShampooPrice() {
        return shampooPrice;
    }

    public void setShampoo(int shampoo) {
        this.shampoo = shampoo;
    }

    public void setFood(int food) {
        this.food = food;
    }
}
```

다마고치의 상태를 관리하기 위한 음식과 샴푸, 돈 변수를 추가하였고 음식과 샴푸의 값을 final로 고정하였다. 그리고 getter와 setter 메서드를 통해 외부에서 private 변수에 접근할 수 있게끔 하였다.

3.usable 추상 클래스

```
package tamagotchiGame;

public abstract class usable {
    protected Item item;
    protected Tamagotchi tamagotchi;

    public usable (Item item, Tamagotchi tamagotchi) {
        this.item = item;
        this.tamagotchi = tamagotchi;
```

```
    }
    public abstract boolean use();
}
```

Activities, Shopping 클래스가 Tamagotchi, Item 클래스를 참조하고, boolean을 이용한 선택지도 포함되어 있으므로 상속 관계를 통해 중복을 줄였다.

4.Activities 클래스

```
package tamagotchiGame;

import java.util.Scanner;

public class Activities extends usable{
    public Activities(Item item, Tamagotchi tamagotchi) {
        super(item, tamagotchi);
    }

    @Override
    public boolean use() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("어떤 행동을 하시겠습니까? (1: 먹이 주기, 2: 목욕시키기. 3: 놀아주기, 4: 탐색 하기)");
        int choice = scanner.nextInt();
        switch (choice) {
            case 1:
                return feed();
            case 2:
                return clean();
            case 3:
                tamagotchi.setTime(tamagotchi.getTime() + 1);
                return play();
            case 4:
                if (tamagotchi.getTime() != 1) {
                    System.out.println("탐색은 아침에만 할 수 있습니다.");
                    return false;
                }
                tamagotchi.setTime(tamagotchi.getTime() + 2);
                return work();
            default:
                System.out.println("올바른 선택이 아닙니다.");
                return false;
        }
    }

    private boolean feed() {
        if (item.getFood() <= 0) {
            System.out.println("먹이가 없습니다. ");
            return false;
        } else {
            System.out.println("당신은 " + tamagotchi.getName() + "에게 먹이를
```

```
주었습니다.");
            item.setFood(item.getFood() - 1);
            System.out.println("남은 먹이 개수: " + item.getFood());

            tamagotchi.setHunger(tamagotchi.getHunger() - 30);
            if (tamagotchi.getHunger() < 0) {
                tamagotchi.setHunger(0);
            }
            tamagotchi.setHappiness(tamagotchi.getHappiness() + 1);
            tamagotchi.setCleanLiness(tamagotchi.getCleanLiness() - 5);
            tamagotchi.setTime(tamagotchi.getTime() + 1);
            return true;
        }
    }

    private boolean clean() {
        if (item.getShampoo() <= 0) {
            System.out.println("샴푸가 없습니다. ");
            return false;
        } else {
            System.out.println("당신은 " + tamagotchi.getName() + "를
목욕시켰습니다.");
            item.setShampoo(item.getShampoo() - 1);
            System.out.println("남은 샴푸 개수: " + item.getShampoo());

            tamagotchi.setHunger(tamagotchi.getHunger() + 5);
            tamagotchi.setHappiness(tamagotchi.getHappiness() + 1);
            tamagotchi.setCleanLiness(tamagotchi.getCleanLiness() + 50);
            if (tamagotchi.getCleanLiness() > 100) {
                tamagotchi.setCleanLiness(100);
            }
            tamagotchi.setTime(tamagotchi.getTime() + 1);
            return true;
        }
    }

    private boolean play() {
        System.out.println("당신은 " + tamagotchi.getName() + "를
놀아주었습니다.");
        tamagotchi.setHappiness(tamagotchi.getHappiness() + 5);
        tamagotchi.setCleanLiness(tamagotchi.getCleanLiness() - 10);
        tamagotchi.setHunger(tamagotchi.getHunger() + 10);
        return true;
    }

    private boolean work() {
        System.out.println("당신은" + tamagotchi.getName() + "와 탐색을 하여 돈을
벌었습니다!");
        item.setMoney(item.getMoney() + (int)(Math.random()*7000+1000));
        System.out.println("가진 돈: " + item.getMoney());
        tamagotchi.setHunger(tamagotchi.getHunger() + 10);
```

```
        tamagotchi.setHappiness(tamagotchi.getHappiness() - 6);
        tamagotchi.setCleanLiness(tamagotchi.getCleanLiness() - 10);
        return true;
    }
}
```

usable 추상 클래스를 상속받아 중복을 줄이고, use() 메서드를 통해 게임을 하면서 선택지가 출력되게끔 하였다. 1번을 누르면 private로 선언된 feed() 메서드를 불러와 item 클래스의 food 변수를 통해 개수를 차감하고 다마고치의 배고픔을 줄이고 행복도를 올리게끔 코드를 추가하였다. 2번을 선택하면 private로 선언된 clean() 메서드를 불러와 item 클래스의 shampoo 변수를 통해 개수를 차감하고 다마고치의 청결도와 행복도를 올리게끔 코드를 추가하였다. 3번을 선택하면 private로 선언된 play() 메서드를 불러와 다마고치의 행복도를 올리게끔 만들었고, 4번을 선택하면 private로 선언된 work() 메서드를 불러와 다마고치와 탐색하여 랜덤하게 돈을 얻어 Item 클래스의 money 변수의 값을 올리는 코드를 추가하였다. 또한 각 선택지를 선택할 때 정상적으로 작동되면 Tamagotchi 클래스의 time 변수의 값이 올라가게끔 만들었다.

5.Shopping 클래스

```
package tamagotchiGame;

import java.util.Scanner;

public class Shopping extends usable{
    public Shopping (Item item, Tamagotchi tamagotchi) {
        super(item, tamagotchi);
    }

    @Override
    public boolean use(){
        if (tamagotchi.getTime() == 3) {
            System.out.println("쇼핑은 아침, 점심에만 할 수 있습니다.");
            return false;
        }
        Scanner scanner = new Scanner(System.in);
        System.out.println("무엇을 사겠습니까? (1: 먹이, 2: 샴푸)");
        int choice = scanner.nextInt();
        switch (choice) {
            case 1:
                return buyFood();
            case 2:
                return buyShampoo();
            default:
                System.out.println("올바른 선택이 아닙니다.");
                return false;
        }
    }

    private boolean buyFood(){
        Scanner scanner = new Scanner(System.in);
        System.out.println("현재 돈: " + item.getMoney());
```

```
        System.out.println("먹이 가격: " + item.getFoodPrice());
        System.out.println("몇 개를 구매하시나요? ");
        int count = scanner.nextInt();

        if (item.getMoney() - item.getFoodPrice() * count < 0) {
            System.out.println("돈이 부족합니다. ");
            return false;
        } else {
            item.setMoney(item.getMoney() - item.getFoodPrice() * count);
            item.setFood(item.getFood() + count);
            System.out.print("현재 먹이 개수: " + item.getFood());
            System.out.println(", 남은 돈: " + item.getMoney());
            tamagotchi.setTime(tamagotchi.getTime() + 1);
            tamagotchi.setHunger(tamagotchi.getHunger() + 5);
            tamagotchi.setHappiness(tamagotchi.getHappiness() - 3);
            tamagotchi.setCleanLiness(tamagotchi.getCleanLiness() - 5);
            return true;
        }
    }

    private boolean buyShampoo() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("현재 돈: " + item.getMoney());
        System.out.println("샴푸 가격: " + item.getShampooPrice());
        System.out.println("몇 개를 구매하시나요? ");
        int count = scanner.nextInt();

        if (item.getMoney() - item.getShampooPrice() * count < 0) {
            System.out.println("돈이 부족합니다. ");
            return false;
        } else {
            item.setMoney(item.getMoney() - item.getShampooPrice() * count);
            item.setShampoo(item.getShampoo() + count);
            System.out.print("현재 샴푸 개수: " + item.getShampoo());
            System.out.println(", 남은 돈: " + item.getMoney());
            tamagotchi.setTime(tamagotchi.getTime() + 1);
            tamagotchi.setHunger(tamagotchi.getHunger() + 5);
            tamagotchi.setHappiness(tamagotchi.getHappiness() - 3);
            tamagotchi.setCleanLiness(tamagotchi.getCleanLiness() - 5);
            return true;
        }
    }
}
```

Activities 클래스와 마찬가지로 usable 추상 클래스를 상속받아 중복을 줄이고, use() 메서드를 통해 게임을 하면서 선택지가 출력되게끔 하였다. 1번과 2번 중에 선택하여 먹이나 샴푸를 원하는 만큼 구매하고 구매한 만큼 Item 클래스 내에 있는 변수의 값이 바뀌게 하고, Tamagotchi 클래스 내에 있는 time 변수가 늘어나게끔 만들었다.

6.Game 클래스

```
package tamagotchiGame;

import java.util.Scanner;

public class Game {
   // 객체 참조 코드들
   private Tamagotchi tamagotchi;
   private Item item;
   private Activities activities;
   private Shopping shopping;


   public Game(String name) {
       this.tamagotchi = new Tamagotchi(name); // name 인자로 받는 Tamagotchi
객체 생성
       this.item = new Item(); // Item 객체 생성
       this.activities = new Activities(this.item, this.tamagotchi);
       this.shopping = new Shopping(this.item, this.tamagotchi);
   }

   public void start() {
       Scanner scanner = new Scanner(System.in);
       boolean gameOver = false;

       while (!gameOver) {
           tamagotchi.printStatus(); // tamagotchi 상태 출력
           System.out.println("경과날짜: " + tamagotchi.getDays() + "일");
           // GameOver 로직
           if (tamagotchi.getTime() == 1) {
               System.out.println("현재시간: 아침");
           } else if (tamagotchi.getTime() == 2) {
               System.out.println("현재시간: 점심");
           } else {
               System.out.println("현재시간: 저녁");
           }

           System.out.println("1. 행동하기, 2. 쇼핑하기 (아침, 점심에만 가능),
3.게임 설명, 4. 게임 종료");
           System.out.print("무엇을 하실 건가요?: ");
           int choice = scanner.nextInt();

           switch (choice) {
               case 1:
                   if (activities.use()){
                       break;
                   } else {
                       continue;
                   }
               case 2:
```

```
                    if (shopping.use()) {
                        break;
                    } else {
                        continue;
                    }
                case 3:
                    System.out.println("다마고치 게임에 오신 것을 환영합니다! 이
게임은 플레이어가 여러 행동을 통해 다마고치를 최종 성장 (레벨 5까지)시키는 것이
목표입니다.");
                    System.out.println("다마고치는 5일에 1번씩 성장을 합니다.
다마고치는 행복도, 청결도 수치가 하나라도 0이 되거나 배고픔 수치가 100이 넘으면
죽습니다. ");
                    System.out.println("1번 선택지로 다마고치에게 먹이를 주거나,
목욕을 시키거나, 놀아주거나, 탐색을 할 수 있습니다. ");
                    System.out.println("탐색은 다마고치와 밖을 다니면서 얻은
물건들을 팔아 1000~8000원 랜덤하게 돈을 얻을 수 있습니다. 탐색은 아침에만 할 수
있습니다. ");
                    System.out.println("2번 선택지로 먹이나 다마고치 샴푸를 살 수
있습니다. 처음에 먹이는 5개, 샴푸는 1개를 가지고 있습니다. ");
                    break;
                case 4:
                    gameOver = true;
                    break;
                default:
                    System.out.println("잘못된 선택지입니다. 다시 입력하세요.");
            }
            if (tamagotchi.getTime() == 4) { // 저녁에서 다음 시간이 경과할 시 다음
날짜
                tamagotchi.setDays(tamagotchi.getDays() + 1);
                tamagotchi.setTime(1);
            }
            if (tamagotchi.getCleanLiness() <= 0 || tamagotchi.getHunger() >= 100
|| tamagotchi.getHappiness() <= 0){
                System.out.println(tamagotchi.getName() + "가 죽었습니다.");
                gameOver = true;
            }
            switch (tamagotchi.getDays()){
                case 5:
                    if (tamagotchi.getGrowthLevel() == 2)
                        break;
                    tamagotchi.setGrowthLevel(tamagotchi.getGrowthLevel() + 1);
                    System.out.println(tamagotchi.getName() + "가
성장하였습니다!");
                    break;
                case 10:
                    if (tamagotchi.getGrowthLevel() == 3)
                        break;
                    tamagotchi.setGrowthLevel(tamagotchi.getGrowthLevel() + 1);
                    System.out.println(tamagotchi.getName() + "가
성장하였습니다!");
                    break;
```

```
                case 15:
                    if (tamagotchi.getGrowthLevel() == 4)
                        break;
                    tamagotchi.setGrowthLevel(tamagotchi.getGrowthLevel() + 1);
                    System.out.println(tamagotchi.getName() + "가
성장하였습니다!");
                    break;
                case 20:
                    tamagotchi.setGrowthLevel(tamagotchi.getGrowthLevel() + 1);
                    System.out.println(tamagotchi.getName() + "가 최종 성장을
하였습니다! 축하합니다!");
                    gameOver = true;
                    break;
            }
        }
        scanner.close();
    }
}
```

다마고치 게임을 진행하기 위한 클래스이다. Game 클래스 내에서 사용할 수 있는 객체를 참조하기 위한 변수들을 선언하고, 생성자를 통해 객체들을 선언한다. 그리고 1번 선택지를 선택하면 Activities 클래스의 use 메서드를, 2번 선택지를 클릭하면 Shopping 클래스의 use 메서드를, 3번 선택지는 게임의 설명을 출력하고, 4번 선택지를 선택하면 게임 종료를 하게끔 만들었다. 그리고 Tamagotchi 클래스의 time과 days 변수를 불러와 만약 time 변수가 3이 넘어가면 days 변수의 값을 1 늘리게 하였고, switch문을 통해 days 변수가 5, 10, 15, 20이 될 때 Tamagotchi 클래스의 성장 레벨 변수의 값을 올리고 만약 레벨 5가 되면 다마고치가 최종 성장을 마쳐 게임이 종료되게끔 만들었다.

7.main 클래스

```
package tamagotchiGame;

import java.util.Scanner;

public class main {
    public static void main(String[] args) {
        String name; // 다마고치 이름 변수
        Scanner scanner = new Scanner(System.in);
        System.out.println("다마고치 게임 실행");
        System.out.print("다마고치의 이름을 입력해주세요: ");
        name = scanner.nextLine();

        Game game = new Game(name); // 다마고치 이름을 인자로 받는 game 객체 생성
        System.out.println("게임 시작!");
        game.start(); // 게임 시작 함수
    }
}
```

마지막으로 main 클래스는 name을 입력받고 그걸 인자로 받는 game 객체를 생성하여 다마고치의 이름을 생성자를 통해 설정하고, start() 함수를 통해 게임을 실행한다.

프로그램 실행 분석

첫 시작

```
다마고치 게임 실행
다마고치의 이름을 입력해주세요: 제니
게임 시작!
제니의 상태
----------------------
배고픔: 50 행복도: 50 청결도: 50 레벨: 1
경과날짜: 0일
현재시간: 아침
1. 행동하기, 2. 쇼핑하기 (아침, 점심에만 가능), 3.게임 설명, 4. 게임 종료
무엇을 하실 건가요?:
```

```
1. 행동하기, 2. 쇼핑하기 (아침, 점심에만 가능), 3.게임 설명, 4. 게임 종료
무엇을 하실 건가요?: 1
어떤 행동을 하시겠습니까? (1: 먹이 주기, 2: 목욕시키기. 3: 놀아주기, 4: 탐색 하기)
```

```
무엇을 하실 건가요?: 2
무엇을 사겠습니까? (1: 먹이, 2: 샴푸)
```

```
무엇을 하실 건가요?: 3
다마고치 게임에 오신 것을 환영합니다! 이 게임은 플레이어가 여러 행동을 통해 다마고치를 최종 성장 (레벨 5까지)시키는 것이 목표입니다.
다마고치는 5일에 1번씩 성장을 합니다. 다마고치는 행복도, 청결도 수치가 하나라도 0이 되거나 배고픔 수치가 100이 넘으면 죽습니다.
1번 선택지로 다마고치에게 먹이를 주거나, 목욕을 시키거나, 놀아주거나, 탐색을 할 수 있습니다.
탐색은 다마고치와 밖을 다니면서 얻은 물건들을 팔아 1000~8000원 랜덤하게 돈을 얻을 수 있습니다. 탐색은 아침에만 할 수 있습니다.
2번 선택지로 먹이나 다마고치 샴푸를 살 수 있습니다. 처음에 먹이는 5개, 샴푸는 1개를 가지고 있습니다.
1. 행동하기, 2. 쇼핑하기 (아침, 점심에만 가능), 3.게임 설명, 4. 게임 종료
무엇을 하실 건가요?: 4


종료 코드 0(으)로 완료된 프로세스
```

첫 시작하면 다마고치의 이름을 입력받고, 선택할 수 있는 선택지가 주어진다. 행동하기를 선택하면 Activities 클래스의 먹이주기(feed), 목욕시키기(clean), 놀아주기(play), 탐색하기(work) 활동을 할 수 있고, 쇼핑하기를 선택하면 Shopping 클래스의 buyFood, buyShampoo 메서드를 사용, 3번 선택지를 선택하면 게임의 설명문을 출력하고 4번 선택지를 선택하면 게임이 종료되게끔 하였다.

행동하기 로직들

```
배고픔: 60 행복도: 44 청결도: 40 레벨: 1
경과날짜: 0일
현재시간: 저녁
1. 행동하기, 2. 쇼핑하기 (아침, 점심에만 가능), 3.게임 설명, 4. 게임 종료
무엇을 하실 건가요?: 1
어떤 행동을 하시겠습니까? (1: 먹이 주기, 2: 목욕시키기. 3: 놀아주기, 4: 탐색 하기)
1
당신은 제니에게 먹이를 주었습니다.
남은 먹이 개수: 4
제니의 상태
--------------------
배고픔: 30 행복도: 45 청결도: 35 레벨: 1
경과날짜: 1일
현재시간: 아침
```

```
배고픔: 30 행복도: 45 청결도: 35 레벨: 1
경과날짜: 1일
현재시간: 아침
1. 행동하기, 2. 쇼핑하기 (아침, 점심에만 가능), 3.게임 설명, 4. 게임 종료
무엇을 하실 건가요?: 1
어떤 행동을 하시겠습니까? (1: 먹이 주기, 2: 목욕시키기. 3: 놀아주기, 4: 탐색 하기)
2
당신은 제니를 목욕시켰습니다.
남은 샴푸 개수: 0
제니의 상태
--------------------
배고픔: 35 행복도: 46 청결도: 85 레벨: 1
경과날짜: 1일
현재시간: 점심
```

```
배고픔: 35 행복도: 46 청결도: 85 레벨: 1
경과날짜: 1일
현재시간: 점심
1. 행동하기, 2. 쇼핑하기 (아침, 점심에만 가능), 3.게임 설명, 4. 게임 종료
무엇을 하실 건가요?: 1
어떤 행동을 하시겠습니까? (1: 먹이 주기, 2: 목욕시키기. 3: 놀아주기, 4: 탐색 하기)
3
당신은 제니를 놀아주었습니다.
제니의 상태
--------------------
배고픔: 45 행복도: 51 청결도: 75 레벨: 1
경과날짜: 1일
현재시간: 저녁
```

```
배고픔: 50 행복도: 50 청결도: 50 레벨: 1
경과날짜: 0일
현재시간: 아침
1. 행동하기, 2. 쇼핑하기 (아침, 점심에만 가능), 3.게임 설명, 4. 게임 종료
무엇을 하실 건가요?: 1
어떤 행동을 하시겠습니까? (1: 먹이 주기, 2: 목욕시키기. 3: 놀아주기, 4: 탐색 하기)
4
당신은제니와 탐색을 하여 돈을 벌었습니다!
가진 돈: 7399
```

행동하기에서 1~4번 선택지를 선택하면 나오는 화면들이다. 먹이 주기를 클릭하면 남은 먹이 개수를 출력하고, 다마고치의 배고픔을 낮추고 행복도를 올린 후 최종 상태를 출력한다. 목욕시키기도 먹이 주기와 비슷하게 청결도를 올리고 행복도를 올린 후 최종 상태를 출력한다. 놀아주기를 선택하면 다마고치의 행복도를 올리고, 탐색하기를 선택하면 랜덤으로 돈을 얻고 가진 돈을 출력한다.

쇼핑하기 로직들

```
무엇을 하실 건가요?: 2
무엇을 사겠습니까? (1: 먹이, 2: 샴푸)
1
현재 돈: 7399
먹이 가격: 1000
몇 개를 구매하시나요?
3
현재 먹이 개수: 7, 남은 돈: 4399
```

```
무엇을 하실 건가요?: 2
무엇을 사겠습니까? (1: 먹이, 2: 샴푸)
2
현재 돈: 4399
샴푸 가격: 2000
몇 개를 구매하시나요?
2
현재 샴푸 개수: 2, 남은 돈: 399
```

쇼핑하기를 클릭하고 1~2번을 선택하면 현재 돈과 그 상품의 가격을 출력하고 몇 개를 구매할 건지 입력받고 구매를 한 후의 상품 개수와 남은 돈을 출력한다.

게임 오버

```
배고픔: 90 행복도: 57 청결도: 30 레벨: 1
경과날짜: 3일
현재시간: 저녁
1. 행동하기, 2. 쇼핑하기 (아침, 점심에만 가능), 3.게임 설명, 4. 게임 종료
무엇을 하실 건가요?: 1
어떤 행동을 하시겠습니까? (1: 먹이 주기, 2: 목욕시키기. 3: 놀아주기, 4: 탐색 하기)
3
당신은 제니 를 놀아주었습니다.
제니 가 죽었습니다.
```

다마고치 게임을 하면서 선택지를 선택하면 시간이 늘어나고, 그에 따라 상태도 감소하는데 (배고픔, 청결도, 행복도) 만약 배고픔이 100이 넘거나, 행복도가 0이하가 되거나, 청결도가 0 이하가 되면 다마고치가 죽었다는 메세지와 함께 게임이 종료된다.

레벨 성장

```
제니가 성장하였습니다!
제니의 상태
----------------------
배고픔: 30 행복도: 41 청결도: 75 레벨: 2
경과날짜: 5일
현재시간: 아침
```

```
제니가 성장하였습니다!
제니의 상태
---------------------
배고픔: 60 행복도: 48 청결도: 30 레벨: 3
경과날짜: 10일
현재시간: 아침
```

```
제니가 성장하였습니다!
제니의 상태
---------------------
배고픔: 10 행복도: 55 청결도: 85 레벨: 4
경과날짜: 15일
현재시간: 아침
```

```
배고픔: 35 행복도: 61 청결도: 70 레벨: 4
경과날짜: 19일
현재시간: 저녁
1. 행동하기, 2. 쇼핑하기 (아침, 점심에만 가능), 3.게임 설명, 4. 게임 종료
무엇을 하실 건가요?: 1
어떤 행동을 하시겠습니까? (1: 먹이 주기, 2: 목욕시키기. 3: 놀아주기, 4: 탐색 하기)
3
당신은 제니를 놀아주었습니다.
제니가 최종 성장을 하였습니다! 축하합니다!

종료 코드 0(으)로 완료된 프로세스
```

게임을 진행하면 할 수록 경과날짜가 늘어나는데 경과날짜가 5, 10, 15, 20이 되면 성장을 했다는 메시지와 함께 성장 레벨이 올라간다. 만약 경과날짜가 20이 되어 레벨 5가 되면 축하 메세지와 함께 게임이 종료된다.