MiniProject

December 7, 2021

1 Miniproject: Pet Program

2 Level 8

2.1 Samiha Kamal

2.2 18/11/2021

2.3 Version 5

2.4 Summary of the Question

The aim of this mini project is too create a program that simulates a pet and lets the user take care of it. There will be features available on this project such as a hunger and feeding system as well as a shop that will allow users to buy toys for their chosen pet.

2.5 The literate program development

2.5.1 Introduction to the petshop

What it does Welcomes the user to the petshop and asks for their name. This will be how the program refers to the user later in the future.

Implementation (how it works) Will initiate a new scanner variable and create a new string called name, the program will then ask the user for their name which will be stored into the variable of type string using the scanner.

[1]:
```
//Method for introduction and asking for name
//
public static String welcome()
{
    Scanner scanner = new Scanner(System.in); //initialising new scanner␣
 ↪variable

    System.out.println("Welcome to the petshop!");
    System.out.println("Please enter your name:"); //asking for user input
    String name = scanner.nextLine(); // input stored into variable name
    return name;
}
```

Testing

[2]:
```
//Testing welcome method
//
welcome();
```

```
Welcome to the petshop!
Please enter your name:

 Samiha
```

[2]:
```
Samiha
```

2.5.2 Picking your pet

What it does The program will give the user a selection of pets to choose from and after making their choice the program will ask the user to name their new pet.

Implementation (how it works) Using an if statement I will give the user a choice between three pets cat, rabbit and dog, the user should then input what pet that they would like. The if statement will check what the user has chosen and then ask them to name their pet.

[2]:
```
// Method for picking you pet
//
public static String pet_picking()
{
    //Variables needed
    String name = "";
    Scanner scanner = new Scanner(System.in); //initialising a new scanner␣
↪variable

    //Printing out which animal they want
    System.out.println("Please choose your pet from the selction below: ");
    System.out.println("Do you want A. Cat,  B. Dog or C. Rabbit?");
    System.out.println("Please enter the letter of the animal you wish to have␣
↪(In capitals please)");
    String pet = scanner.nextLine(); //storing the user input into a string

    //If statements to give the correct output for whatever animal they have␣
↪chosen
    if (pet.equals("A")){
        System.out.println("Thank you for choosing the cat! Please name your␣
↪cat: ");
        name = scanner.nextLine();
    }else if (pet.equals("B")){
        System.out.println("Thank you for choosing the dog! Please name your␣
↪dog: ");
        name = scanner.nextLine();
```

```
    }else if (pet.equals("C")){
        System.out.println("Thank you for choosing the rabbit! Please name your␣
↪rabbit: ");
        name = scanner.nextLine();
    }else{
        System.out.println("Invalid input"); //In case user hasnt written a␣
↪valid input message will tell them.
    }
    return name;
}
```

Testing

[5]:
```
//Testing the pet picking method
//
pet_picking();
```

```
Please choose your pet from the selction below:
Do you want A. Cat,  B. Dog or C. Rabbit?
Please enter the letter of the animal you wish to have (In capitals please)

 A

Thank you for choosing the cat! Please name your cat:

 CAT
```

[6]:
```
//Invalid testing the pet picking method
//
pet_picking();
```

```
Please choose your pet from the selction below:
Do you want A. Cat,  B. Dog or C. Rabbit?
Please enter the letter of the animal you wish to have (In capitals please)

 S

Invalid input
```

2.5.3 Introduction method

What it does Calls both methods so that the program does not have to call them one by one on the main method.

Implementation (how it works) Program will call the welcome method first and then the pet picking method after.

[3]:
```
//Methods for both calls
//
public static void introduction()
```

```
{
    welcome();
    pet_picking();
}
```

Testing

[8]:
```
//Testing the introduction method
//
introduction();
```

```
Welcome to the petshop!
Please enter your name:
```

```
 Samiha
```

```
Please choose your pet from the selction below:
Do you want A. Cat,  B. Dog or C. Rabbit?
Please enter the letter of the animal you wish to have (In capitals please)
```

```
 B
```

```
Thank you for choosing the dog! Please name your dog:
```

```
 DOG
```

2.6 The Hunger System

2.6.1 Input String method

What it does Takes a string and then returns a value.

Implementation (how it works) When called a String is passed through to the method, the method then uses a Scanner variable to ask for user input and then stored into a variable. This variable is then returned.

[3]:
```
// Input String method
//
public static String inputString(String message)
{
    Scanner scanner = new Scanner(System.in); //initiate a scanner variable
    System.out.println(message); //print out the String passed
    String userInput = scanner.nextLine(); //Store the user input

    return userInput; //return the user input
}
```

Testing

[15]:
```
// TEST CODE for inputString
//
```

```
String inputStringTest = inputString("Write the letter a down");
```

```
Write the letter a down

 a
```

2.6.2 Random number generator

What it does This will randomly generate a value between 0 and 5, this will be used to see whether the pet is hungry where 5 is bloated and 0 is extremely hungry.

Implementation (how it works) I will initialise a new Random variable which produces random variables. This integer will then be printed out.

[4]:
```
//Code for random number generator
//
Random rand = new Random();
int n = rand.nextInt(6);

System.out.println(n);
```

```
4
```

The code above is a random number generator, so now I will create a new method whose purpose is to create random numbers and then call the new method using an integer variable.

[5]:
```
//Code for hunger method
//
public static int hunger()
{
    // creates a random number
    Random rand = new Random();
    int n = rand.nextInt(6);

    return n;
}
```

Testing

[6]:
```
//Testing the hunger method
//
int a  = hunger();
System.out.println(a);
```

```
1
```

2.6.3 Asking to check hunger method

What it does What this method will do is ask the user whether they want to check their pets hunger levels or not.

Implementation (how it works) I will ask the user if they want to check their pets hunger and depending on their answer will either call another method that will output what the hunger is or will just stop the program.

[7]:
```
//Method for hunger checking
//
public static void checkHunger(int [] a)
{
    //asking for user input whether they want to check their pets hunger or not
    String ans = inputString("Do you want to check your pet's hunger? (y/n) ");

    //if statement to see what occurs when the user makes their choice
    if (ans.equals("y"))
    {
        //method call here
        hungerOutput(a);
    }
    else if (ans.equals("n"))
    {
        System.out.println("Ok then"); //prints out ok message
    }
    else
    {
        System.out.println("I dont understand that input"); // if input is not␣
 ↪expected prints out message
    }
}
```

2.6.4 Outputting the hunger level

What it does The program will output the pets hunger level and then say what that number mean, for example a level 5 hunger would mean that the pet is full and cannot eat any more.

Implementation (how it works) What the program will do is call the hunger method and then then output the value. I have also used a switch case statement to output how severe or normal the pets hunger level is and then asks if the user wants to feed their pet. This will call another method to feed the users pet.

[8]:
```
//method to ouput hunger
//
public static void  hungerOutput(int [] b)
{
    int hungerRand = hunger(); // call the hunger method
    int a = hungerRand; // a is the hunger level
    sortArray(storeArray(a, b));

    System.out.println("Your pet's hunger is at level " + hungerRand); //prints␣
 ↪out the hunger variable
```

```
    //checks what the hunger level is and then the corresponding messages shows up based on each case
    switch (hungerRand){
        case 0: case 1:
            System.out.println("Hunger level is low! Please feed your pet!");
            break;
        case 2: case 3:
            System.out.println("Hunger level is moderate! Please feed your pet");
            break;
        case 4:
            System.out.println("Hunger level is full! No need to feed your pet");
            break;
        case 5:
            System.out.println("Hunger level is bloated! Stop feeding your pet!");
            break;
    }

    //asks to feed pet
    String ans = inputString("Do you want to feed your pet? (y/n) ");
    if (ans.equals("y"))
    {
        //method call to feed pet here
        a = feed(hungerRand); // new value of the hunger is calculated after feeding.

    }
    else if (ans.equals("n"))
    {
        System.out.println("Fair enough");
    }
    else
    {
        System.out.println("Sorry I do not understand that input"); //catch back if invalid input
    }

    //return hunger level
    return;
}
```

Testing

[22]:
```
//Testing hungerOutput method
//
hungerOutput();
```

```
Your pet's hunger is at level 3
Hunger level is moderate! Please feed your pet
Do you want to feed your pet? (y/n)

 n

Fair enough
```

2.6.5 Feed the pet

What it does It asks the user if they want to feed their pet and if they do then the program gives them two options between premium feed and normal feed that helps alleviate the hunger of the pet by different amounts.

Implementation (how it works) What the program will do is call the hunger method and then then output the value. I have also used a switch case statement to output how severe or normal the pets hunger level is and then asks if the user wants to feed their pet. This will call another method to feed the users pet.

[9]:
```
//Method to feed your pets
//
public static int feed(int n)
{
    int hungerLevel = 0; //initialises the hunger level

    //which feed that they want to use
    System.out.println("Do you want to use A.premium feed (Hunger level + 4) or␣
↪B.normal feed (Hunger level + 2)?");
    String feed = inputString("Please input your answer in capitals, thank␣
↪you"); //stores their answer

    //checks what they picked and then does the correct calcualtions
    if (feed.equals("A"))
    {
        hungerLevel = n + 4;
        //hunger level is a max of 5, if the level is greater than  then it is␣
↪capped at 5
        if (hungerLevel > 5)
        {
            hungerLevel = 5;
        }
    }
    else if (feed.equals("B"))
    {
```

```
        hungerLevel = n + 2;
        if (hungerLevel > 5)
        {
            hungerLevel = 5;
        }
    }
    else
    {
        System.out.println("Sorry we dont have that feed"); //catch
    }

    //prints out hunger level
    System.out.println("Your pets hunger level is now at level " + hungerLevel);

    return hungerLevel;
}
```

Testing

[28]:
```
//Testing feed method
//
hungerOutput();
```

```
Your pet's hunger is at level 1
Hunger level is low! Please feed your pet!
Do you want to feed your pet? (y/n)

 y

Do you want to use A.premium feed (Hunger level + 4) or B.normal feed (Hunger
level + 2)?
Please input your answer in capitals, thank you

 A

Your pets hunger level is now at level 5
```

2.6.6 Sorting the hunger levels

What it does This method will collect the values of the hunger method and then sort them.

Implementation (how it works) I will create a new array value and then store the values of the hunger which will then be sorted.

[10]:
```
//Method to store array values
//
public static int [] storeArray(int a, int [] b)
{
    boolean space = false;
```

```
    for (int i = 0; i < b.length; i++) //loops through the length of the whole
 ↪array
    {
        if (b[i] == 0) //if there is no value stored then
        {
            b[i] = a; //new value stored into array
            space = true;
            return b;
        }
        else
        {
            space = false; //no space
        }
    }

    return b;
}

//Method to sort array
//
public static void sortArray(int [] a)
{
    int temp = 0; //creates a temporary storage

    for (int j = 0; j < a.length; j++)
    {
        for (int i = 0; i < a.length-1; i++)
        {
            if (a[i] > a[i+1]) //uses bubble sort to sort elements in list
            {
                temp = a[i];
                a[i] = a[i+1];
                a[i+1] = temp;
            }
        }
    }
}
```

2.6.7 Petting your animals

What it does This method asks the user if they want to pet their animal, the program will then give the users two options of using a brush or using their hand.

Implementation (how it works) I will use a while loop and ask if the user wants to pet their pet. This will be looped indefinitely until the user exits from the loop by typing XXXX.

[10]:
```
String stopPetting = ""; //initialising stopPetting to enter loop
int [] numberOfPets = new int[2]; //creating a new array to keep track of␣
 ↪numbers
int brushCount = 0, handCount = 0; //initialising counts and setting them to 0

while (!(stopPetting.equals("XXXX"))) //checks if user wants to end the program␣
 ↪or not before starting loop
{
    //asks if they want to choose between the brush or their hand
    String toPet = inputString("Do you want to pet you pet with A.Brush or B.
 ↪Hand ");
    if (toPet.equals("A"))
    {
        //uses an array to store the total amount of pets thats were done by␣
 ↪each option
        numberOfPets[0] = (brushCount = brushCount + 1);
        System.out.println("Your pet enjoyed the brushes!");
    }
    else if (toPet.equals("B"))
    {
        numberOfPets[1] = (handCount = handCount + 1);
        System.out.println("Your pet enjoyed the pets!");
    }
    else
    {
        System.out.println("Sorry invalid input"); //catch for if the user␣
 ↪enters invalid input
    }
    //user confirmation incase they want to stop petting
    stopPetting = inputString("Do you want to stop petting now? (XXXX to stop,␣
 ↪or enter any key to continue)");
}
```

```
Do you want to pet you pet with A.Brush or B.Hand
 a
Sorry invalid input
Do you want to stop petting now? (XXXX to stop, or enter any key to continue)
 XXXX
```

2.6.8 Calculating how many pats have been done

What it does This method calculates the total amount of pets done and then outputs the result

Implementation (how it works) I will create a method that needs the brushCount, handCount and the array to be passed through before it can output the correct information to the user.

[11]:
```
//method to output how many pets were done
public static void calculatePats(int [] n)
{
    //outputs information from the array
    System.out.println("You brushed your pet " + n[0] + " times.");
    System.out.println("You hand petted your pet " + n[1] + " times.");
}
```

2.6.9 Introduction method

What it does The introduction method will now contain the hunger system and also give the user the chance to pet their animals.

Implementation (how it works) I will loop both the hunger and petting systems so that if the user wants to feed their pet again they are able to, and that they will remain within the pet program until they type XXXX in which they can finally exit the program as they wish.

[12]:
```
//Introduction method
//
public static void introduction()
{
    String userInput = ""; //creating string variable to enter loop
    String stopPetting = ""; //initialising stopPetting to enter loop
    int [] numberOfPets = new int[2]; //creating a new array to keep track of␣
↪numbers
    int brushCount = 0, handCount = 0; //initialising counts and setting them␣
↪to 0
    int [] hungerArray = new int[100]; //creating a new array for storing␣
↪hunger variables

    welcome();
    pet_picking();

    while (!(userInput.equals("XXXX"))) //checks if while loop should carry on
    {
        checkHunger(hungerArray); //goes into the hunger system
        while (!(stopPetting.equals("XXXX"))) //checks if user wants to end the␣
↪program or not before starting loop
        {
            //asks if they want to choose between the brush or their hand
            String toPet = inputString("Do you want to pet you pet with A.Brush␣
↪or B.Hand ");
            if (toPet.equals("A"))
            {
                //uses an array to store the total amount of pets thats were␣
↪done by each option
                numberOfPets[0] = (brushCount = brushCount + 1);
```

```
                System.out.println("Your pet enjoyed the brushes!");
            }
            else if (toPet.equals("B"))
            {
                numberOfPets[1] = (handCount = handCount + 1);
                System.out.println("Your pet enjoyed the pets!");
            }
            else
            {
                System.out.println("Sorry invalid input"); //catch for if the␣
↪user enters invalid input
            }
            //user confirmation incase they want to stop petting
            stopPetting = inputString("Do you want to stop petting now? (XXXX␣
↪to stop, or enter any key to continue)");
        }
        //output how many pets have occurred
        calculatePats(numberOfPets);
        //ask the user if they want to exit the pet program
        userInput = inputString("Do you want to exit the program? (Type XXXX␣
↪for yes, else press any key to continue) ");
    }
    return;
}
```

Testing

[37]:
```
//Testing introduction method
//
introduction();
```

```
Welcome to the petshop!
Please enter your name:

 Samiha

Please choose your pet from the selction below:
Do you want A. Cat,  B. Dog or C. Rabbit?
Please enter the letter of the animal you wish to have (In capitals please)

 A

Thank you for choosing the cat! Please name your cat:

 ACAT

Do you want to check your pet's hunger? (y/n)

 y

Your pet's hunger is at level 4
```

```
Hunger level is full! No need to feed your pet
Do you want to feed your pet? (y/n)

 n

Fair enough
Do you want to pet you pet with A.Brush or B.Hand

 A

Your pet enjoyed the brushes!
Do you want to stop petting now? (XXXX to stop, or enter any key to continue)

 a

Do you want to pet you pet with A.Brush or B.Hand

 B

Your pet enjoyed the pets!
Do you want to stop petting now? (XXXX to stop, or enter any key to continue)

 XXXX

You brushed your pet 1 times.
You hand petted your pet 1 times.
Do you want to exit the program? (Type XXXX for yes, else press any key to
continue)

 as

Do you want to check your pet's hunger? (y/n)

 y

Your pet's hunger is at level 4
Hunger level is full! No need to feed your pet
Do you want to feed your pet? (y/n)

 n

Fair enough
You brushed your pet 1 times.
You hand petted your pet 1 times.
Do you want to exit the program? (Type XXXX for yes, else press any key to
continue)

 XXXX
```

2.7 The shop system

2.7.1 Creating the ADT

What it does I will create a new record type called class and also the getter and setter methods to access data within that record.

Implementation (how it works) I will create a new class called toys with variables that I believe is necessary for toys and then create methods to get data from within the record and set data. I will also have a method to create a new instance of the record toys. The abstract data type will be completed when everything can be accessed through methods.

[13]:
```
//Creating the toys class
//
class toys
{
    String name; //name of toy
    int happinessLevel; //how much happiness a pet will have after playing
    double cost; //cost of toy
}
```

Setter methods

[14]:
```
//Methods to set data in the record
//
public static toys setToyName(toys ty, String nm) //sets the name of the toy
{
    ty.name = nm;
    return ty;
}

public static toys setToyHappinessLevel(toys ty, int hl) //sets the happiness␣
 ↪level of the toy
{
    ty.happinessLevel = hl;
    return ty;
}

public static toys setToyCost(toys ty, double cst) //sets the cost of the toy
{
    ty.cost = cst;
    return ty;
}
```

Getter methods

[15]:
```
//Methods to return data to whereever it was called
//
public static String getToyName(toys ty) //return the toys name
{
    return ty.name;
}

public static int getToyHappinessLevel(toys ty) //return the toys happiness␣
 ↪level
```

```
{
    return ty.happinessLevel;
}

public static double getToyCost(toys ty) //return the cost of the toy
{
    return ty.cost;
}
```

ToString method The toString method is used to display whatever is stored into the record. It takes in a toy and then outputs the corresponding information pertaining to it.

[16]:
```
//Method to ouptput toys record
//
public static String toStringToys(toys ty) //takes in a toy
{
    String a = "The " + getToyName(ty) + " increases happiness level by " +␣
 ↪getToyHappinessLevel(ty)
                + ", it costs £" + getToyCost(ty) + "0"; //outputs the details␣
 ↪of the record into a string
    return a; //returns the string.
}
```

Create new toy method This method creates a new instance of the class toy. This is so that instead of writing multiple lines of code for various toys, I only need to write one for each toy since they will use the same create toy method to be created.

[17]:
```
//Method to create toys
//
public static toys createToys(String nm, int hl, double cst) //pass on values
{
    toys newToys = new toys(); //new instance of a toy
    newToys = setToyName(newToys,nm); //name of the toy is set
    newToys = setToyHappinessLevel(newToys,hl); //happiness level of the toy is␣
 ↪set
    newToys = setToyCost(newToys,cst); //cost of the toy is set
    return newToys;
}
```

Method to store and display data I will have one method that stores the data of the record and also outputs them. The reason that they are both in the same method is because whenever I try to call the records from a separate method I would receive error messages, as a result I have decided to do everything within the same method. This method will also take in a String value, this is because I am planning on having a main shop interface that asks the user to choose between three toys. The user will make their selection by typing the letter that represents the toy they want and after that I will display the information of the selected toy. Therefore to make sure that

I have displayed the correct information of the toy the program will pass on the letter the user has entered and use if statements to check what data it should output.

[18]:
```
//Method to store and display data
//
public static void storeAndDisplayToyInformation(String letter) //takes in␣
↪value
{
    String nm = "";
    //creates new toys
    toys catTower = createToys("Cat Tower", 5, 50.0);
    toys dogToy = createToys("Doggy Chew Toy", 4, 3.0);
    toys rabbitTube = createToys("Willow Tunnel", 2, 6.0);

    switch (letter){ //checks what the user input is and displays correct␣
↪information
        case "A":
            nm = getToyName(catTower); //stores the name in a variable which␣
↪will be used later on
            System.out.println(toStringToys(catTower)); //prints out the␣
↪message
            break;
        case "B":
            nm = getToyName(dogToy);
            System.out.println(toStringToys(dogToy));
            break;
        case "C":
            nm = getToyName(rabbitTube);
            System.out.println(toStringToys(rabbitTube));
            break;
        default: //if none of the above are entered then a message will be␣
↪outputted
            System.out.println("Sorry I dont understand that input.");
            return; //the method will end here if the user writes the wrong␣
↪input
    }
    System.out.println(); //Space between the information and the dialogue to␣
↪purchase item
    //will have a method call here to let users purchase the toys
    purchaseOutput(toPurchase(nm));
    return;
}
```

Testing

[31]:
```
//Testing the store and display toy information method
//
```

```
storeAndDisplayToyInformation("A");
```

```
The Cat Tower increases happiness level by 5, it costs £50.00

Do you want to purchase the Cat Tower? (y/n)

 n

If you do not want this item, there are plenty more in store.
```

[31]:
```
//Invalid testing of the same method
//
storeAndDisplayToyInformation("Random");
```

```
Sorry I dont understand that input.
```

Purchasing methods In order to purchase items I have created two method, one takes a string value that is the name of the toy that is currently being browsed and asks the user if they want to buy the current toy that they are looking at, depending on the answer the method returns a boolean value. This boolean value is then passed to another method called purchase output, this method is specifically created for outputting the congratulations message if they have a purchased a new toy or it says that there are plenty more toys to browse if they decide against it.

[19]:
```
//Method for the boolean toy check
//
public static boolean toPurchase(String toyName) //takes the toys name
{
    String ans = inputString("Do you want to purchase the " + toyName + "? (y/
 ↪n)"); //asks for a user output
    boolean b = (ans.equals("y")); //depending on the users answer it returns a␣
 ↪boolean value of either true or false
    return b;
}

//Method to output purchase messages
//
public static void purchaseOutput(Boolean a) //takes in the boolean value from␣
 ↪toPurchase
{
    if (a) //if its true
    {
        System.out.println("Congratulations for purchasing an item"); //
 ↪congratulatory message
    }
    else //if its false
    {
        System.out.println("If you do not want this item, there are plenty more␣
 ↪in store.");
```

```
    }
}
```

Testing

[35]:
```
//Testing the purchase methods
//
purchaseOutput(toPurchase("someToyName"));
```

```
Do you want to purchase the someToyName? (y/n)

 y

Congratulations for purchasing an item
```

[36]:
```
//Invalid testing the purchase methods
//
purchaseOutput(toPurchase("anotherToyName"));
```

```
Do you want to purchase the anotherToyName? (y/n)

 n

If you do not want this item, there are plenty more in store.
```

This method call will be placed within the storeAndDisplayToyInformation() method. This is so that right after the user looks through the toys information they are given the option to purchase the item.

2.7.2 Shop interface

What it does Creates a shop system where user can buy toys for their pet.

Implementation (how it works) I will add a a question on whether user wants to visit the shop, if they say yes then the shop will open with toys that have information stored in a record, user can view information on each toy. The method is what the user will see once they have said yes to going to the store. It will loop the store so that the user can browse the store for as long as they like and will only exit it the interface when the user has entered XXXX.

[20]:
```
//Method for the main body of the shop system
//
public static void shopInterface()
{
    String closeShop = ""; //variable to enter loop
    while (!(closeShop.equals("XXXX"))) //checks userinput to see if the store␣
 ↪interface should close.
    {
        System.out.println("Hello! Welcome to the shop! This is our current␣
 ↪stock available: ");
```

```
        System.out.println("A.Cat Tower, B. Doggy Chew Toy, C. Willow Rabbit␣
↪Tunnel."); //choose the toy you want
        String toyInput = inputString("Type a letter to view the toys␣
↪information (in capitals please): ");
        storeAndDisplayToyInformation(toyInput); //method that will display the␣
↪toys information
        //asks if user wants to close store
        closeShop = inputString("To close the shop XXXX, else type anything␣
↪else to continue browsing.");
    }
    return;
}
```

Testing

[38]:
```
//Testing the shop interface method
//
shopInterface();
```

```
Hello! Welcome to the shop! This is our current stock available:
A.Cat Tower, B. Doggy Chew Toy, C. Willow Rabbit Tunnel.
Type a letter to view the toys information (in capitals please):

 A

The Cat Tower increases happiness level by 5, it costs £50.00

Do you want to purchase the Cat Tower? (y/n)

 y

Congratulations for purchasing an item
To close the shop XXXX, else type anything else to continue browsing.

 as

Hello! Welcome to the shop! This is our current stock available:
A.Cat Tower, B. Doggy Chew Toy, C. Willow Rabbit Tunnel.
Type a letter to view the toys information (in capitals please):

 B

The Doggy Chew Toy increases happiness level by 4, it costs £3.00

Do you want to purchase the Doggy Chew Toy? (y/n)

 n

If you do not want this item, there are plenty more in store.
To close the shop XXXX, else type anything else to continue browsing.

 XXXX
```

2.7.3 Asking to enter shop method

What it does What this method will do is ask the user whether they want to enter the store or not

Implementation (how it works) I will ask the user if they want to enter the toy store and depending on their answer will either call the shopInterface method. This is done so that it only takes one line of code within the introduction method to implement the whole toy shop.

[21]:
```
//method to enter store
//
public static void enterShop()
{
    String enterShop = inputString("Do you want to enter the store? (y/n)");
    if (enterShop.equals("y")) //checks user input
    {
        shopInterface(); //opens the shop interface
    }
    else
    {
        System.out.println("Okay then");
    }
    return;
}
```

Testing

[40]:
```
//Testing the enterShop method
//
enterShop();
```

```
Do you want to enter the store? (y/n)

 y

Hello! Welcome to the shop! This is our current stock available:
A.Cat Tower, B. Doggy Chew Toy, C. Willow Rabbit Tunnel.
Type a letter to view the toys information (in capitals please):

 A

The Cat Tower increases happiness level by 5, it costs £50.00


Do you want to purchase the Cat Tower? (y/n)

 N

If you do not want this item, there are plenty more in store.
To close the shop XXXX, else type anything else to continue browsing.

 as
```

```
Hello! Welcome to the shop! This is our current stock available:
A.Cat Tower, B. Doggy Chew Toy, C. Willow Rabbit Tunnel.
Type a letter to view the toys information (in capitals please):

 S

Sorry I dont understand that input.
To close the shop XXXX, else type anything else to continue browsing.

 XXXX
```

2.7.4 Introduction method

What it does The introduction method will now contain the shopping system

Implementation (how it works) I will add the shop with the hunger system and the petting code so that the users can enter and leave the store for as many times as they want before they leave the full program.

[22]:
```
//method for the updated introduction.
//
//Introduction method
//
public static void introduction()
{
    String userInput = ""; //creating string variable to enter loop
    String stopPetting = ""; //initialising stopPetting to enter loop
    int [] numberOfPets = new int[2]; //creating a new array to keep track of␣
↪numbers
    int brushCount = 0, handCount = 0; //initialising counts and setting them␣
↪to 0
    int [] hungerArray = new int[100]; //creating a new array for storing␣
↪hunger variables

    welcome();
    pet_picking();

    while (!(userInput.equals("XXXX"))) //checks if while loop should carry on
    {
        checkHunger(hungerArray); //goes into the hunger system
        System.out.println(); //Space between the hunger system and shop system
        enterShop();
        while (!(stopPetting.equals("XXXX"))) //checks if user wants to end the␣
↪program or not before starting loop
        {
            //asks if they want to choose between the brush or their hand
            String toPet = inputString("Do you want to pet you pet with A.Brush␣
↪or B.Hand ");
            if (toPet.equals("A"))
```

```
            {
                //uses an array to store the total amount of pets thats were␣
↪done by each option
                numberOfPets[0] = (brushCount = brushCount + 1);
                System.out.println("Your pet enjoyed the brushes!");
            }
            else if (toPet.equals("B"))
            {
                numberOfPets[1] = (handCount = handCount + 1);
                System.out.println("Your pet enjoyed the pets!");
            }
            else
            {
                System.out.println("Sorry invalid input"); //catch for if the␣
↪user enters invalid input
            }
            //user confirmation incase they want to stop petting
            stopPetting = inputString("Do you want to stop petting now? (XXXX␣
↪to stop, or enter any key to continue)");
        }
        //output how many pets have occurred
        calculatePats(numberOfPets);
        //ask the user if they want to exit the pet program
        userInput = inputString("Do you want to exit the program? (Type XXXX␣
↪for yes, else press any key to continue) ");
    }
    return;
}
```

Testing

[46]:
```
//Testing the introduction method
//
introduction();
```

```
Welcome to the petshop!
Please enter your name:

 Samiha

Please choose your pet from the selction below:
Do you want A. Cat,  B. Dog or C. Rabbit?
Please enter the letter of the animal you wish to have (In capitals please)

 A

Thank you for choosing the cat! Please name your cat:

 CAT
```

```
Do you want to check your pet's hunger? (y/n)

 y

Your pet's hunger is at level 0
Hunger level is low! Please feed your pet!
Do you want to feed your pet? (y/n)

 y

Do you want to use A.premium feed (Hunger level + 4) or B.normal feed (Hunger
level + 2)?
Please input your answer in capitals, thank you

 A

Your pets hunger level is now at level 4


Do you want to enter the store? (y/n)

 y

Hello! Welcome to the shop! This is our current stock available:
A.Cat Tower, B. Doggy Chew Toy, C. Willow Rabbit Tunnel.
Type a letter to view the toys information (in capitals please):

 A

The Cat Tower increases happiness level by 5, it costs £50.00


Do you want to purchase the Cat Tower? (y/n)

 y

Congratulations for purchasing an item
To close the shop XXXX, else type anything else to continue browsing.

 aa

Hello! Welcome to the shop! This is our current stock available:
A.Cat Tower, B. Doggy Chew Toy, C. Willow Rabbit Tunnel.
Type a letter to view the toys information (in capitals please):

 B

The Doggy Chew Toy increases happiness level by 4, it costs £3.00


Do you want to purchase the Doggy Chew Toy? (y/n)

 n

If you do not want this item, there are plenty more in store.
To close the shop XXXX, else type anything else to continue browsing.

 XXXX

Do you want to pet you pet with A.Brush or B.Hand
```

```
 A

Your pet enjoyed the brushes!
Do you want to stop petting now? (XXXX to stop, or enter any key to continue)

 DSDSD

Do you want to pet you pet with A.Brush or B.Hand

 A

Your pet enjoyed the brushes!
Do you want to stop petting now? (XXXX to stop, or enter any key to continue)

 SDSD

Do you want to pet you pet with A.Brush or B.Hand

 B

Your pet enjoyed the pets!
Do you want to stop petting now? (XXXX to stop, or enter any key to continue)

 XXXX

You brushed your pet 2 times.
You hand petted your pet 1 times.
Do you want to exit the program? (Type XXXX for yes, else press any key to
continue)

 as

Do you want to check your pet's hunger? (y/n)

 n

Ok then


Do you want to enter the store? (y/n)

 n

Okay then
You brushed your pet 2 times.
You hand petted your pet 1 times.
Do you want to exit the program? (Type XXXX for yes, else press any key to
continue)

 XXXX
```

2.8 File input/output

2.8.1 File input

What it does Will store the users name and the pets name into a file

Implementation (how it works) I will use the file input method to store a data into a new file.

[23]:
```
//Method for file input
//
public static void storeData(String a, String b) throws IOException
{
    PrintWriter saving = new PrintWriter(new FileWriter("petprogram.txt")); //
 ↪new PrintWriter variable

    saving.println(a); //Write a on file
    saving.println(b); //Write b on file

    saving.close(); //Close file writer
}
```

Testing

[63]:
```
//Testing store Data method
//
storeData("name", "petsname");
```

The code above created a new text file that stored name and petname on different lines each

2.8.2 File Output

What it does Will get the users data that is stored in a separate file and store it into variables

Implementation (how it works) I will use the file output method to store a data into a some variables to use for later.

[24]:
```
//Method for file output
//
public static String retrieveData() throws IOException //File output for␣
 ↪leaving game
{
    BufferedReader retrieving = new BufferedReader(new FileReader("petprogram.
 ↪txt")); //retrieves data from text file
    String userName = retrieving.readLine();
    String petName = retrieving.readLine();

    String sentence = "Goodbye " + userName + "! " + petName + " will miss you!
 ↪"; //outputs string
    retrieving.close();
    return sentence;
}
```

[31]:
```
//Method for file output
//
public static String continueData() throws IOException //file output for␣
 ↪continuing game
{
    BufferedReader retrieving = new BufferedReader(new FileReader("petprogram.
 ↪txt")); //retrieves data from text file
    String userName = retrieving.readLine();
    String petName = retrieving.readLine();

    String sentence = "Welcome back " + userName + "! " + petName + " is happy␣
 ↪to see you again!"; //outputs string
    retrieving.close();
    return sentence;
}
```

Testing

[86]:
```
//Testing retrieve data method
retrieveData();
```

[86]:
```
Goodbye name! petsname will miss you!
```

[25]:
```
public static void test() throws IOException
{
    try {System.out.println(retrieveData());}
    catch (IOException e){
    }
}
```

[85]:
```
test();
```

```
Goodbye name! petsname will miss you!
```

2.8.3 Introduction method

What it does The introduction method will now contain the file input output methods

Implementation (how it works) I will add the shop with the file input outputs to save their data.

[39]:
```
//Method for the updated introduction.
//
//Introduction method
//
public static void introduction(String a)
{
    String userInput = ""; //creating string variable to enter loop
```

```
   String stopPetting = ""; //initialising stopPetting to enter loop
   int [] numberOfPets = new int[2]; //creating a new array to keep track of␣
↪numbers
   int brushCount = 0, handCount = 0; //initialising counts and setting them␣
↪to 0
   int [] hungerArray = new int[100]; //creating a new array for storing␣
↪hunger variables

   if (a.equals("yes")) //If they want to start a new game then:
   {
       String name = welcome();
       String petName = pet_picking();
       try {storeData(name,petName);}
       catch (IOException e){
       }//stores data in files
   }

   while (!(userInput.equals("XXXX"))) //checks if while loop should carry on
   {
       checkHunger(hungerArray); //goes into the hunger system
       System.out.println(); //Space between the hunger system and shop system
       enterShop();
       while (!(stopPetting.equals("XXXX"))) //checks if user wants to end the␣
↪program or not before starting loop
       {

           //asks if they want to choose between the brush or their hand
           String toPet = inputString("Do you want to pet you pet with A.Brush␣
↪or B.Hand ");
           if (toPet.equals("A"))
           {
               //uses an array to store the total amount of pets thats were␣
↪done by each option
               numberOfPets[0] = (brushCount = brushCount + 1);
               System.out.println("Your pet enjoyed the brushes!");
           }
           else if (toPet.equals("B"))
           {
               numberOfPets[1] = (handCount = handCount + 1);
               System.out.println("Your pet enjoyed the pets!");
           }
           else
           {
               System.out.println("Sorry invalid input"); //catch for if the␣
↪user enters invalid input
           }
```

```
            //user confirmation incase they want to stop petting
            stopPetting = inputString("Do you want to stop petting now? (XXXX␣
↪to stop, or enter any key to continue)");
        }
        //output how many pets have occurred
        calculatePats(numberOfPets);
        //ask the user if they want to exit the pet program
        userInput = inputString("Do you want to exit the program? (Type XXXX␣
↪for yes, else press any key to continue) ");
    }
    try {System.out.println(retrieveData());} //prints out message taken from␣
↪file
    catch (IOException e){
    }
    return;
}
```

Testing

[28]:
```
//Testing introduction method
//
introduction();
```

```
Welcome to the petshop!
Please enter your name:

 Sam

Please choose your pet from the selction below:
Do you want A. Cat,  B. Dog or C. Rabbit?
Please enter the letter of the animal you wish to have (In capitals please)

 C

Thank you for choosing the rabbit! Please name your rabbit:

 RAB

Do you want to check your pet's hunger? (y/n)

 n

Ok then


Do you want to enter the store? (y/n)

 n

Okay then
Do you want to pet you pet with A.Brush or B.Hand

 B
```

```
Your pet enjoyed the pets!
Do you want to stop petting now? (XXXX to stop, or enter any key to continue)

 XXXX

You brushed your pet 0 times.
You hand petted your pet 1 times.
Do you want to exit the program? (Type XXXX for yes, else press any key to
continue)

 XXXX

Goodbye Sam! RAB will miss you!
```

[41]:
```
//Method for game start menu
//
public static void gameStart()
{
    String userInput = inputString("Do you want to continue your game?(y/n)"); /
 ↪/Asks if user wants to continue their game
    if (userInput.equals("y"))
    {
        try {System.out.println(continueData());} //prints out message taken␣
 ↪from file
        catch (IOException e){
        }
        introduction("no"); //Calls introduction method and passes a no

    }
    else
    {
        introduction("yes"); //Calls introduction method and passes yes
    }
    return;
}
```

Testing

[43]:
```
//Testing gameStart method
//
gameStart();
```

```
Do you want to continue your game?(y/n)

 y

Welcome back Samiha! acaca is happy to see you again!
Do you want to check your pet's hunger? (y/n)

 n
```

```
Ok then


Do you want to enter the store? (y/n)

 n

Okay then
Do you want to pet you pet with A.Brush or B.Hand

 A

Your pet enjoyed the brushes!
Do you want to stop petting now? (XXXX to stop, or enter any key to continue)

 XXXX

You brushed your pet 1 times.
You hand petted your pet 0 times.
Do you want to exit the program? (Type XXXX for yes, else press any key to
continue)

 XXXX

Goodbye Samiha! acaca will miss you!
```

2.8.4 Running the program

Run the following call to simulate running the complete program.

[44]:
```
gameStart();
```

```
Do you want to continue your game?(y/n)

 n

Welcome to the petshop!
Please enter your name:

 Samiahasd

Please choose your pet from the selction below:
Do you want A. Cat,  B. Dog or C. Rabbit?
Please enter the letter of the animal you wish to have (In capitals please)

 A

Thank you for choosing the cat! Please name your cat:

 AASDA

Do you want to check your pet's hunger? (y/n)

 y

Your pet's hunger is at level 3
Hunger level is moderate! Please feed your pet
Do you want to feed your pet? (y/n)
```

```
 y
Do you want to use A.premium feed (Hunger level + 4) or B.normal feed (Hunger
level + 2)?
Please input your answer in capitals, thank you
 A
Your pets hunger level is now at level 5

Do you want to enter the store? (y/n)
 y
Hello! Welcome to the shop! This is our current stock available:
A.Cat Tower, B. Doggy Chew Toy, C. Willow Rabbit Tunnel.
Type a letter to view the toys information (in capitals please):
 A
The Cat Tower increases happiness level by 5, it costs £50.00

Do you want to purchase the Cat Tower? (y/n)
 y
Congratulations for purchasing an item
To close the shop XXXX, else type anything else to continue browsing.
 XXXX
Do you want to pet you pet with A.Brush or B.Hand
 A
Your pet enjoyed the brushes!
Do you want to stop petting now? (XXXX to stop, or enter any key to continue)
 XXXX
You brushed your pet 1 times.
You hand petted your pet 0 times.
Do you want to exit the program? (Type XXXX for yes, else press any key to
continue)
 XXXX
Goodbye Samiahasd! AASDA will miss you!
```

2.9 The complete program

This version will only compile here. To run it copy it into a file called initials.java on your local computer and compile and run it there.

[35]:
```
// SAMIHA KAMAL
// 18/11/2021
```

```
// VERSION 3
// Pet program that lets the user pick their pet and name it

import java.util.*; // Needed to make everything available

//Creating the toys class
//
class toys
{
    String name; //name of toy
    int happinessLevel; //how much happiness a pet will have after playing
    double cost; //cost of toy
}
class pets
{
    public static void main (String [] a)
    {
        gameStart();
        System.exit(0);
    }

    public static void gameStart()
    {
        String userInput = inputString("Do you want to continue your game?(y/
↪n)");
        if (userInput.equals("y"))
        {
            try {System.out.println(continueData());} //prints out message␣
↪taken from file
            catch (IOException e){
            }
            introduction("no");

        }
        else
        {
            introduction("yes");
        }
        return;
    }

   //Introduction method
    //
    public static void introduction(String a)
    {
        String userInput = ""; //creating string variable to enter loop
        String stopPetting = ""; //initialising stopPetting to enter loop
```

```
        int [] numberOfPets = new int[2]; //creating a new array to keep track␣
↪of numbers
        int brushCount = 0, handCount = 0; //initialising counts and setting␣
↪them to 0
        int [] hungerArray = new int[100]; //creating a new array for storing␣
↪hunger variables

        if (a.equals("yes"))
        {
            String name = welcome();
            String petName = pet_picking();
            try {storeData(name,petName);}
            catch (IOException e){
            }//stores data in files
        }

        while (!(userInput.equals("XXXX"))) //checks if while loop should carry␣
↪on
        {
            checkHunger(hungerArray); //goes into the hunger system
            System.out.println(); //Space between the hunger system and shop␣
↪system
            enterShop();
            while (!(stopPetting.equals("XXXX"))) //checks if user wants to end␣
↪the program or not before starting loop
            {

                //asks if they want to choose between the brush or their hand
                String toPet = inputString("Do you want to pet you pet with A.
↪Brush or B.Hand ");
                if (toPet.equals("A"))
                {
                    //uses an array to store the total amount of pets thats␣
↪were done by each option
                    numberOfPets[0] = (brushCount = brushCount + 1);
                    System.out.println("Your pet enjoyed the brushes!");
                }
                else if (toPet.equals("B"))
                {
                    numberOfPets[1] = (handCount = handCount + 1);
                    System.out.println("Your pet enjoyed the pets!");
                }
                else
                {
                    System.out.println("Sorry invalid input"); //catch for if␣
↪the user enters invalid input
```

```
                }
                //user confirmation incase they want to stop petting
                stopPetting = inputString("Do you want to stop petting now?␣
↪(XXXX to stop, or enter any key to continue)");
            }
            //output how many pets have occurred
            calculatePats(numberOfPets);
            //ask the user if they want to exit the pet program
            userInput = inputString("Do you want to exit the program? (Type␣
↪XXXX for yes, else press any key to continue) ");
        }
        try {System.out.println(retrieveData());} //prints out message taken␣
↪from file
        catch (IOException e){
        }
        return;
    }

    //Method for introduction and asking for name
    //
    public static void welcome()
    {
        Scanner scanner = new Scanner(System.in); //initialising new scanner␣
↪variable
        System.out.println("Welcome to the petshop!");
        System.out.println("Please enter your name:"); //asking for user input
        String name = scanner.nextLine(); // input stored into variable name
        return;
    }

    // Method for picking you pet
    //
    public static void pet_picking()
    {
        //Variables needed
        String name;
        Scanner scanner = new Scanner(System.in); //initialising a new scanner␣
↪variable

        //Printing out which animal they want
        System.out.println("Please choose your pet from the selction below: ");
        System.out.println("Do you want A. Cat,  B. Dog or C. Rabbit?");
        System.out.println("Please enter the letter of the animal you wish to␣
↪have (In capitals please)");
        String pet = scanner.nextLine(); //storing the user input into a string
```

```
        //If statements to give the correct output for whatever animal they␣
↪have chosen
        if (pet.equals("A")){
            System.out.println("Thank you for choosing the cat! Please name␣
↪your cat: ");
            name = scanner.nextLine();
        }else if (pet.equals("B")){
            System.out.println("Thank you for choosing the dog! Please name␣
↪your dog: ");
            name = scanner.nextLine();
        }else if (pet.equals("C")){
            System.out.println("Thank you for choosing the rabbit! Please name␣
↪your rabbit: ");
            name = scanner.nextLine();
        }else{
            System.out.println("Invalid input"); //In case user hasnt written a␣
↪valid input message will tell them.
        }
        return;
    }

    //Method for hunger checking
    //
    public static void checkHunger()
    {
        //asking for user input whether they want to check their pets hunger or␣
↪not
        String ans = inputString("Do you want to check your pet's hunger? (y/n)␣
↪");

        //if statement to see what occurs when the user makes their choice
        if (ans.equals("y"))
        {
            //method call here
            hungerOutput();
        }
        else if (ans.equals("n"))
        {
            System.out.println("Ok then"); //prints out ok message
        }
        else
        {
            System.out.println("I dont understand that input"); // if input is␣
↪not expected prints out message
        }
    }
```

```
   //Method for file input
   //
   public static void storeData(String a, String b) throws IOException
   {
       PrintWriter saving = new PrintWriter(new FileWriter("petprogram.txt"));␣
↪//new PrintWriter variable

       saving.println(a); //Write a on file
       saving.println(b); //Write b on file

       saving.close(); //Close file writer
   }

   //Method for file output
   //
   public static String retrieveData() throws IOException
   {
       BufferedReader retrieving = new BufferedReader(new␣
↪FileReader("petprogram.txt"));
       String userName = retrieving.readLine(); //stores data into variables
       String petName = retrieving.readLine();

       String sentence = "Goodbye " + userName + "! " + petName + " will miss␣
↪you!";
       retrieving.close(); //closing file reader

       return sentence;
   }

   //Method for file output
   //
   public static String continueData() throws IOException //file output for␣
↪continuing game
   {
       BufferedReader retrieving = new BufferedReader(new␣
↪FileReader("petprogram.txt")); //retrieves data from text file
       String userName = retrieving.readLine();
       String petName = retrieving.readLine();

       String sentence = "Welcome back " + userName + "! " + petName + " is␣
↪happy to see you again!"; //outputs string
       retrieving.close();
       return sentence;
   }
```

```
    //method to ouput hunger
    //
    public static void  hungerOutput()
    {
        int hungerRand = hunger(); // call the hunger method
        int a = hungerRand; // a is the hunger level
        System.out.println("Your pet's hunger is at level " + hungerRand); //
↪prints out the hunger variable

        //checks what the hunger level is and then the corresponding messages␣
↪shows up based on each case
        switch (hungerRand){
            case 0: case 1:
                System.out.println("Hunger level is low! Please feed your pet!
↪");
                break;
            case 2: case 3:
                System.out.println("Hunger level is moderate! Please feed your␣
↪pet");
                break;
            case 4:
                System.out.println("Hunger level is full! No need to feed your␣
↪pet");
                break;
            case 5:
                System.out.println("Hunger level is bloated! Stop feeding your␣
↪pet!");
                break;
        }
        //asks to feed pet
        String ans = inputString("Do you want to feed your pet? (y/n) ");
        if (ans.equals("y"))
        {
            //method call to feed pet here
            a = feed(hungerRand); // new value of the hunger is calculated␣
↪after feeding.
        }
        else if (ans.equals("n"))
        {
            System.out.println("Fair enough");
        }
        else
        {
            System.out.println("Sorry I do not understand that input"); //catch␣
↪back if invalid input
        }
```

```
        //return hunger level
        return;
    }

    //Code for hunger method
    //
    public static int hunger()
    {
        // creates a random number
        Random rand = new Random();
        int n = rand.nextInt(6); /* I want a hunger range from 0-5 however if 5␣
↪is in the brackets
                                    the range would only be 0-4 so therefore␣
↪its 6 */
        return n;
    }

    //Method to feed your pets
    //
    public static int feed(int n)
    {
        int hungerLevel = 0; //initialises the hunger level

        //which feed that they want to use
        System.out.println("Do you want to use A.premium feed (Hunger level +␣
↪4) or B.normal feed (Hunger level + 2)?");
        String feed = inputString("Please input your answer in capitals, thank␣
↪you"); //stores their answer

        //checks what they picked and then does the correct calcualtions
        if (feed.equals("A"))
        {
            hungerLevel = n + 4;
            //hunger level is a max of 5, if the level is greater than  then it␣
↪is capped at 5
            if (hungerLevel > 5)
            {
                hungerLevel = 5;
            }
        }
        else if (feed.equals("B"))
        {
            hungerLevel = n + 2;
            if (hungerLevel > 5)
            {
                hungerLevel = 5;
            }
```

```
        }
        else
        {
            System.out.println("Sorry we dont have that feed"); //catch
        }
        //prints out hunger level
        System.out.println("Your pets hunger level is now at level " +␣
↪hungerLevel);
        return hungerLevel;
    }

    //method to output how many pets were done
    public static void calculatePats(int [] n)
    {
        //outputs information from the array
        System.out.println("You brushed your pet " + n[0] + " times.");
        System.out.println("You hand petted your pet " + n[1] + " times.");
    }

    //Method to store array values
    //
    public static int [] storeArray(int a, int [] b)
    {
        boolean space = false;
        for (int i = 0; i < b.length; i++) //loops through the length of the␣
↪whole array
        {
            if (b[i] == 0) //if there is no value stored then
            {
                b[i] = a; //new value stored into array
                space = true;
                return b;
            }
            else
            {
                space = false; //no space
            }
        }

        return b;
    }

    //Method to sort array
    //
    public static void sortArray(int [] a)
    {
        int temp = 0; //creates a temporary storage
```

```
        for (int j = 0; j < a.length; j++)
        {
            for (int i = 0; i < a.length-1; i++)
            {
                if (a[i] > a[i+1]) //uses bubble sort to sort elements in list
                {
                    temp = a[i];
                    a[i] = a[i+1];
                    a[i+1] = temp;
                }
            }
        }
    }

    //method to enter store
    //
    public static void enterShop()
    {
        String enterShop = inputString("Do you want to enter the store? (y/n)");
        if (enterShop.equals("y")) //checks user input
        {
            shopInterface(); //opens the shop interface
        }
        else
        {
            System.out.println("Okay then");
        }
        return;
    }

    //Method for the main body of the shop system
    //
    public static void shopInterface()
    {
        String closeShop = ""; //variable to enter loop
        while (!(closeShop.equals("XXXX"))) //checks userinput to see if the store interface should close.
        {
            System.out.println("Hello! Welcome to the shop! This is our current stock available: ");
            System.out.println("A.Cat Tower, B. Doggy Chew Toy, C. Willow Rabbit Tunnel."); //choose the toy you want
            String toyInput = inputString("Type a letter to view the toys information (in capitals please): ");
            storeAndDisplayToyInformation(toyInput); //method that will display the toys information
```

```
            //asks if user wants to close store
            closeShop = inputString("To close the shop XXXX, else type anything␣
↪else to continue browsing.");
        }
        return;
    }

    //Method to store and display data
    //
    public static void storeAndDisplayToyInformation(String letter) //takes in␣
↪value
    {
        String nm = "";
        //creates new toys
        toys catTower = createToys("Cat Tower", 5, 50.0);
        toys dogToy = createToys("Doggy Chew Toy", 4, 3.0);
        toys rabbitTube = createToys("Willow Tunnel", 2, 6.0);

        switch (letter){ //checks what the user input is and displays correct␣
↪information
            case "A":
                nm = getToyName(catTower); //stores the name in a variable␣
↪which will be used later on
                System.out.println(toStringToys(catTower)); //prints out the␣
↪message
                break;
            case "B":
                nm = getToyName(dogToy);
                System.out.println(toStringToys(dogToy));
                break;
            case "C":
                nm = getToyName(rabbitTube);
                System.out.println(toStringToys(rabbitTube));
                break;
            default: //if none of the above are entered then a message will be␣
↪outputted
                System.out.println("Sorry I dont understand that input.");
                return; //the method will end here if the user writes the wrong␣
↪input
        }
        System.out.println(); //Space between the information and the dialogue␣
↪to purchase item
        //will have a method call here to let users purchase the toys
        purchaseOutput(toPurchase(nm));
        return;
    }
```

```
    // Input String method
    //
    public static String inputString(String message)
    {
        Scanner scanner = new Scanner(System.in); //initiate a scanner variable
        System.out.println(message); //print out the String passed
        String userInput = scanner.nextLine(); //Store the user input

        return userInput; //return the user input
    }

    //Methods to set data in the record
    //
    public static toys setToyName(toys ty, String nm) //sets the name of the toy
    {
        ty.name = nm;
        return ty;
    }

    public static toys setToyHappinessLevel(toys ty, int hl) //sets the␣
↪happiness level of the toy
    {
        ty.happinessLevel = hl;
        return ty;
    }

    public static toys setToyCost(toys ty, double cst) //sets the cost of the␣
↪toy
    {
        ty.cost = cst;
        return ty;
    }

    //Methods to return data to whereever it was called
    //
    public static String getToyName(toys ty) //return the toys name
    {
        return ty.name;
    }

    public static int getToyHappinessLevel(toys ty) //return the toys happiness␣
↪level
    {
        return ty.happinessLevel;
    }
```

```
    public static double getToyCost(toys ty) //return the cost of the toy
    {
        return ty.cost;
    }

    //Method to output data
    //
    public static String toStringToys(toys ty) //takes in a toy
    {
        String a = "The " + getToyName(ty) + " increases happiness level by " +␣
↪getToyHappinessLevel(ty)
                    + ", it costs £" + getToyCost(ty) + "0"; //outputs the␣
↪details of the record into a string
        return a; //returns the string.
    }

    //Method to create toys
    //
    public static toys createToys(String nm, int hl, double cst) //pass on␣
↪values
    {
        toys newToys = new toys(); //new instance of a toy
        newToys.name = nm; //name of the toy is set
        newToys.happinessLevel = hl; //happiness level of the toy is set
        newToys.cost = cst; //cost of the toy is set
        return newToys;
    }

    //Method for the boolean toy check
    //
    public static boolean toPurchase(String toyName) //takes the toys name
    {
        String ans = inputString("Do you want to purchase the " + toyName + "?␣
↪(y/n)"); //asks for a user output
        boolean b = (ans.equals("y")); //depending on the users answer it␣
↪returns a boolean value of either true or false
        return b;
    }

    //Method to output purchase messages
    //
    public static void purchaseOutput(Boolean a) //takes in the boolean value␣
↪from toPurchase
    {
        if (a) //if its true
        {
```

```
            System.out.println("Congratulations for purchasing an item"); //
↪congratulatory message
        }
        else //if its false
        {
            System.out.println("If you do not want this item, there are plenty␣
↪more in store.");
        }
    }
}
```

END OF LITERATE DOCUMENT