Solution :-

Package hw 7;

Public class AnimalClass Tester{

Public boolean testAnimalobject(object animal)}

if(animal instance of cow){

Cow c = (cow) animal;

Exxpected cow ex = new ExpectedCow();

if(c.aboutMe().equals(ex.expectedName())){

if(c.diet().equals(ex.expectedFood())))[

if

(c.produces().equals(ex.expectedinfo())){

return true;

}

}

}

} else if (animal instance of Tyrannosaur){

Tyrannasaur t = (Tyrannasaur ) animal;

if

(t.getName().contectEquals("Tyrannasaur Rex"))

{

if(t.roar().equals("other dinosaurs")){

return true;

}

}

}

}else if (animal instance of penguin){

Penguin p = (Penguin ) aniamal;

if(p.fightless().equals("Penguin ")){

if(p. Eat().equals("mostly finsh")){

if

(P.moment().equalsIgnorecase("waddle and swim")){

return true;

}

}

}

}

return false;

}

}

**Code 2**

Package hw 7;

Public class Cow {

Public String aboutMe(){

return "Cow";

}

Public String diet(){

return "grass, hay, and, corn";

}

Public String Produces(){

return "Milk and Cheese";

}

}

Code 4

Package hw7;

Public class hw{

Public Static void main (String [] arys) {

Animals class Tests qa = new

Animal class Tester();

Trynnosaur clearance = new Trynnosaur();

if(qa.testAnimalobject((object)clearance )){

System.out.println("Tyrranosaur class Passes");

} else {

System.out.println(" >>>> Tyrranosaur class Failed! <<<<");

}

cow mrsolearys = new cow();

if (qa.textAnimalobject(mrsolearys)){

System.out.println("Cow Class passes");

} else {

  System.out.println(">>>>Cow Class Failes<<<<");

}

}

}

**Code 5**

Package hw7:

Public Class Penguin{

Punlic String Fightless(){

return "mostly fish";

}

Public String movement (){

return "Waddle and Swim";

}

}

**Code 6**

Package hw7;

Public class Tyrannosaur{

Public String getName(){

return "Tyrannosaur Rex";

}

Public String roar(){

return " Roarrrr!";

}

Public String MyFood(){

return " other dinosaurs ";

}

We will create a card game which player will be able to play.

The notations :

1. **High card:** Simple value of the card. Lowest: 2 – Highest: Ace (1).
2. **Pair:** 2 cards with same value.
3. **3-of-a-kind:** 3 cards with same value.
4. **Flush:** 5 cards of the same suit which is usually taken as flash/flush in (texas hold'em poker. )
5. **Full house:** Combination of three of a kind and a pair
6. **Four of a kind:** Four cards of the same value
7. **Straight flush:** Straight of the same suit
8. **Royal flush Straight:** flush from Ten to Ace

In this implemenation we will have main 2 classes in **javascript :**

|  |
| --- |
| DECK |
| Deck |
| resetDeck()  shuffleDeck()  dealDeck()  isDeckEmpty()  lengthDeck() |

|  |
| --- |
| CARD |
| card  valueOfCard  suit  flipedCard  placeholderCard |
| displayCard()  flipTheCard() |

class Deck {

constructor() { //start of the constructor

this.deck = [];

this.reset(); //will add 52 cards to the deck

this.shuffle(); //will shuffle the deck

} //End of constructor

reset() {

this.deck = [];

const suits = ['Hearts', 'Diamonds', 'Clubs', 'Spades']; //array

const values = ['Ace', 2, 3, 4, 5, 6, 7, 8, 9, 10, 'Jack', 'Queen', 'King']; //values array

for (let suit in suits) {

for (let value in values) {

this.deck.push(values[value] + " of " + suits[suit]);

}

}

} //End of reset()

shuffle() { // function for shuffle()

let numberOfCards = this.deck.length;

for (var i=0; i<numberOfCards; i++) {

let j = Math.floor(Math.random() \* numberOfCards);

let tmp = this.deck[i];

this.deck[i] = this.deck[j];

this.deck[j] = tmp;

}

} //End of shuffle()

deal(){

return this.deck.pop();

} //End of deal()

isEmpty() {

return (this.deck.length==0);

} //End of isEmpty()

length() {

return this.deck.length;

}

} //End of Deck Class

class Card {

constructor(card) {

this.card = card;

const cardValues = {"Ace of Hearts":1, "2 of Hearts":2, "3 of Hearts":3, "4 of Hearts":4, "5 of Hearts":5, "6 of Hearts":6, "7 of Hearts":7, "8 of Hearts":8, "9 of Hearts":9, "10 of Hearts":10, "Jack of Hearts":11, "Queen of Hearts":12, "King of Hearts":13, "Ace of Diamonds":1, "2 of Diamonds":2, "3 of Diamonds":3, "4 of Diamonds":4, "5 of Diamonds":5, "6 of Diamonds":6, "7 of Diamonds":7, "8 of Diamonds":8, "9 of Diamonds":9, "10 of Diamonds":10, "Jack of Diamonds":11, "Queen of Diamonds":12, "King of Diamonds":13, "Ace of Clubs":1, "2 of Clubs":2, "3 of Clubs":3, "4 of Clubs":4, "5 of Clubs":5, "6 of Clubs":6, "7 of Clubs":7, "8 of Clubs":8, "9 of Clubs":9, "10 of Clubs":10, "Jack of Clubs":11, "Queen of Clubs":12, "King of Clubs":13, "Ace of Spades":1, "2 of Spades":2, "3 of Spades":3, "4 of Spades":4, "5 of Spades":5, "6 of Spades":6, "7 of Spades":7, "8 of Spades":8, "9 of Spades":9, "10 of Spades":10, "Jack of Spades":11, "Queen of Spades":12, "King of Spades":13};

this.value = cardValues[card];

this.suit = card.substring(card.indexOf(" of ")+4);

this.placeHolder = null;

this.flipped = false;

var suits = {'Hearts':0, 'Diamonds':13, 'Clubs':26, 'Spades':39 }

this.position = suits[this.suit] + this.value; //Position in a sorted deck

} //End of Constructor

displayCard(placeHolder,flipped=true) {

this.placeHolder = document.getElementById(placeHolder);

this.placeHolder.classList.add("card");

this.flipped=flipped;

if (flipped) {

this.placeHolder.style.backgroundPosition = -150\*this.position + "px";

} else {

this.placeHolder.style.backgroundPosition = "0px";

}

}

flip() {

if (this.flipped) {

this.placeHolder.style.backgroundPosition = "0px";

this.flipped=false;

} else {

this.placeHolder.style.backgroundPosition = -150\*this.position + "px";

this.flipped=true;

}

} //End of flip()

} //End of Card class

The main driver code will be as :

const deck = new Deck();

let card1,card2,card3,card4,card5,playerCard1,playerCard2;

function deal() { // function for dealCards

if (deck.length()<7) {

deck.reset();

deck.shuffle();

}

card1 = new Card(deck.deal());

card2 = new Card(deck.deal());

card3 = new Card(deck.deal());

card4 = new Card(deck.deal());

card5 = new Card(deck.deal());

playerCard1 = new Card(deck.deal());

playerCard2 = new Card(deck.deal());

card1.displayCard("card1",false);

card2.displayCard("card2",false);

card3.displayCard("card3",false);

card4.displayCard("card4",false);

card5.displayCard("card5",false);

playerCard1.displayCard("playerCard1",true);

playerCard2.displayCard("playerCard2",true);

} //End of deal()

function playTheNextStep(el) {

if (!card1.flipped) {

card1.flip();

card2.flip();

card3.flip();

el.innerHTML="Show 4<sup>th</sup> card";

} else if(!card4.flipped) {

card4.flip();

el.innerHTML="Show 5<sup>th</sup> card";

} else if(!card5.flipped) {

card5.flip();

el.innerHTML="Next Game";

} else {

deal();

el.innerHTML="Reveal first three cards.";

}

} //End of playTheNextStep()

Html Code snippet :

Board:<div id="board" style="display: block;">

<div id="card1"></div>

<div id="card2"></div>

<div id="card3"></div>

<div id="card4"></div>

<div id="card5"></div>

</div>

<hr/>

Player's card:

<div id="playerCards">

<div id="playerCard1"></div>

<div id="playerCard2"></div>

<button onclick="javascript: nextStep(this);">Show first 3 cards</button>

</div>

Css :

BODY {

background-image: radial-gradient(circle at 29% 55%, hsla(329,0%,99%,0.05) 0%, hsla(329,0%,99%,0.05) 4%,transparent 4%, transparent 44%,transparent 44%, transparent 100%),radial-gradient(circle at 85% 89%, hsla(329,0%,99%,0.05) 0%, hsla(329,0%,99%,0.05) 51%,transparent 51%, transparent 52%,transparent 52%, transparent 100%),radial-gradient(circle at 6% 90%, hsla(329,0%,99%,0.05) 0%, hsla(329,0%,99%,0.05) 53%,transparent 53%, transparent 64%,transparent 64%, transparent 100%),radial-gradient(circle at 35% 75%, hsla(329,0%,99%,0.05) 0%, hsla(329,0%,99%,0.05) 6%,transparent 6%, transparent 98%,transparent 98%, transparent 100%),radial-gradient(circle at 56% 75%, hsla(329,0%,99%,0.05) 0%, hsla(329,0%,99%,0.05) 16%,transparent 16%, transparent 23%,transparent 23%, transparent 100%),radial-gradient(circle at 42% 0%, hsla(329,0%,99%,0.05) 0%, hsla(329,0%,99%,0.05) 3%,transparent 3%, transparent 26%,transparent 26%, transparent 100%),radial-gradient(circle at 29% 28%, hsla(329,0%,99%,0.05) 0%, hsla(329,0%,99%,0.05) 51%,transparent 51%, transparent 75%,transparent 75%, transparent 100%),radial-gradient(circle at 77% 21%, hsla(329,0%,99%,0.05) 0%, hsla(329,0%,99%,0.05) 35%,transparent 35%, transparent 55%,transparent 55%, transparent 100%),radial-gradient(circle at 65% 91%, hsla(329,0%,99%,0.05) 0%, hsla(329,0%,99%,0.05) 46%,transparent 46%, transparent 76%,transparent 76%, transparent 100%),linear-gradient(45deg, rgb(83, 91, 235),rgb(76, 11, 174));

min-height:400px;

padding:20px;

color:#DDDDDD;

font-family:verdana;

}

#board, #playerCards {

display: block;

clear: both;

width:100%;

min-height: 240px;

}

.card {

box-sizing: border-box;

width: 150px;

height: 214px;

margin:10px;

float: left;

background-color: white;

background-image: url('https://i.imgur.com/yKggR5N.png');

background-position-x: 0px;

background-position-y: center;

background-repeat: none;

background-position: middle center;

border: 1px #555555 solid;

border-radius: 10px;

-webkit-box-shadow: 2px 2px 3px 0px #000000;

box-shadow: 2px 2px 3px 0px #000000;

}

**NOTE:- IN CASE OF ANY QUERY FEEL FREE TO ASK IN COMMENT ANYTIME.........HAPPY LEARNING AND KEEP CHEGGING.....**

Solution :-

Package hw 7;

Public class AnimalClass Tester{

Public boolean testAnimalobject(object animal)}

if(animal instance of cow){

Cow c = (cow) animal;

Exxpected cow ex = new ExpectedCow();

if(c.aboutMe().equals(ex.expectedName())){

if(c.diet().equals(ex.expectedFood())))[

if

(c.produces().equals(ex.expectedinfo())){

return true;

}

}

}

} else if (animal instance of Tyrannosaur){

Tyrannasaur t = (Tyrannasaur ) animal;

if

(t.getName().contectEquals("Tyrannasaur Rex"))

{

if(t.roar().equals("other dinosaurs")){

return true;

}

}

}

}else if (animal instance of penguin){

Penguin p = (Penguin ) aniamal;

if(p.fightless().equals("Penguin ")){

if(p. Eat().equals("mostly finsh")){

if

(P.moment().equalsIgnorecase("waddle and swim")){

return true;

}

}

}

}

return false;

}

}

**Code 2**

Package hw 7;

Public class Cow {

Public String aboutMe(){

return "Cow";

}

Public String diet(){

return "grass, hay, and, corn";

}

Public String Produces(){

return "Milk and Cheese";

}

}

Code 4

Package hw7;

Public class hw{

Public Static void main (String [] arys) {

Animals class Tests qa = new

Animal class Tester();

Trynnosaur clearance = new Trynnosaur();

if(qa.testAnimalobject((object)clearance )){

System.out.println("Tyrranosaur class Passes");

} else {

System.out.println(" >>>> Tyrranosaur class Failed! <<<<");

}

cow mrsolearys = new cow();

if (qa.textAnimalobject(mrsolearys)){

System.out.println("Cow Class passes");

} else {

  System.out.println(">>>>Cow Class Failes<<<<");

}

}

}

**Code 5**

Package hw7:

Public Class Penguin{

Punlic String Fightless(){

return "mostly fish";

}

Public String movement (){

return "Waddle and Swim";

}

}

**Code 6**

Package hw7;

Public class Tyrannosaur{

Public String getName(){

return "Tyrannosaur Rex";

}

Public String roar(){

return " Roarrrr!";

}

Public String MyFood(){

return " other dinosaurs ";

}

**——————I hope my answer met all your requirements......Thank You——————**

00 00 00 32 00 19 41 00 64 46 00 02 76 00 04 56 00 02 DA 00 02 81 00 01 39 00 09 10  
​andlo r0, r0, #0  
subeq r1, r1, r0, lsl #18  
andeq r4, r0, #100, #12  
r0 = r0 + 0

r1 = r1 - (r0 << 18)

r4 = r0 + 100 + 12