7.1 PRACTICE

1.Create a simple class Shape that will represent a 2-dimensional shape with line segments for edges. It should have the following instance variables: numSides (int), regular (boolean). Create at least two constructors and getter and setter methods.

package shapeclass;

public class Shape {

// Instance variables

private int numSides;

private boolean regular;

// Default constructor

public Shape() {

this.numSides = 0; // Default to 0 sides

this.regular = false; // Default to irregular

}

// Parameterized constructor

public Shape(int numSides, boolean regular) {

this.numSides = numSides;

this.regular = regular;

}

// Getter for numSides

public int getNumSides() {

return numSides;

}

// Setter for numSides

public void setNumSides(int numSides) {

this.numSides = numSides;

}

// Getter for regular

public boolean isRegular() {

return regular;

}

// Setter for regular

public void setRegular(boolean regular) {

this.regular = regular;

}

// Example method to display the shape's properties

public void displayShapeInfo() {

System.***out***.println("Number of sides: " + numSides);

System.***out***.println("Is regular: " + regular);

}

// Main method to test the class

public static void main(String[] args) {

// Using default constructor

Shape shape1 = new Shape();

shape1.displayShapeInfo();

// Using parameterized constructor

Shape shape2 = new Shape(4, true);

shape2.displayShapeInfo();

// Modifying the shape using setters

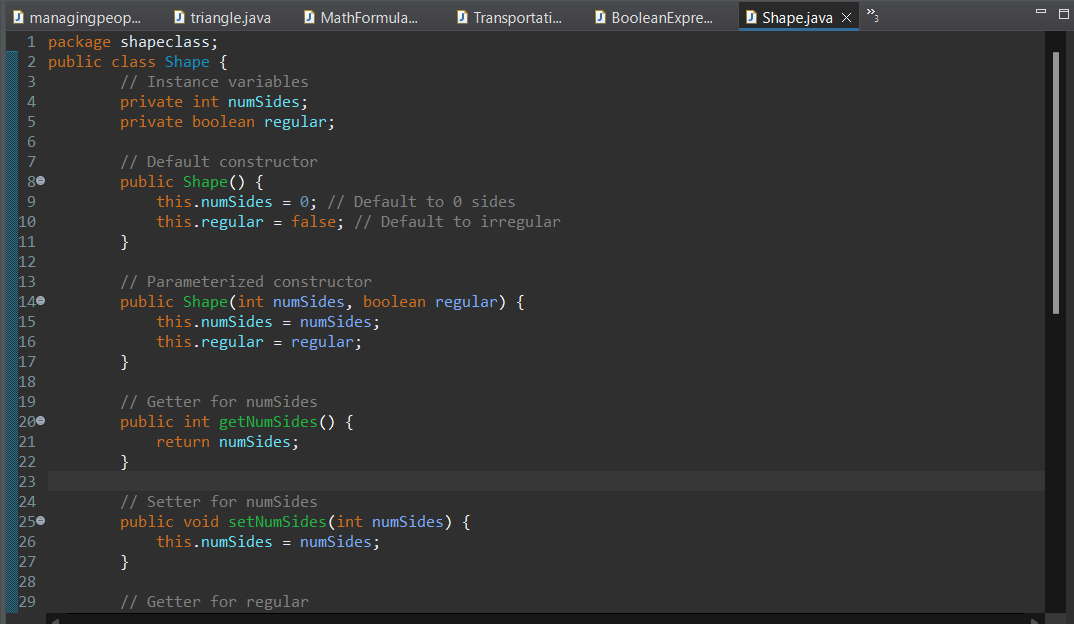
shape1.setNumSides(3);

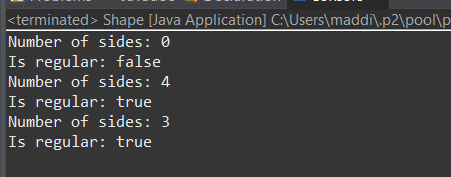
shape1.setRegular(true);

shape1.displayShapeInfo();

}

}





2. Identify the key parts of the Java Class below. Put asterisks next to all the instance variables. Place a box around each

constructor. Circle the signature of methods other than the constructor method. Place triangles around the parameters.

Underline the return types of methods.

public class Animal {

int weight, height;

double speed;

Animal() {

weight = 50;

height = 4;

speed = 2; //miles per hour

}

Animal(int w, int h, int s ) {

weight = w;

h = height;

speed = s

}

public double getTime(double miles) { //gets the number of hours to go these

miles

return miles/speed;

}

public int getWeight() {

return weight;

}

public int getHeight() {

return height;

}

public double getSpeed() {

return speed;

}

}

JAVA CODE:

package animal;

public class Animal {

int age, legs;

double speed;

// Default constructor

public Animal() {

age = 5;

legs = 4;

speed = 10.0; // meters per second

}

// Parameterized constructor

public Animal(int a, int l, double s) {

age = a;

legs = l;

speed = s;

}

// Method to calculate distance

public double calculateDistance(double time) {

return speed \* time;

}

// Getter for age

public int getAge() {

return age;

}

// Getter for legs

public int getLegs() {

return legs;

}

// Getter for speed

public double getSpeed() {

return speed;

}

// Main method to test the class

public static void main(String[] args) {

// Creating an object using the default constructor

Animal animal1 = new Animal();

System.***out***.println("Animal1 Age: " + animal1.getAge());

System.***out***.println("Animal1 Legs: " + animal1.getLegs());

System.***out***.println("Animal1 Speed: " + animal1.getSpeed() + " m/s");

System.***out***.println("Animal1 Distance in 5 seconds: " + animal1.calculateDistance(5) + " meters");

// Creating an object using the parameterized constructor

Animal animal2 = new Animal(3, 2, 20.0);

System.***out***.println("\nAnimal2 Age: " + animal2.getAge());

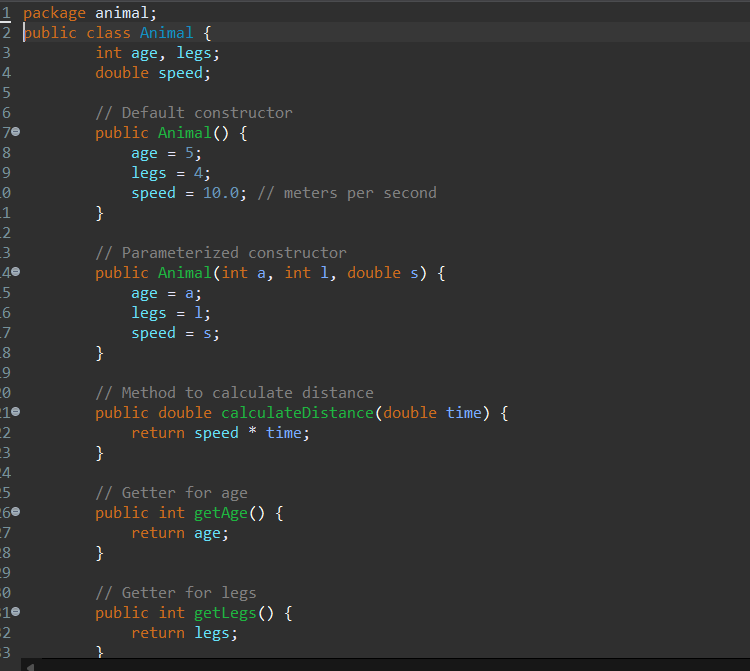
System.***out***.println("Animal2 Legs: " + animal2.getLegs());

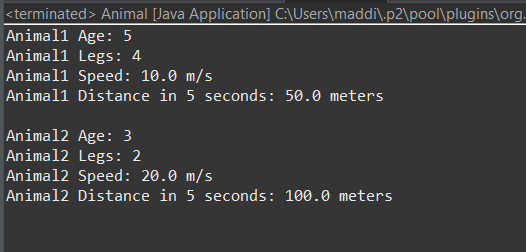
System.***out***.println("Animal2 Speed: " + animal2.getSpeed() + " m/s");

System.***out***.println("Animal2 Distance in 5 seconds: " + animal2.calculateDistance(5) + " meters");

}

}





3. Write code to create two instances of the Animal class template listed in problem #2. Be sure to use each of the two constructors provided. Then add Java code that will print the following: a. Animal #1 has a speed of \_\_\_. b. Animal #2 has a speed of \_\_\_. Be sure that the blanks are automatically filled in with the actual speeds. Use the methods provided to access the speeds. package animalspeed;

public class Animal {

int age, legs;

double speed;

// Default constructor

public Animal() {

age = 5;

legs = 4;

speed = 10.0; // meters per second

}

// Parameterized constructor

public Animal(int a, int l, double s) {

age = a;

legs = l;

speed = s;

}

// Getter for speed

public double getSpeed() {

return speed;

}

public static void main(String[] args) {

// Creating the first instance using the default constructor

Animal animal1 = new Animal();

// Creating the second instance using the parameterized constructor

Animal animal2 = new Animal(3, 2, 20.0);

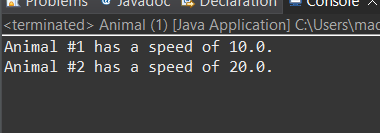
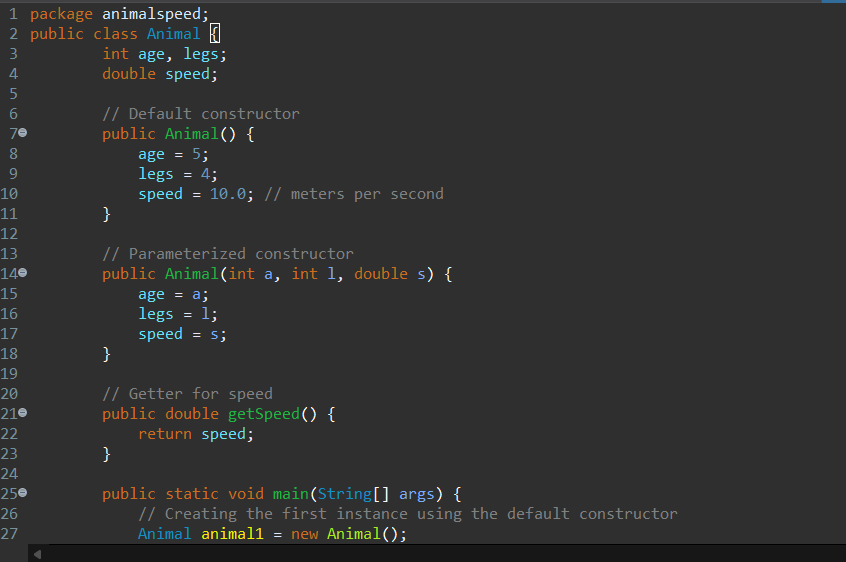
// Printing the speeds of both animals

System.***out***.println("Animal #1 has a speed of " + animal1.getSpeed() + ".");

System.***out***.println("Animal #2 has a speed of " + animal2.getSpeed() + ".");

}

}



4. Write a class Student. It should have the following instance variables for the name, credits, grade point average (GPA), and quality Points. Create a constructor method. Create two other methods as follows: a. A method that will return the current grade point average which will be the quality points divided by the credits. b. A method that will take in the credits for a class or semester along with the quality points. It should update the credits, the quality points, and the GPA.

package student;

public class Student {

// Instance variables

private String name;

private int credits;

private double gpa;

private double qualityPoints;

// Constructor

public Student(String name, int credits, double qualityPoints) {

this.name = name;

this.credits = credits;

this.qualityPoints = qualityPoints;

this.gpa = calculateGPA();

}

// Method to calculate the GPA

public double calculateGPA() {

if (credits == 0) {

return 0.0; // To avoid division by zero

}

return qualityPoints / credits;

}

// Method to update credits, quality points, and GPA

public void updateRecord(int newCredits, double newQualityPoints) {

credits += newCredits;

qualityPoints += newQualityPoints;

gpa = calculateGPA(); // Recalculate GPA after updating

}

// Method to return the current GPA

public double getGPA() {

return gpa;

}

// Getter methods for other instance variables (optional)

public String getName() {

return name;

}

public int getCredits() {

return credits;

}

public double getQualityPoints() {

return qualityPoints;

}

// Main method to test the class

public static void main(String[] args) {

// Create a Student object

Student student = new Student("John Doe", 30, 120.0);

// Print initial GPA

System.***out***.println("Initial GPA: " + student.getGPA());

// Update the student's record with new credits and quality points

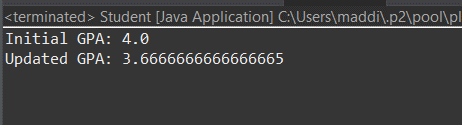
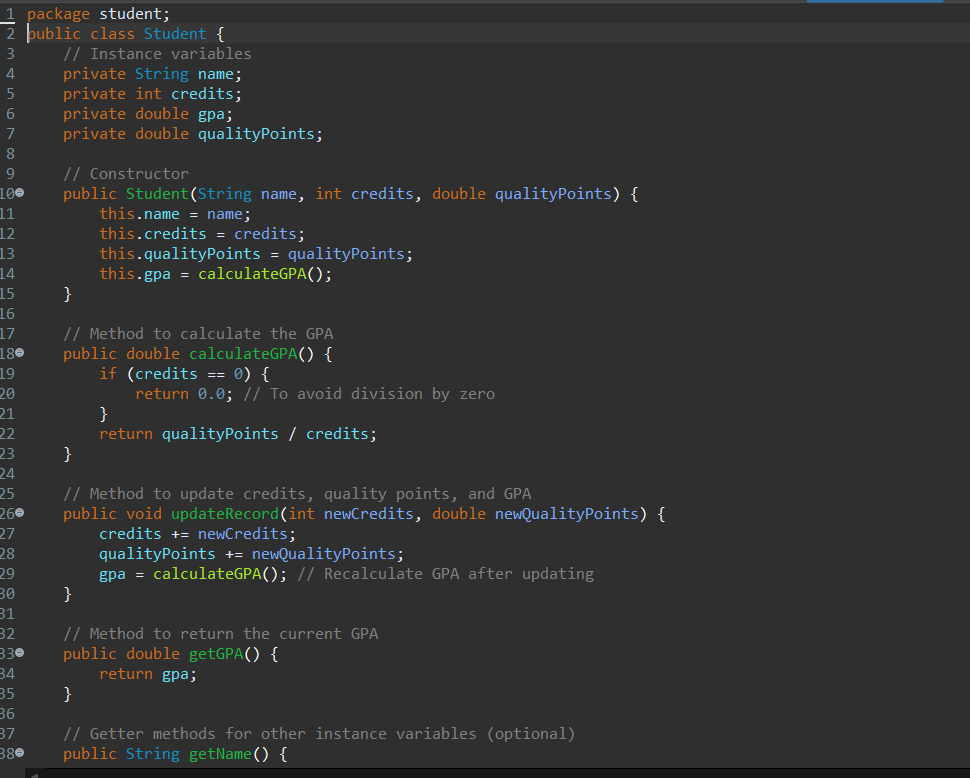
student.updateRecord(15, 45.0);

// Print updated GPA

System.***out***.println("Updated GPA: " + student.getGPA());

}

}



5. Using the class you created in #4, create three instances of the Student Class from the table below: Name Credits Quality Points Mary Jones 14 46 John Stiner 60 173 Ari Samala 31 69.

package studentgpa;

public class Student {

// Instance variables

private String name;

private int credits;

private double gpa;

private double qualityPoints;

// Constructor

public Student(String name, int credits, double qualityPoints) {

this.name = name;

this.credits = credits;

this.qualityPoints = qualityPoints;

this.gpa = calculateGPA();

}

// Method to calculate the GPA

public double calculateGPA() {

if (credits == 0) {

return 0.0; // To avoid division by zero

}

return qualityPoints / credits;

}

// Method to update credits, quality points, and GPA

public void updateRecord(int newCredits, double newQualityPoints) {

credits += newCredits;

qualityPoints += newQualityPoints;

gpa = calculateGPA(); // Recalculate GPA after updating

}

// Method to return the current GPA

public double getGPA() {

return gpa;

}

// Getter methods for other instance variables

public String getName() {

return name;

}

public int getCredits() {

return credits;

}

public double getQualityPoints() {

return qualityPoints;

}

// Main method to test the class

public static void main(String[] args) {

// Create three instances of the Student class

Student student1 = new Student("Mary Jones", 14, 46.0);

Student student2 = new Student("John Stiner", 60, 173.0);

Student student3 = new Student("Ari Samala", 31, 69.0);

// Print the details of each student

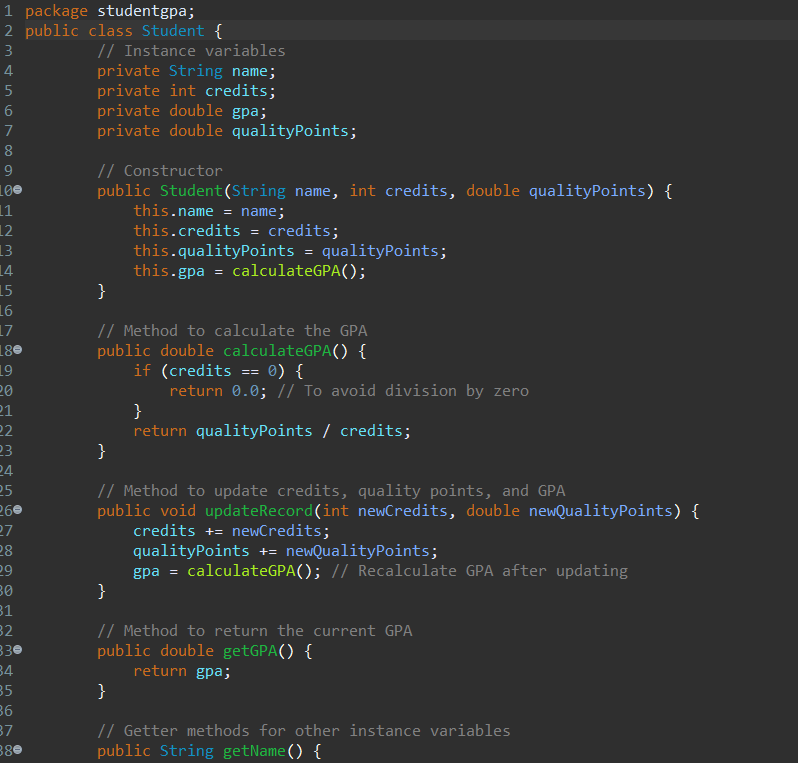
System.***out***.println(student1.getName() + " has a GPA of " + student1.getGPA());

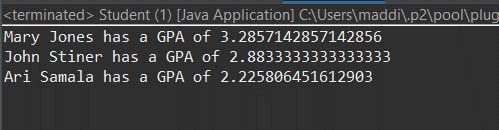
System.***out***.println(student2.getName() + " has a GPA of " + student2.getGPA());

System.***out***.println(student3.getName() + " has a GPA of " + student3.getGPA());

}

}





6. Using the instance variables created in #5, add 13 credits and 52 quality points to the student “Ari Samala”.

package gpa;

public class Student {

// Instance variables

private String name;

private int credits;

private double gpa;

private double qualityPoints;

// Constructor

public Student(String name, int credits, double qualityPoints) {

this.name = name;

this.credits = credits;

this.qualityPoints = qualityPoints;

this.gpa = calculateGPA();

}

// Method to calculate the GPA

public double calculateGPA() {

if (credits == 0) {

return 0.0; // To avoid division by zero

}

return qualityPoints / credits;

}

// Method to update credits, quality points, and GPA

public void updateRecord(int newCredits, double newQualityPoints) {

credits += newCredits;

qualityPoints += newQualityPoints;

gpa = calculateGPA(); // Recalculate GPA after updating

}

// Method to return the current GPA

public double getGPA() {

return gpa;

}

// Getter methods for other instance variables

public String getName() {

return name;

}

public int getCredits() {

return credits;

}

public double getQualityPoints() {

return qualityPoints;

}

// Main method to test the class

public static void main(String[] args) {

// Create three instances of the Student class

Student student1 = new Student("Mary Jones", 14, 46.0);

Student student2 = new Student("John Stiner", 60, 173.0);

Student student3 = new Student("Ari Samala", 31, 69.0);

// Print the initial GPA of Ari Samala

System.***out***.println("Initial GPA of " + student3.getName() + ": " + student3.getGPA());

// Add 13 credits and 52 quality points to Ari Samala's record

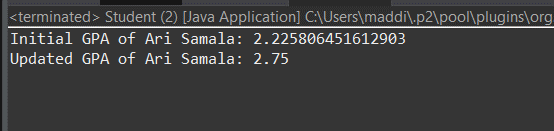
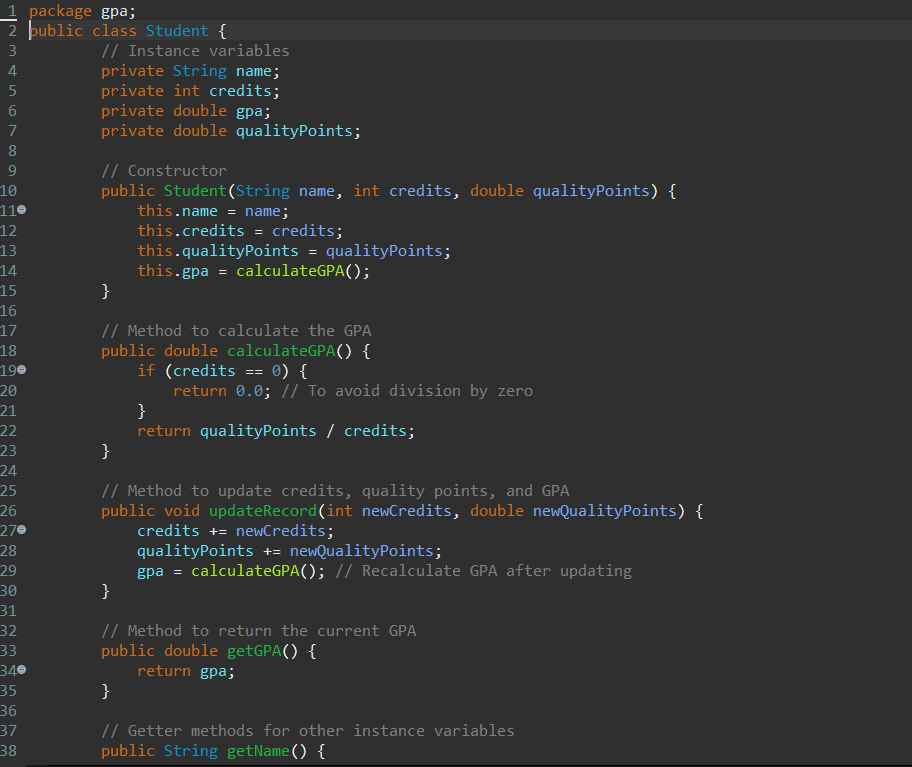
student3.updateRecord(13, 52.0);

// Print the updated GPA of Ari Samala

System.***out***.println("Updated GPA of " + student3.getName() + ": " + student3.getGPA());

}

}



7. Using the Card class from the slides and test the program to make sure it works. Add a second random Card. Code is

included below:

public class Card{

String suit,name;

int points;

Card(int n1, int n2){

suit = getSuit(n1);

name = getName(n2);

points = getPoints(name);

}

public String toString(){

return "The " + name + " of " + suit;

}

public String getName(int i){

if(i == 1) return "Ace";

if(i == 2) return "Two";

if(i == 3) return "Three";

if(i == 4) return "Four";

if(i == 5) return "Five";

if(i == 6) return "Six";

if(i == 7) return "Seven";

if(i == 8) return "Eight";

if(i == 9) return "Nine";

if(i == 10) return "Ten";

if(i == 11) return "Jack";

Copyright © 2022, Oracle and/or its affiliates. Oracle, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners. 4

if(i == 12) return "Queen";

if(i == 13) return "King";

return "error";

}

public int getPoints(String n){

if(n == "Jack" ||n == "Queen" ||n == "King"||n == "Ten")

return 10;

if(n == "Two")

return 2;

if(n == "Three")

return 3;

if(n == "Four")

return 4;

if(n == "Five")

return 5;

if(n == "Six")

return 6;

if(n == "Seven")

return 7;

if(n == "Eight")

return 8;

if(n == "Nine")

return 9;

if(n == "Ace")

return 1;

return -1;

}

public String getSuit(int i){

if(i == 1) return "Diamonds";

if(i == 2) return "Clubs";

if(i == 3) return "Spades";

if(i == 4) return "Hearts";

return "error";

}

}

public class Main {

public static void main(String args[]){

int suitNumber = (int)(Math.random()\*4.0+1);

int faceNumber = (int)(Math.random()\*13.0+1);

Card newCard = new Card(suitNumber,faceNumber);

System.out.println(newCard);

}

}

JAVA CODE:

package card;

public class Card {

String suit, name;

int points;

// Constructor

public Card(int suitNumber, int faceNumber) {

suit = getSuit(suitNumber);

name = getName(faceNumber);

points = getPoints(name);

}

// Method to get the string representation of the card

@Override

public String toString() {

return "The " + name + " of " + suit;

}

// Method to get the name of the card based on face number

private String getName(int i) {

switch (i) {

case 1: return "Ace";

case 2: return "Two";

case 3: return "Three";

case 4: return "Four";

case 5: return "Five";

case 6: return "Six";

case 7: return "Seven";

case 8: return "Eight";

case 9: return "Nine";

case 10: return "Ten";

case 11: return "Jack";

case 12: return "Queen";

case 13: return "King";

default: return "error";

}

}

// Method to get the points of the card based on name

private int getPoints(String n) {

switch (n) {

case "Jack":

case "Queen":

case "King":

case "Ten": return 10;

case "Two": return 2;

case "Three": return 3;

case "Four": return 4;

case "Five": return 5;

case "Six": return 6;

case "Seven": return 7;

case "Eight": return 8;

case "Nine": return 9;

case "Ace": return 1;

default: return -1;

}

}

// Method to get the suit of the card based on suit number

private String getSuit(int i) {

switch (i) {

case 1: return "Diamonds";

case 2: return "Clubs";

case 3: return "Spades";

case 4: return "Hearts";

default: return "error";

}

}

}

package card;

public class Main {

public static void main(String[] args) {

// Generate random suit and face numbers

int suitNumber1 = (int) (Math.random() \* 4) + 1; // Values between 1 and 4

int faceNumber1 = (int) (Math.random() \* 13) + 1; // Values between 1 and 13

int suitNumber2 = (int) (Math.random() \* 4) + 1; // Values between 1 and 4

int faceNumber2 = (int) (Math.random() \* 13) + 1; // Values between 1 and 13

// Create two random Card instances

Card card1 = new Card(suitNumber1, faceNumber1);

Card card2 = new Card(suitNumber2, faceNumber2);

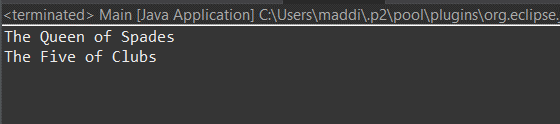
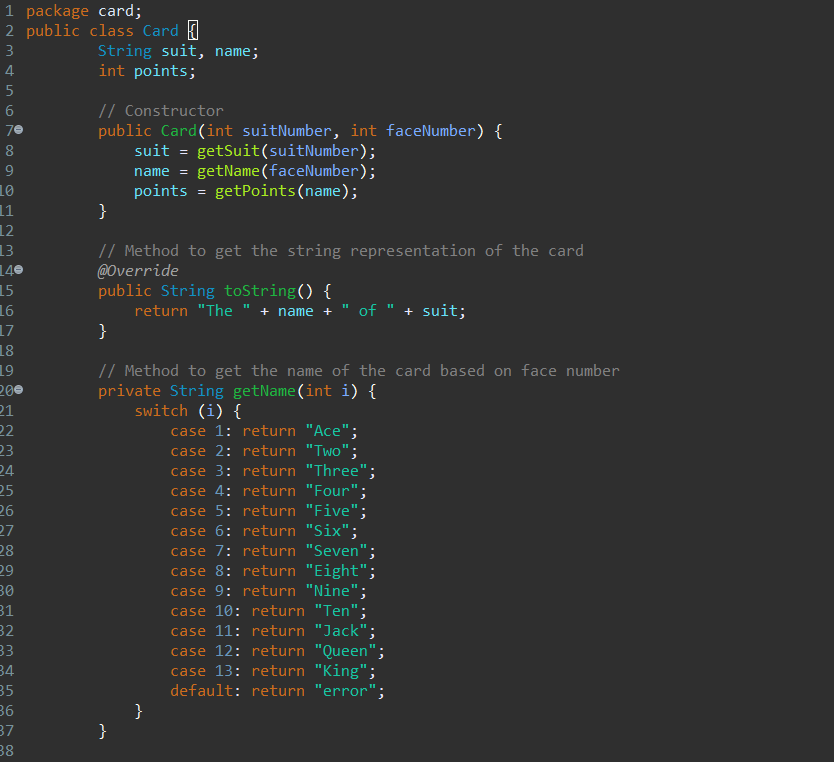
// Print the details of both cards

System.out.println(card1);

System.out.println(card2);

}

}



9. Add code to the Main class in exercise #7 to the following: a. Display the total point value for the two random cards. b. Ask the user if they would like another card. If they say yes display the new card and the points for all 3 cards in their “Hand”. c. Loop to allow the user to continue to add cards to the hand until the number of points goes over 21 or the user decides not to add any more cards or the total number of cards is 5.

JAVA CODE:

package card;

import java.util.ArrayList;

import java.util.Scanner;

public class Main1 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.***in***);

ArrayList<Card> hand = new ArrayList<>();

int totalPoints = 0;

final int MAX\_CARDS = 5;

// Add two random cards to the hand

for (int i = 0; i < 2; i++) {

int suitNumber = (int) (Math.*random*() \* 4) + 1; // Values between 1 and 4

int faceNumber = (int) (Math.*random*() \* 13) + 1; // Values between 1 and 13

Card card = new Card(suitNumber, faceNumber);

hand.add(card);

totalPoints += card.points;

}

// Display the initial hand

System.***out***.println("Initial hand:");

*displayHand*(hand);

System.***out***.println("Total points: " + totalPoints);

// Loop to add more cards until the conditions are met

while (totalPoints <= 21 && hand.size() < MAX\_CARDS) {

System.***out***.println("Would you like another card? (yes/no)");

String response = scanner.nextLine();

if (response.equalsIgnoreCase("yes")) {

int suitNumber = (int) (Math.*random*() \* 4) + 1; // Values between 1 and 4

int faceNumber = (int) (Math.*random*() \* 13) + 1; // Values between 1 and 13

Card card = new Card(suitNumber, faceNumber);

hand.add(card);

totalPoints += card.points;

// Display the updated hand

System.***out***.println("New card:");

System.***out***.println(card);

System.***out***.println("Updated hand:");

*displayHand*(hand);

System.***out***.println("Total points: " + totalPoints);

if (totalPoints > 21) {

System.***out***.println("You have gone over 21 points!");

break;

}

} else {

break;

}

}

// Close the scanner

scanner.close();

}

// Method to display all cards in the hand

private static void displayHand(ArrayList<Card> hand) {

for (Card card : hand) {

System.***out***.println(card);

}

}

}

