**Задание 2.**

***1. Для классов из Задания 1 текущего ДЗ реализовать 2 интерфейса: IDrawable, IPrintable***. Первый имеет метод draw для отрисовки (фигуры) на консоли (псевдографикой), а второй имеет метод print для печати в консоли.

***2. Фигуры, которые вы хотите рисовать (минимум 2), должны имплементировать интерфейс IDrawable.*** Заимплиментить интерфейс у данных фигур.

***3. Для фигур, которые вы хотите печатать (по сути для всех), реализовать интерфейс IPrintable.***

***4. Сделать класс «Сцена фигур»***, который содержит массив объектов «геометрическая фигура» (можно использовать класс составная фигура из задания 1).

***5. Сделать в этом классе метод «отрисовать всю сцену» и «напечатать всю сцену»***, которые в цикле пробегают все фигуры массива, и пытаются вызвать нужный интерфейс. Если очередная фигура реализует данный интерфейс, то вызывается метод, иначе выводится сообщение, что интерфейс не поддерживается.

P.S. Нужно использовать instanceof или иные операторы каста типов.

P.P.S. Вместо массива фигур можно использовать список List фигур.

P.P.P.S. Так же класс «Составная фигура» при желании можно сделать наследником базовой фигуры и поэкспериментировать с ситуацией, когда в составную фигуру входят другие составные фигуры (когда в сцену входят другие сцены).

***abstract class GeometricFigure***

package com.company;

//Абстрактный класс «геометрическая фигура»  
public abstract class GeometricFigure *{* // имя фигуры  
 protected String name;  
  
 // конструкторы  
 //1. конструктор по умолчанию  
 public GeometricFigure*() {* name = "";  
 *}* //2. конструктор с 1 параметром  
 public GeometricFigure*(*String *name) {* this.name = *name*;  
 *}* // методы  
 // 1. метод получения площади  
 public abstract double S*()*;  
  
 // 2. метод получения периметра  
 public abstract double P*()*;  
  
 // 3. метод представления класса в виде строки  
 @Override  
 public String toString*() {* return name;  
 *}  
}*

***class Square extends GeometricFigure implements IPrintable, IDrawable***

package com.company;  
  
// класс квадрат  
public class Square extends GeometricFigure implements IPrintable, IDrawable *{* // поля квадрата  
 private double a;  
  
 // конструкторы  
 //1. конструктор без параметров  
 public Square*() {* a = 0;  
 *}* //2. конструктор с двумя параметрами  
 public Square*(*String *name*, double *a) {* super*(name)*;  
 this.a = *a*;  
 *}* // методы  
 // 1. метод получения площади - перегрузка метода S() базового класса  
 @Override  
 public double S*() {* return a \* a;  
 *}* // 2. метод получения периметра - перегрузка метода P() базового класса  
 @Override  
 public double P*() {* return a \* 4;  
 *}* // 3. метод представления класса в виде строки - перегрузка метода toString  
 @Override  
 public String toString*() {* return name + ": a = " + a;  
 *}* // 4. метод печати фигуры на консоль  
 @Override  
 public void consolePrint*() {* System.*out*.println*(*this*)*;  
 *}* // 5. метод отрисовки фигуры в консоль  
 @Override  
 public void consoleDraw*() {* for *(*int *i* = 0; *i* < *(*int*)* a; *i*++*) {* for *(*int *j* = 0; *j* < *(*int*)* a; *j*++*)* System.*out*.print*(*" \* "*)*;  
 System.*out*.println*()*;  
 *}  
 }  
}*

***class Rectangle extends GeometricFigure implements IPrintable, IDrawable***

package com.company;  
  
// класс прямоугольник  
public class Rectangle extends GeometricFigure implements IPrintable, IDrawable *{* // поля прямоугольника  
 private double a;  
 private double b;  
  
 // конструкторы  
 //1.конструктор без параметров  
 public Rectangle*() {* a = b = 0;  
 *}* //2. конструктор с 3 параметрами  
 public Rectangle*(*String *name*, double *a*, double *b) {* super*(name)*;  
 this.a = *a*;  
 this.b = *b*;  
 *}* //3. делегируем конструктор с 3 параметрами  
 public Rectangle*(*String *name*, double *x) {* this*(name*, *x*, *x)*;  
 *}* // методы  
 // 1. метод получения площади - перегрузка метода S() базового класса  
 @Override  
 public double S*() {* return a \* b;  
 *}* // 2. метод получения периметра - перегрузка метода P() базового класса  
 @Override  
 public double P*() {* return *(*a + b*)* \* 2;  
 *}* // 3. метод представления класса в виде строки - перегрузка метода toString  
 @Override  
 public String toString*() {* return name + ": a = " + a + "," + " b = " + b;  
 *}* // 4. метод печати фигуры на консоль  
 @Override  
 public void consolePrint*() {* System.*out*.println*(*this*)*;  
 *}* // 5. метод отрисовки фигуры в консоль  
 @Override  
 public void consoleDraw*() {* for *(*int *i* = 0; *i* < *(*int*)* a; *i*++*) {* for *(*int *j* = 0; *j* < *(*int*)* b; *j*++*)* System.*out*.print*(*" \* "*)*;  
 System.*out*.println*()*;  
 *}  
 }*}

***class Circle extends GeometricFigure implements IPrintable***

package com.company;  
  
// класс окружность  
public class Circle extends GeometricFigure implements IPrintable *{* // поля окружности  
 private double r; // радиус окружности  
  
 // конструкторы  
 //1. конструктор без параметров  
 public Circle*() {* r = 0;  
 *}* //2. конструктор с одним параметром  
 public Circle*(*double *r) {* this.r = *r*;  
 *}* //3. конструктор с двумя параметрами  
 public Circle*(*String *name*, double *r) {* super*(name)*;  
 this.r = *r*;  
 *}* // методы  
 // 1. метод получения площади - перегрузка метода S() базового класса  
 @Override  
 public double S*() {* return Math.*PI* \* r \* r;  
 *}* // 2. метод получения периметра - перегрузка метода P() базового класса  
 @Override  
 public double P*() {* return 2 \* Math.*PI* \* r;  
 *}* // 3. метод представления класса в виде строки - перегрузка метода toString  
 @Override  
 public String toString*() {* return name + ": r = " + r;  
 *}* // реализация интерфейса IPrintable  
 @Override  
 public void consolePrint*() {* System.*out*.println*(*this*)*; // выводим toString  
 *}  
  
}*

***class Triangle extends GeometricFigure implements IPrintable, IDrawable***

package com.company;  
  
// класс треугольник  
public class Triangle extends GeometricFigure implements IPrintable, IDrawable *{* // поля треугольника  
 private double a; // сторона a  
 private double b; // сторона b  
 private double c; // сторона b  
  
 // 1. конструктор без параметров  
 public Triangle*() {* this.name = "name";  
 this.a = 1;  
 this.b = 1;  
 this.c = 1;  
 *}* // 2. конструктор с 4-мя параметрами  
 public Triangle*(*String *name*, double *a*, double *b*, double *c) {* this.name = *name*;  
 this.a = *a*;  
 this.b = *b*;  
 this.c = *c*;  
 *}* // методы  
 // 1. метод получения площади - перегрузка метода S() базового класса  
 @Override  
 public double S*() {* double *halfP* = P*()* / 2;  
 return Math.*sqrt(halfP* \* *(halfP* - a*)* \* *(halfP* - b*)* \* *(halfP* - c*))*;  
 *}* // 2. метод получения периметра - перегрузка метода P() базового класса  
 @Override  
 public double P*() {* return a + b + c;  
 *}* // 3. метод представления класса в виде строки - перегрузка метода toString  
 @Override  
 public String toString*() {* return name + ": a = " + a + ", b = " + b + ", c = " + c;  
 *}* // 4. метод печати фигуры на консоль  
 @Override  
 public void consolePrint*() {* System.*out*.println*(*this*)*;  
 *}* // 5. метод отрисовки фигуры в консоль  
 @Override  
 public void consoleDraw*() {* // Прямоугольный треугольник  
 for *(*int *i* = 1; *i* <= *(*int*)* a; ++*i) {* //Получаем строки  
 for *(*int *j* = *(*int*)* b; *j* > *i*; --*j) {* //Получаем пробелы  
 System.*out*.print*(*" "*)*;  
 *}* for *(*int *j* = 1; *j* < 2 \* *i*; ++*j) {* // Выводим фигуру звездочками  
 System.*out*.print*(*"\*"*)*;  
 *}* System.*out*.println*()*;//Переходим на новую строку после прохода основного цикла for  
 *}  
 }  
}*

***class FigureScene implements IPrintable, IDrawable***

package com.company;  
  
import java.util.Arrays;  
  
// класс «Сцена фигур»  
public class FigureScene implements IPrintable, IDrawable *{* // поля  
 private GeometricFigure*[]* figures;  
  
 // конструктор с одним параметром  
 public FigureScene*(*int *figure) {* figures = new GeometricFigure*[figure]*;  
 *}* //setter  
 public void setFigure*(*GeometricFigure *figure*, int *pos) {* figures*[pos]* = *figure*;  
 *}* // методы  
 // 1. метод добавления новой фигуры в массив  
 public void addFigure*() {* figures = Arrays.*copyOf(*figures, figures.length + 1*)*;  
 *}* // 2. метод вывода параметров всех фигур  
 public void allOptionsFigure*() {* for *(*GeometricFigure *figure* : figures*) {* if *(figure* != null*)* System.*out*.println*(figure* + ", S = " + *figure*.S*()* + ", P = " + *figure*.P*())*;  
 *}  
 }* // 3. метод вывода общей площади фигур  
 public double generalAreaFigure*() {* double *res* = 0;  
 for *(*GeometricFigure *figure* : figures*) {  
 res* += *figure*.S*()*;  
 *}* return *res*;  
 *}* // 4. метод вывода общего периметра фигур  
 public double generalPerimeterFigure*() {* double *res* = 0;  
 for *(*GeometricFigure *figure* : figures*) {  
 res* += *figure*.P*()*;  
 *}* return *res*;  
 *}* // 5. метод представления класса в виде строки - перегрузка метода toString  
 @Override  
 public String toString*() {* StringBuilder *sb* = new StringBuilder*()*;  
  
 for *(*GeometricFigure *figure* : figures*) {* if *(figure* == null*)  
 sb*.append*(*"empty seat"*)*;  
 else  
 *sb*.append*(figure)*.append*(*"\n"*)*;  
 *}* return *sb*.toString*()*;  
 *}* // 6. метод печати фигур на консоль  
 @Override  
 public void consolePrint*() {* System.*out*.println*(*toString*())*;  
 *}* // 7. метод отрисовки фигур в консоль  
 @Override  
 public void consoleDraw*() {* for *(*GeometricFigure *figure* : figures*) {* if *(figure* instanceof IDrawable*) {  
 ((*IDrawable*) figure)*.consoleDraw*()*;  
 *}* else *{* System.*out*.println*(*"Error: " + *figure* + " IDrawable не поддерживается"*)*;  
 *}* System.*out*.println*()*;  
 *}  
 }  
}*

***interface IPrintable***

package com.company;  
  
// интерфейс "Способный быть напечатнным"  
public interface IPrintable *{* void consolePrint*()*;  
*}*

***interface IDrawable***

package com.company;  
  
// интерфейс "Способный быть нарисованным"  
public interface IDrawable *{* void consoleDraw*()*;  
*}*

***class ShowFigure***

package com.company;  
  
public class ShowFigure {  
  
 //метод тестирования фигуры  
 static void printGeometricFigure(GeometricFigure *figure*) {  
 System.*out*.println("S = " + *figure*.S());  
 System.*out*.println("P = " + *figure*.P());  
 }  
  
 // тестирование фигуры и интерфейсов  
 static void testFigureAndInterfaces(GeometricFigure *figure*) {  
 // безопасный вызов печати  
 if (*figure* instanceof IPrintable)  
 *makeConsolePrint*((IPrintable) *figure*);  
 else  
 System.*out*.println("IPrintable не поддерживается");  
 // безопасный вызов отрисовки  
 if (*figure* instanceof IDrawable)  
 *makeConsoleDraw*((IDrawable) *figure*);  
 else  
 System.*out*.println("IDrawable не поддерживается");  
 }  
  
 // тестирование интерфейса IPrintable  
 static void makeConsolePrint(IPrintable *printable*) {  
 *printable*.consolePrint();  
 }  
  
 // тестирование интерфейса IDrawable  
 static void makeConsoleDraw(IDrawable *drawable*) {  
 *drawable*.consoleDraw();  
 }  
  
 public static void main(String[] *args*) {  
  
 System.*out*.println("1.Тестирование интерфейсов \"IPrintable и IDrawable \": \n");  
 // содаём объект прямоугольник  
 Rectangle *rect* = new Rectangle("Rectangle", 4, 8);  
 *testFigureAndInterfaces*(*rect*);//печать и отрисовка фигуры в консоль  
 *printGeometricFigure*(*rect*);//вывод площади и периметра  
 System.*out*.println();  
  
 //содаём объект треугольник  
 Triangle *triangle* = new Triangle("Triangle", 4, 5, 6);  
 *testFigureAndInterfaces*(*triangle*);//печать и отрисовка фигуры в консоль  
 *printGeometricFigure*(*triangle*);//вывод площади и периметра  
 System.*out*.println();  
  
 //содаём объект квадрат  
 Square *square* = new Square("Square", 4);  
 *testFigureAndInterfaces*(*square*);//печать и отрисовка фигуры в консоль  
 *printGeometricFigure*(*square*);//вывод площади и периметра  
 System.*out*.println();  
  
 //содаём объект окружность  
 Circle *circle* = new Circle("Circle", 3);  
 *testFigureAndInterfaces*(*circle*);//печать и отрисовка фигуры в консоль  
 *printGeometricFigure*(*circle*);//вывод площади и периметра  
 System.*out*.println();  
  
 System.*out*.println("2.Тестирование класса \"Сцена фигур\": ");  
 FigureScene *comp* = new FigureScene(4);  
 // поместили прямоугольник в класс «Сцена фигур», на 1-е место в массиве  
 *comp*.setFigure(*rect*, 0);  
  
 // поместили треугольник в класс «Сцена фигур», на 2-е место в массиве  
 *comp*.setFigure(*triangle*, 1);  
  
 // поместили квадрат в класс «Сцена фигур», на 3-е место в массиве  
 *comp*.setFigure(*square*, 2);  
  
 // поместили окружность в класс «Сцена фигур», на 4-е место в массиве  
 *comp*.setFigure(*circle*, 3);  
 *makeConsolePrint*(*comp*);  
 *makeConsoleDraw*(*comp*);  
  
 //Вывод в консоль общей площади и периметра всех фигур массива  
 System.*out*.println("Общая площадь S = " + *comp*.generalAreaFigure());  
 System.*out*.println("Общий периметр P = " + *comp*.generalPerimeterFigure());  
  
 }  
}

***Тест:***





