Московский государственный технический университет им. Н.Э. Баумана

Факультет “Информатика и системы управления”
Кафедра “Системы обработки информации и управления”

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №2
«Классы на языке TypeScript»

Выполнил:
Студент группы ИУ5-31Б
Попов А.С.
Подпись и дата:

Проверил:
Преподаватель каф. ИУ5
Гапанюк Ю.Е.
Подпись и дата:

Москва, 2025 г.

Задание

Разработать программу, реализующую работу с классами.

1. Программа должна быть разработана в виде консольного приложения на языке TypeScript.
2. Абстрактный класс «Геометрическая фигура» содержит виртуальный метод для вычисления площади фигуры.
3. Класс «Прямоугольник» наследуется от «Геометрическая фигура». Ширина и высота объявляются как свойства (property). Класс должен содержать конструктор по параметрам «ширина» и «высота».
4. Класс «Квадрат» наследуется от «Прямоугольник». Класс должен содержать конструктор по длине стороны.
5. Класс «Круг» наследуется от «Геометрическая фигура». Радиус объявляется как свойство (property). Класс должен содержать конструктор по параметру «радиус».
6. Для классов «Прямоугольник», «Квадрат», «Круг» переопределить виртуальный метод Object.ToString(), который возвращает в виде строки основные параметры фигуры и ее площадь.
7. Разработать интерфейс IPrint. Интерфейс содержит метод Print(), который не принимает параметров и возвращает void. Для классов «Прямоугольник», «Квадрат», «Круг» реализовать наследование от интерфейса IPrint. Переопределяемый метод Print() выводит на консоль информацию, возвращаемую переопределенным методом ToString().

Текст программы

index.ts

```
interface Printable {
  Print(): void
}

abstract class Figure {
  constructor(public name: string) { }

  public abstract GetArea(): number
}

class Rectangle
  extends Figure
  implements Printable {
  constructor(
    name: string,
    private width: number,
    private height: number,
  ) {
    super(name)
  }

  Print() {
    console.log(`${this.name} (${this.width}x${this.height}),
area=${this.GetArea()}`)
  }

  GetArea() {
    return this.width * this.height
  }
}

class Square
  extends Rectangle
  implements Printable {
  constructor(
    name: string,
    private side: number,
  ) {
    super(name, side, side)
  }

  override Print() {
    console.log(`${this.name} (${this.side}x${this.side}),
area=${this.GetArea()}`)
```

```
  }

  override GetArea() {
    return this.side * this.side
  }
}

class Circle
  extends Figure
  implements Printable {
  constructor(
    name: string,
    private radius: number,
  ) {
    super(name)
  }

  Print() {
    console.log(`${this.name} with radius=${this.radius},
area=${this.GetArea()}`)
  }

  GetArea() {
    return Math.PI * this.radius * this.radius
  }
}

function Print(...printables: Printable[]) {
  for (const printable of printables) {
    printable.Print()
  }
}

function GetTotalAreaOfFigures(...figures: Figure[]) {
  let area = 0
  for (const figure of figures) {
    console.info(`Adding figure "${figure.name}" with
area=${figure.GetArea()}...`)
    area += figure.GetArea()
  }
  return area
}

const square = new Square('Square', 3)
const rectangle = new Rectangle('Rectangle', 3, 4)
const circle = new Circle('Circle', 5)

Print(square, rectangle, circle)
console.log(GetTotalAreaOfFigures(square, rectangle, circle))
```

Скриншоты работы программы

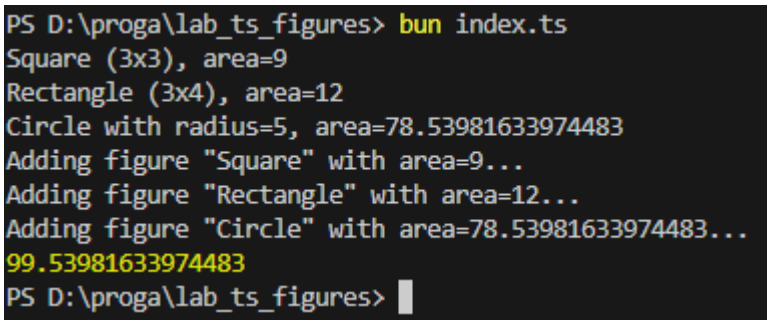

Рис. 1 – Работа приложения