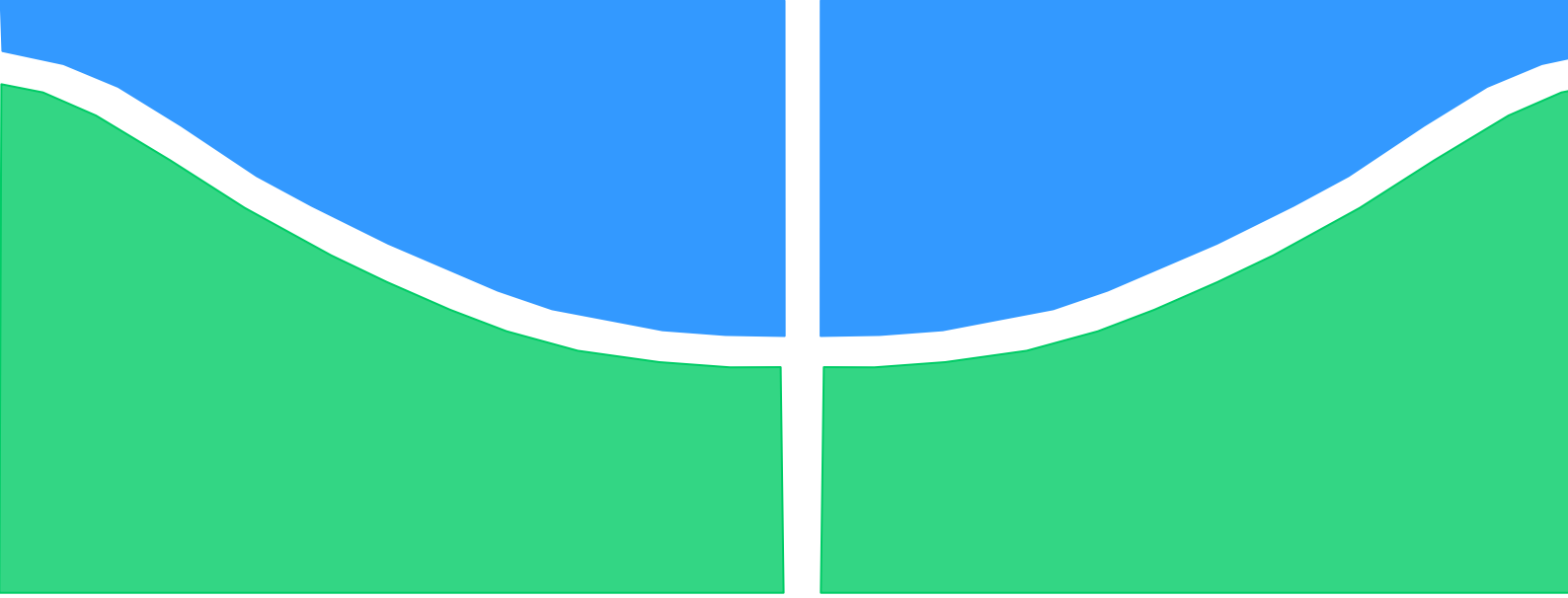


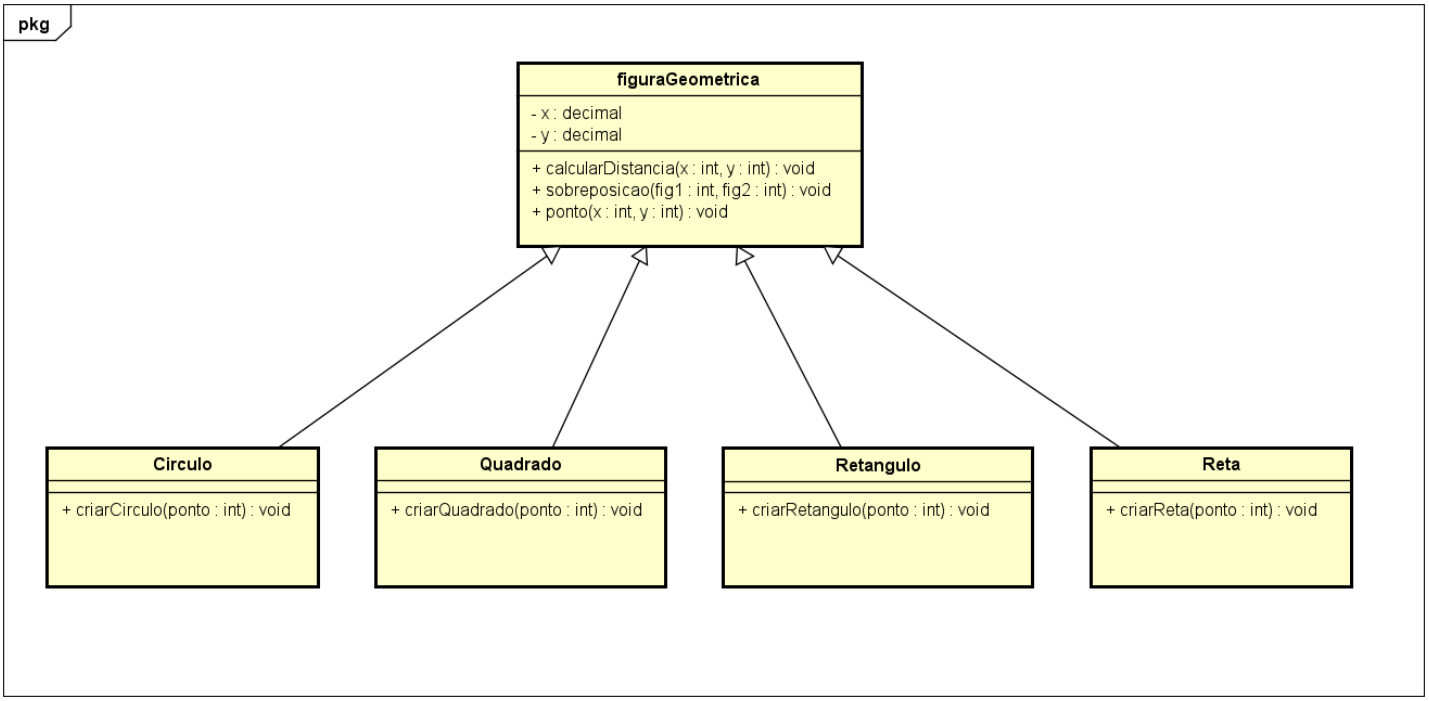
Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Curso de Engenharia de Software

Projeto: universo interativo para registrar formas geométricas bidimensionais

Assunto: Lançamento 01
Disciplina Obrigatória: Orientação a Objetos
Aluno (a): Felipe das Neves Freire
Matrícula: 202046102
E-mail: felipe.neves.freire@gmail.com
Período: 2/2023
Professor: Henrique Gomes de Moura
Data da Entrega: 15 de Setembro de 2023

Brasília, DF
2023

1. UML



2. PSEUDOCÓDIGO

Aqui criarei as classes com seus respectivos atributos e métodos:

// A classe figuraGeometrica é a classe mãe com os atributos genéricos x, y, bem como as funções

Classe FiguraGeometrica

```
    x: decimal
    y: decimal

função calcularDistância(ponto1, ponto2):
    deltaX <- ponto2.x - ponto1.x
    deltaY <- ponto2.y - ponto1.y
    retorno √(deltaX^2 + deltaY^2)


função criarPonto():
    x <- lerDecimal("Digite a coordenada x do ponto: ")
    y <- lerDecimal("Digite a coordenada y do ponto: ")
    ponto <- Ponto(x, y)
    retorno ponto

função verificarSobreposicao(figuras):
    para i de 0 até tamanho(figuras) - 1:
        para j de i + 1 até tamanho(figuras):
            se estãoSobrepostas(figuras[i], figuras[j]):
                exibir("As figuras", figuras[i].tipo, "e", figuras[j].tipo, "estão sobrepostas.")

função estãoSobrepostas(figura1, figura2):
    para cada ponto1 em figura1.pontos:
        para cada ponto2 em figura2.pontos:
            distância <- calcularDistância(ponto1, ponto2)
            se distância == 0:
                retorno verdadeiro
    retorno falso
```

// As classes seguintes herdam da classe figuraGeometrica, tendo somente os métodos específicos de cada classe

Classe Circulo

```
função criarCírculo(pontos):
    raio <- calcularDistância(pontos[1], pontos[2])
    área <- π * raio * raio
    perímetro <- 2 * π * raio
    exibir("Círculo criado com raio:", raio)
    exibir("Área do círculo:", área)
    exibir("Perímetro do círculo:", perímetro)
    retorno FiguraGeometrica("Círculo", pontos)
```

Classe Retangulo

```
função criarRetângulo(pontos):
    largura <- calcularDistância(pontos[1], pontos[2])
    altura <- calcularDistância(pontos[2], pontos[3])
    área <- largura * altura
    perímetro <- 2 * (largura + altura)
    exibir("Retângulo criado com largura:", largura, "e altura:", altura)
    exibir("Área do retângulo:", área)
    exibir("Perímetro do retângulo:", perímetro)
    retorno FiguraGeometrica("Retângulo", pontos)
```

Classe Quadrado

```
função criarQuadrado(pontos):
lado <- calcularDistância(pontos[1], pontos[2])
    área <- lado * lado
    perímetro <- 4 * lado
    exibir("Quadrado criado com lado:", lado)
    exibir("Área do quadrado:", área)
    exibir("Perímetro do quadrado:", perímetro)
    retorno FiguraGeometrica("Quadrado", pontos)
```

Classe Reta

```
função criarReta(pontos):
    exibir("Reta criada com pontos:")
    para ponto em pontos:
        exibir("(", ponto.x, ",", ponto.y, ")")
    retorno FiguraGeometrica("Reta", pontos)
```

// O main() irá executar meu código, nele coloquei também um menu interativo

main(){

```
   criar uma lista vazia de formasGeometricas

    enquanto verdadeiro:
        exibir("1. Criar um ponto")
        exibir("2. Criar uma reta")
        exibir("3. Criar uma figura geométrica")
        exibir("4. Verificar sobreposição de figuras")
        exibir("5. Sair")

        opção <- lerInteiro("Digite o número da opção desejada: ")

        se opção == 1:
            ponto <- criarPonto()
```

```
        exibir("Ponto criado em (", ponto.x, ",", ponto.y, ")")

    senão se opção == 2:
        exibir("Digite os pontos para criar a reta:")
        ponto1 <- criarPonto()
        ponto2 <- criarPonto()
        figuraGeométrica <- criarReta([ponto1, ponto2])
        adicionar figuraGeométrica à lista de formasGeometricas

    senão se opção == 3:
        se tamanho da lista de pontos < 2:
            exibir("É necessário pelo menos 2 pontos para criar uma figura.")
        senão:
            exibir("Escolha uma figura geométrica para criar:")
            exibir("1. Círculo")
            exibir("2. Retângulo")
            exibir("3. Quadrado")
            figura <- lerInteiro("Digite o número da figura desejada: ")

            se figura == 1:
                figuraGeométrica <- criarCírculo(pontos)

            senão se figura == 2:
                figuraGeométrica <- criarRetângulo(pontos)

            senão se figura == 3:
                figuraGeométrica <- criarQuadrado(pontos)

            senão:
                exibir("Opção inválida.")
            adicionar figuraGeométrica à lista de formasGeometricas

    senão se opção == 4:
        verificarSobreposicao(formasGeometricas)

    senão se opção == 5:
        exibir("Sair!")
        parar

    senão:
        exibir("Opção inválida.")


}
```

BIBLIOGRAFIA

Medeiros, A. V. M. ([s.d.]). Portugol Online. Github.io. Recuperado 16 de setembro de 2023, de https://vinyanalista.github.io/portugol/