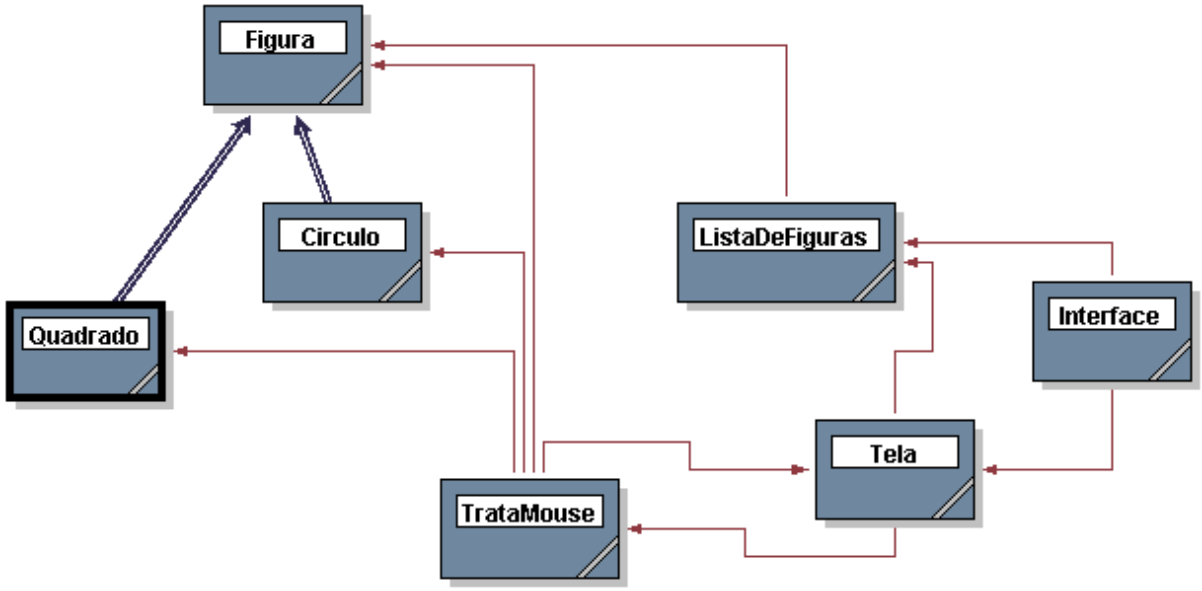
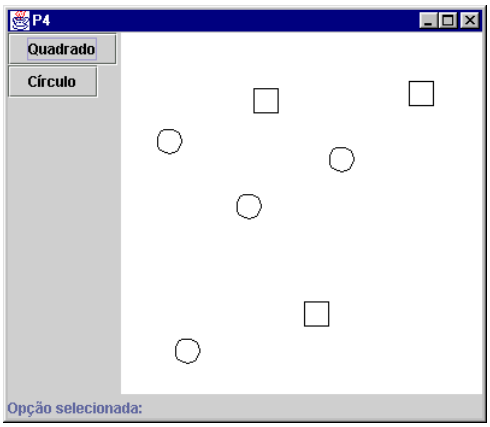
Nome legível do aluno(a): _____



O diagrama de classes acima gera um programa simples para desenho de figuras geométricas. É possível desenhar quadrados e círculos. Todos são armazenados em um vetor de figuras.A seguir, segue a descrição de cada classe:

- Figura: representa uma figura qualquer. Recebe um ponto (x,y) indicando a posição da figura. Possui métodos para setar uma cor de desenho, para obter seu tamanho e para desenhá-la, embora estes últimos não tenham função.
- Quadrado: representa um quadrado geométrico. Recebe um ponto (x,y) e o lado.
- Círculo: representa um círculo geométrico. Recebe um ponto (x,y) e o raio.
- ListaDeFiguras: armazena um vetor de figuras. Possui métodos para inicializar o vetor (com um tamanho máximo), inserir uma nova figura, procurar por uma figura através da posição do mouse e desenhar todas as figuras em uma tela qualquer.
- Tela: representa a tela gráfica, responsável pelo desenho das figuras.
- TrataMouse: responsável pelo tratamento dos eventos de mouse sobre a tela gráfica
- Interface: classe principal, cria a janela e monta a interface gráfica. Responsável pelo tratamento dos eventos de botão. A interface gráfica é apresentada na figura abaixo:



1. (3 pt) Complete os trechos de código abaixo, de forma a realizar a função proposta nos comentários:

```
class Figura implements Serializable
{
  protected int x,y;

  public Figura(int px,int py) {
    // Seta as coordenadas x,y da figura
    [          ]
  } ...
}

class Quadrado [          ] implements Serializable
{
  protected int lado;

  public Quadrado(int x, int y, int lado) {
    // seta coordenadas e o lado
   [          ]

  }
  public void desenha(Graphics g) { ... }
}

class Circulo [          ] implements Serializable
{
  protected int raio;

  public Circulo(int x1, int y1, int raio) {
    // seta coordenadas e o raio
    [          ]
  }
  ...
}

class ListaDeFiguras implements Serializable
{
  private Figura [          ] ;

  private int tmax;
  private int total;

  // Inicializa vetor de figuras com um tamanho máximo
  public ListaDeFiguras(int t) {
    [          ]
  }

  // Insere uma figura no vetor, se ainda tiver lugar
  public void insere( [          ] ) {
    [          ]
  }

  // Desenha todas as figuras armazenadas no vetor
  public void desenha(Graphics g) {
    [          ]
```

```
  }
}
```

2. (3 pt) Preencha as lacunas, com o código correto para criar a interface gráfica mostrada anteriormente:

```
class Interface extends JFrame implements [          ]
{
      ListaDeFiguras lista;

      JButton but,but2;
      JPanel painel;
      JPanel botoes;
      JLabel texto;
      Tela tela;

      public Interface()
      {
            super("P4");
            // Cria os botões
            [          ]

            // Adiciona um listener para eventos de botão
            [          ]

            texto = new JLabel("Opção selecionada:");

            // Cria a tela
            tela = new Tela(300,300,lista);

            // Cria o painel de botões
            botoes = new JPanel();
            botoes.setLayout(new BoxLayout(botoes,BoxLayout.Y_AXIS));
            // Preenche o painel
            [          ]

            // Cria o painel principal
            painel = new JPanel();
            painel.setLayout(new BorderLayout(3,3));
            // Preenche o painel
            [          ]

            // Adiciona ao JFrame
            getContentPane().add(painel);
            ...
      }
}

public class Tela extends JPanel
{
      ListaDeFiguras lista;
      TrataMouse mouse;

      public Tela(int w, int h, ListaDeFiguras lista)
      {
            super();
            setPreferredSize(new Dimension(w,h));
            mouse = new TrataMouse(this);
            // Ativa eventos de mouse nesta tela
            [          ]
```

```
            this.lista = lista;
    } ...
}
```

3. (2 pt) Considere o código apresentado abaixo, da classe TrataMouse:

```
public class TrataMouse extends MouseAdapter implements MouseMotionListener
{
      Tela tela;  // referência para a tela de desenho
      Figura sel; // figura selecionada ou null
      int op;     // operação: 0 - seleção, 1 - criar quadrado
                  // 2 - criar círculo

      public TrataMouse(Tela tela)
      {
            this.tela = tela; // seta a tela
            op = 0;     // operação = seleção
            sel = null; // ninguém selecionado
      }

      // Ativa a opção desejada (chamado por click nos botões)
      public void setOption(int op)
      {
            this.op = op;
      }

...
}
```

A partir deste código, implemente o método **`public void mousePressed(MouseEvent e)`**. Este método deverá verificar a opção selecionada: se esta for criação de quadrado ou círculo, deverá criar um novo objeto na posição do mouse e inseri-lo na lista, redesenhando a tela em seguida.

4. (2 pt) Observe que todos os objetos implementam Serializable: crie dois métodos na classe Interface, para salvar e recuperar a lista de objetos. Não é necessário reescrever toda a classe. A chamada dos métodos deve ser como segue:

   - `public ListaDeFiguras carregaLista(String nome)` – recebe um string como nome de arquivo e tenta recuperar a lista de figuras a partir dele. Se não conseguir por qualquer motivo, cria uma lista com 20 posições. Retorna a lista lida ou criada.

   - `public void salvaLista(String nome)` – recebe um string como nome de arquivo e tenta gravar a lista de figuras nele.

Obviamente, ambos os métodos devem tratar as exceções necessárias.