Trabalho de Preenchimento

Aluno: Rosialdo Vicente

N° de Matricula: 2020018122

Objetivo do Programa

Desenvolver um programa que permita desenhar Preencher utilizando os algoritmos:

- Floodfill
- Varredura com Análise Geométrica

Tecnologias Utilizadas

Linguagem de Programação

Para a implementação deste trabalho, optei pela linguagem `Python` devido à minha familiaridade e à sua flexibilidade para desenvolvimento de aplicações gráficas.

Bibliotecas Utilizadas

As seguintes bibliotecas foram utilizadas no projeto:

- `sys`: Utilizada para encerrar o programa de forma adequada.
- `pygame`: Como eu já tinha usado para realizar os outros trabalhos eu escolhi ela para facilitar a implementação do

Implementação dos Algoritmos

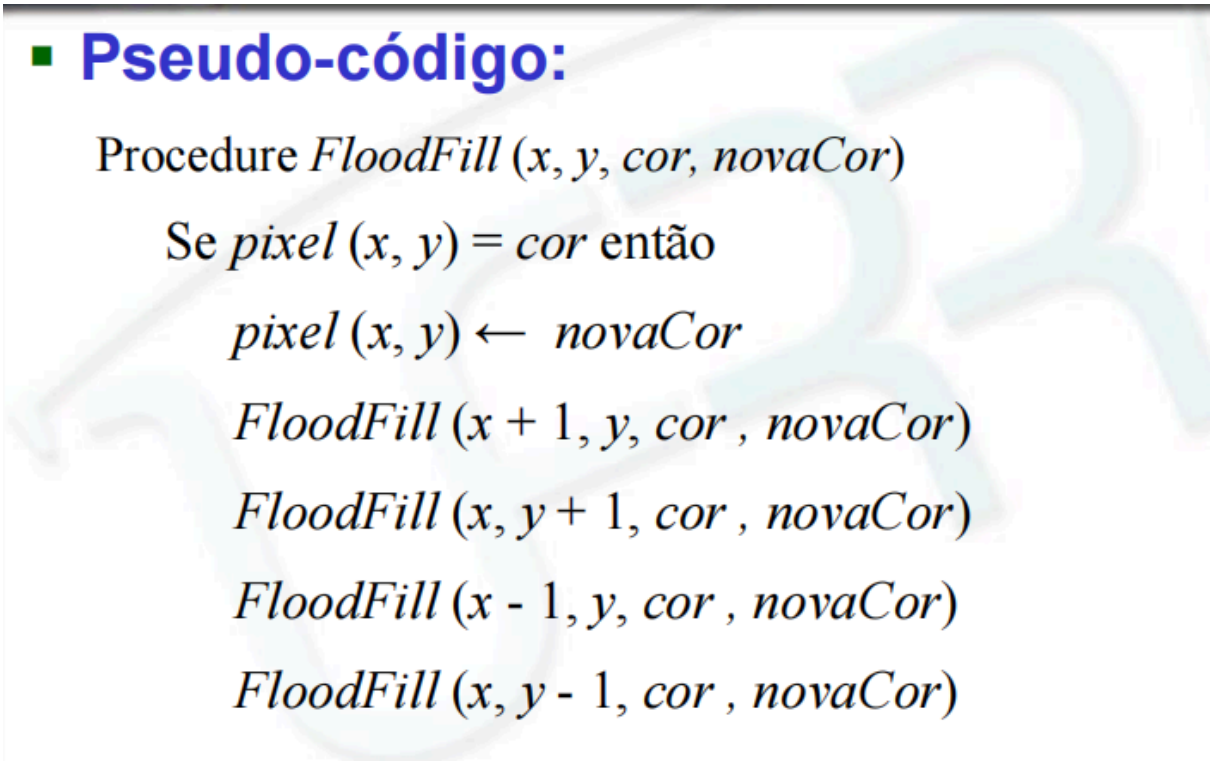
Foi implementado apenas o Floodfill, na circunferência e no retângulo, enfrentei algumas dificuldades pra implementar o algoritmo de varredura com Análise Geométrica e usar o Floodfill para as outras formas geométricas, então a implementação foi nesses dois exemplos.

Algoritmo Floodfill

O **algoritmo Flood Fill** é um método de preenchimento de regiões baseado na propagação a partir de um **pixel semente**. Ele examina os pixels vizinhos e preenche aqueles que atendem a um critério específico, como ter uma cor semelhante à da semente.

O algoritmo pode ser implementado de forma **recursiva**, onde cada pixel válido se torna uma nova semente, expandindo o preenchimento até que não haja mais vizinhos que satisfaçam o critério.

Referência para o Algoritmo Floodfill:



Desenvolvimento

Durante o desenvolvimento, enfrentei dificuldades ao lidar com a recursão. O código frequentemente apresentava estouro de memória, e levei um tempo considerável para solucionar esse problema.

A solução encontrada consistiu em aumentar os recursos disponíveis para o Python durante a execução do algoritmo e reduzir o tamanho das formas processadas. Após essa modificação, o algoritmo passou a realizar o preenchimento corretamente.

Para a construção da circunferência, reutilizei o algoritmo de Bresenham para rasterização de circunferências. Já para o triângulo, utilizei uma função nativa

do Python.

Gostaria de ter implementado um recurso dentro do Pygame que permitisse alternar entre diferentes formas diretamente na interface, pois, no momento, a alteração só pode ser feita manualmente no código. Apesar de não ter conseguido implementar todos os algoritmos e formas solicitados, acredito que a implementação do **Flood Fill** exemplifica satisfatoriamente o funcionamento do algoritmo.

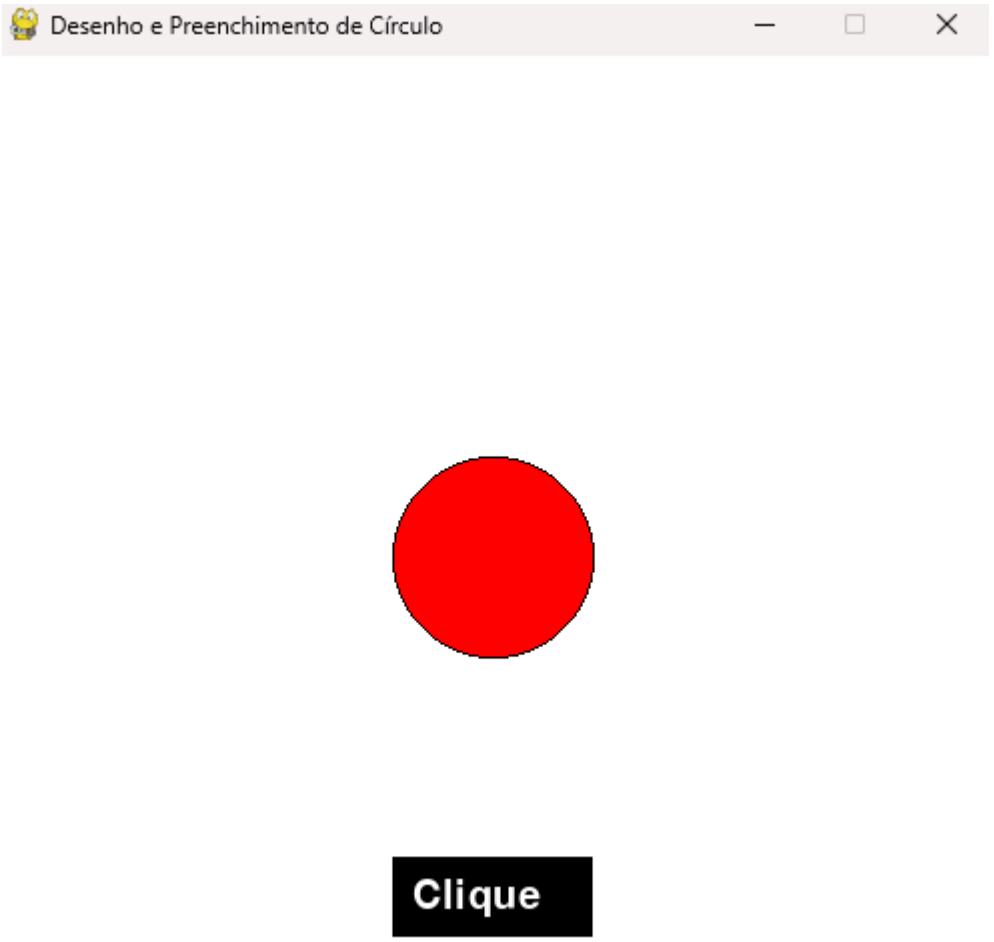
Testes

Todos os testes foram feitos com os seguintes valores:

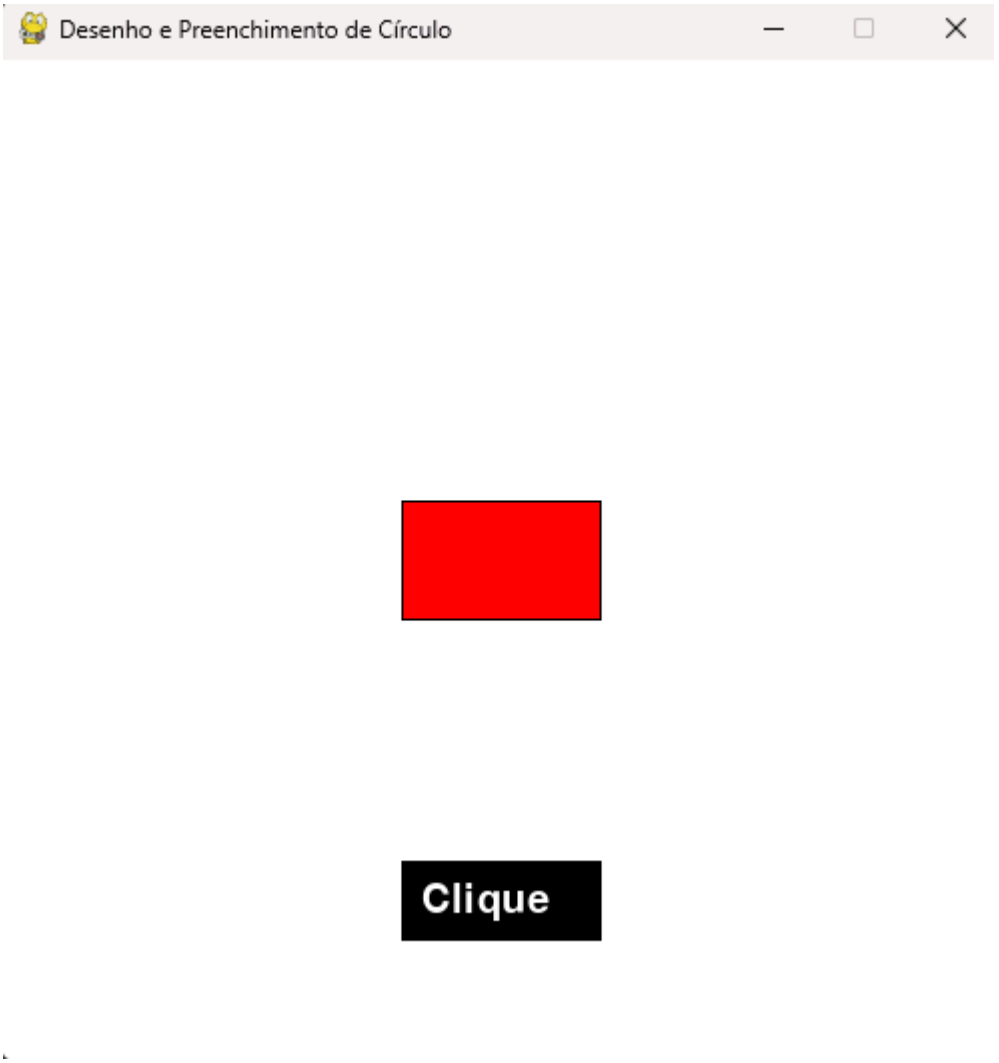
`Circunferencia: (screen, xc, yc, 50, BLACK)`

`Retangulo: (screen, xc - 50, yc - 30, 100, 60, BLACK)`

Teste Circunferência



Teste Retangulo



Explicação do Algoritmo

O algoritmo **Flood Fill** é uma técnica eficiente para preenchimento de áreas em computação gráfica, sendo amplamente utilizado em editores de imagens e jogos. Um dos seus principais pontos fortes é a **simplicidade de implementação**, especialmente na forma recursiva, que permite propagar o preenchimento a partir de um **pixel semente**, expandindo-se para os pixels vizinhos de acordo com um critério específico. Além disso, ele pode ser adaptado para diferentes estratégias de vizinhança (4 ou 8 direções) e para diferentes condições de preenchimento, tornando-o versátil em diversas aplicações.

No entanto, o algoritmo apresenta **limitações significativas**. Sua implementação recursiva pode levar a **estouro de pilha**, especialmente para áreas extensas, devido ao grande número de chamadas recursivas aninhadas.

Além disso, a eficiência do Flood Fill pode ser comprometida quando aplicado a imagens de alta resolução, pois a abordagem tradicional não utiliza estruturas de dados otimizadas para evitar o processamento redundante de pixels já visitados.