



# Instalação:

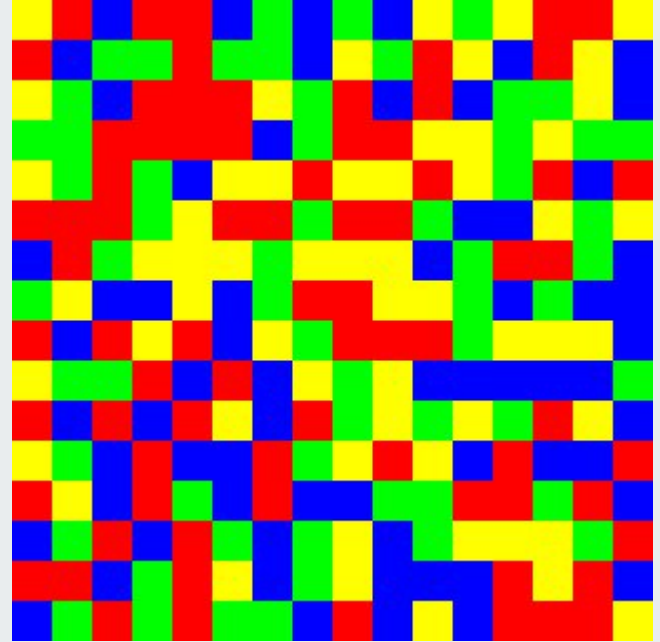
1. Pressionar: ícone do windows + R
2. Digitar `cmd`
3. **ENTER**
4. `pip install -U pygame --user`

<https://www.pygame.org/docs/>



# Flood fill

<https://unixpapa.com/floodit/?sz=6&nc=4>





## Regras:

- O objetivo é completar a matriz com a mesma cor em um número máximo de passos;
- Cada rodada o jogador tem direito de mudar a cor APENAS do primeiro elemento da matriz: (o primeiro quadrado) ;
- Caso os “vizinhos” sejam da mesma cor (a cor anterior, não a nova), deve-se mudar também a cor deles para a nova cor; (como infecção viral)
- Os vizinhos estão nas quatro direções: norte, sul, leste e oeste.

## Relembrando 1:

```
import pygame
import random
#inicia o pygame
pygame.init()
#cria as cores
white = (255,255,255)
black = (0,0,0)
red = (255,0,0)
green = (0,255,0)
blue = (0,0,255)
#cria a tela
screen = pygame.display.set_mode((640,480))
pygame.display.set_caption("flood")
#finaliza o jogo
pygame.quit()
```

## Relembrando 2:

- Devemos colocar o jogo em um laço de repetição para que ele permaneça em um loop até que o usuário deseje sair.
- O laço é controlado por uma variável **True** ou **False**.

. . .

```
sair = True
while sair:
    pygame.quit()
```

- Agora precisamos “capturar” os eventos no for.
- O jogo ficará congelado esperando um evento.
- Caso ocorra um evento, tudo que estiver dentro do **for** será executado e a tela será atualizada.

. . .

```
sair = True
while sair:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sair = False

pygame.quit()
```

# Criar a matriz

- Vamos criar uma abstração onde cada cor vai ser na verdade um número “por trás das cortinas”.
- Iniciamos a matriz com zeros.
- As cores ficarão em um vetor de cores, para que, quando eu quiser acessar, basta escolher um índice do vetor.

```
. . .  
#no início do jogo  
screen.fill(white)  
cores = [red,green,blue,black]  
matriz = [[0 for x in range(7)] for y in range(7)]  
  
. . .
```

# Matriz aleatória



- Vamos preencher a matriz com números aleatórios. Os números poderão ser: 0, 1, 2 ou 3, pois são as posições do vetor de cores.
- Lembrando que essa parte é **fora do while** pois a matriz só será preenchida aleatoriamente uma vez antes do jogo começar de fato.

```
. . .  
for i in range(0,6):  
    for j in range(0,6):  
        matriz[i][j] = random.randint(0,3)  
. . .
```

# Desenhar a matriz

- Cada elemento da matriz será um quadrado da matriz!
- Por isso precisamos criar um quadrado para cada elemento dentro de dois **for**, que é a forma como acessamos os elementos da matriz.
- Agora chamamos a função `pygame.draw.rect()` para desenhar o elemento.

```
. . .
sair = True
while sair:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sair = False
    for i in range(0,9):
        for j in range(0,9):
            pygame.draw.rect(screen,colors[matriz[i][j]],(i*50,j*50,50,50),
0)
            pygame.display.update()
pygame.quit()
```



# Mudando a cor

- Para mudar a cor vamos fazer isso dentro de uma função.
- A função vai se chamar **flood()** e vai ter os seguintes parâmetros: a cor alvo, a matriz, as posições do elemento da matriz (i,j) e a nova cor que deve ser pintada.
- “Pintar” significa atribuir a nova cor (que na verdade é um número) à matriz no elemento que se encontra.
- Se a cor alvo for igual a nova cor, não é necessário pintar.
- O **elif** garante que estou numa posição válida na matriz.

```
def flood(cor_alvo,matriz,i,j,nova_cor):  
    if(matriz[i][j]!=cor_alvo or cor_alvo == nova_cor):  
        return  
    elif(0<=i<5 and 0<=j<5):  
        matriz[i][j] = nova_cor
```

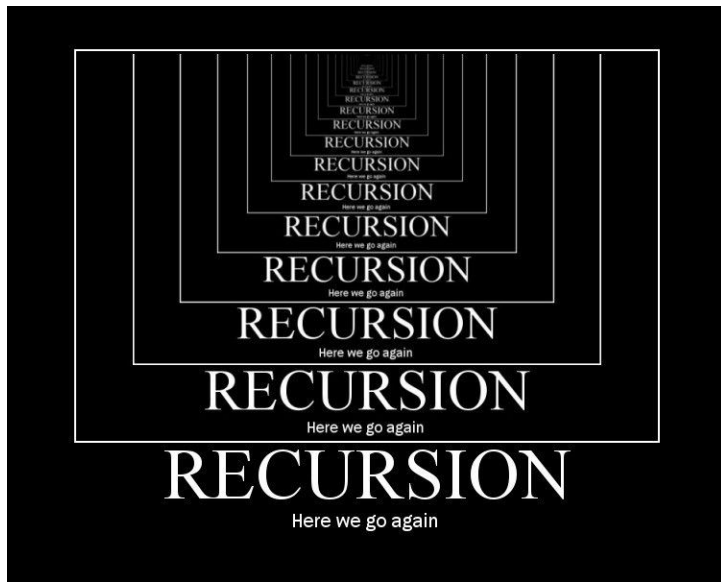
# Finalizando...


- Agora só precisamos chamar a função dentro do loop do jogo para que ela floode a matriz.
- Vamos usar a entrada do teclado para associar cada cor a um caractere do teclado.

```
. . .
    keys=pygame.key.get_pressed()
    if keys[pygame.K_0]:
        flood(matriz[0][0],matriz,0,0,0)
    if keys[pygame.K_1]:
        flood(matriz[0][0],matriz,0,0,1)
. . .
```

# Recursão

- Estamos pintando só o primeiro quadrado, mas ele não consegue pintar seus vizinhos.
- Para resolver esse problema vamos usar a ideia de recursão: a função **flood()** vai chamar ela mesma, mas com parâmetros diferentes.
- A função vai chamar ela mesma até que uma condição não seja mais verdadeira.





```
def flood(cor_alvo,matriz,i,j,nova_cor):  
    if(matriz[i][j]!=cor_alvo or cor_alvo == nova_cor):  
        return  
    elif(0<=i<6 and 0<=j<6):  
        matriz[i][j] = nova_cor  
        flood(cor_alvo,matriz,i+1,j,nova_cor)    #para direita  
        flood(cor_alvo,matriz,i,j+1,nova_cor)    #para baixo  
        flood(cor_alvo,matriz,i-1,j,nova_cor)    #para esquerda  
        flood(cor_alvo,matriz,i,j-1,nova_cor)    #para cima
```

# Mouse

- Agora vamos mudar um pouco o código. Vamos pegar a cor sem precisar usar o teclado e sim o mouse.
- Vamos pegar a posição x e y do mouse e consultar qual cor está pintada naquela posição.

```
. . .  
    if (evento.type == pygame.MOUSEBUTTONDOWN):  
        x, y = pygame.mouse.get_pos()  
        flood(matriz[0][0],matriz,0,0,matriz[x//50][y//50])  
  
. . .
```

# Texto


- Para criar texto no pygame existe o módulo font com funções apropriadas.
- Vamos criar um texto que nos informa quantas jogadas já realizamos.
- Primeiro criamos a variável jogada = 0,
- Definimos o estilo da fonte,
- E criamos o texto.

<https://www.pygame.org/docs/ref/font.html#pygame.font.Font.render>

. . .

```
jogada = 0
fonte = pygame.font.SysFont( "Comic Sams MS", 30)
texto = fonte.render( "Jogada "+str(jogada), False, (0,0,0))
```

. . .

- 
- Agora dentro do loop podemos mostrar o texto logo abaixo da matriz.
  - Mas é necessário atualizar a variável texto a cada jogada, além de incrementar a jogada.
  - A Função **blit()** coloca o texto na tela na posição informada.

```
. . .  
  
if (evento.type == pygame.MOUSEBUTTONDOWN):  
    x, y = pygame.mouse.get_pos()  
    flood(matriz[0][0],matriz,0,0,matriz[x//50][y//50])  
    jogada = jogada+1  
    texto = fonte.render( "Jogada "+str(jogada), False, (0,0,0))  
  
screen.blit(texto, (0,310))  
  
. . .
```