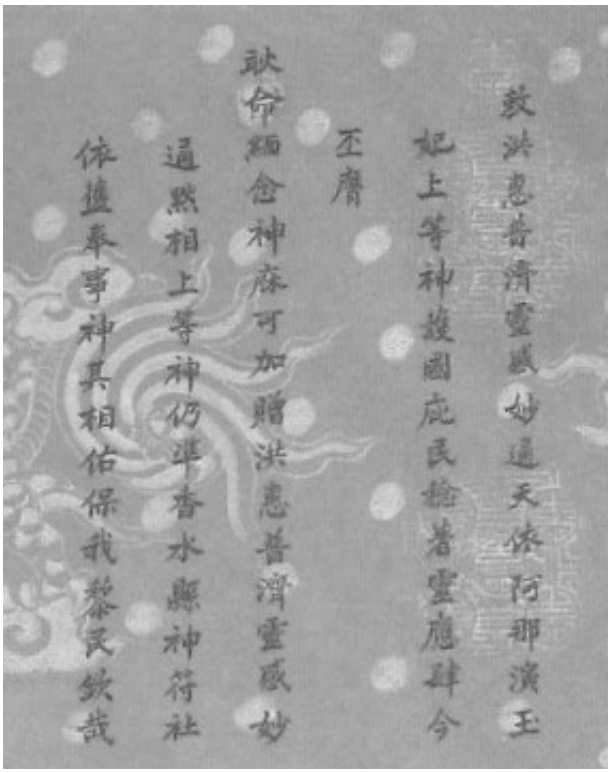
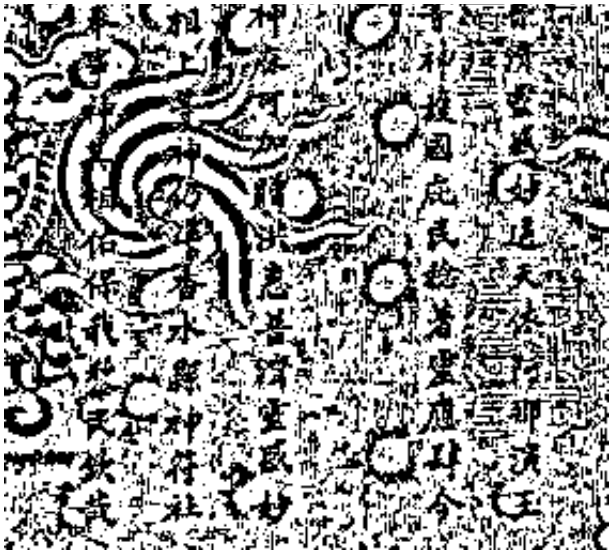
In [17]:
```
import matplotlib.pyplot as plt
import cv2
from PIL import Image
from IPython.display import display
```

Phân ngưỡng động

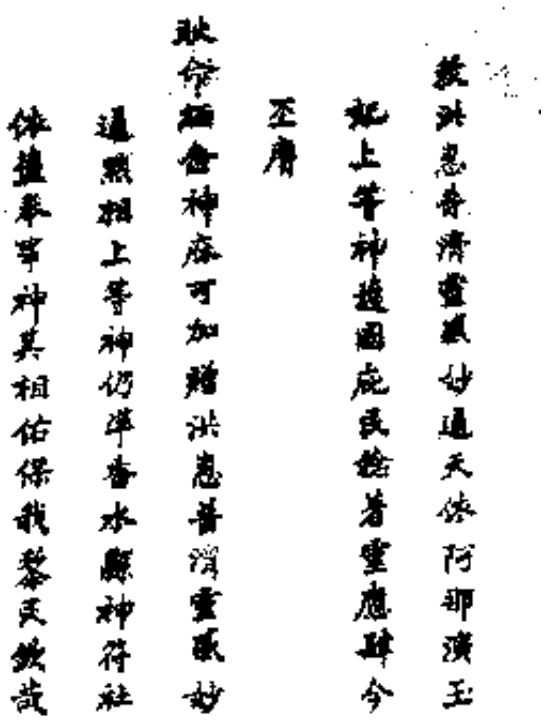
In [23]:
```
image = cv2.imread('./data_process/a2.jpg', cv2.IMREAD_COLOR)

# Chuyển ảnh sang ảnh xám
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Áp dụng phân ngưỡng động
thresh = cv2.adaptiveThreshold(gray_image, 255, cv2.ADAPTIVE_THRESH

image_show = Image.fromarray(gray_image)
display(image_show)

image_show = Image.fromarray(thresh)
display(image_show)
```





Phân ngưỡng bình thường

In [15]:
```
import cv2

image = cv2.imread('./data_process/a2.jpg', cv2.IMREAD_COLOR)

# Chuyển ảnh sang ảnh xám
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Áp dụng phân ngưỡng
_, thresh = cv2.threshold(gray_image, 0, 255, cv2.THRESH_BINARY + c

image_show = Image.fromarray(thresh)
display(image_show)
```

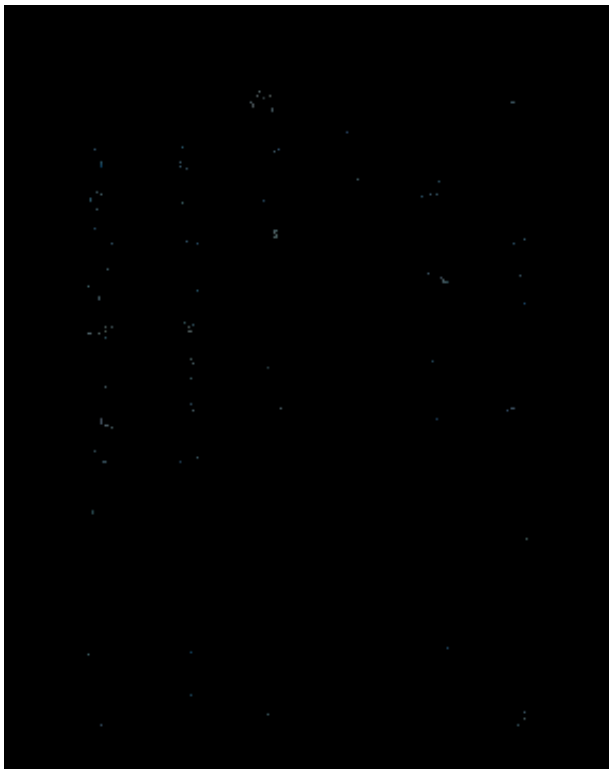


Dựa trên màu nền (color based)

In [14]:
```
import cv2
import numpy as np

image = cv2.imread('./data_process/a2.jpg', cv2.IMREAD_COLOR)

# Màu nền (background)
background_color = [234,182,118]

# Tạo một mask để chọn vùng cần giữ lại
mask = np.all(image != background_color, axis=-1)

# Tạo ảnh mới với nền loại bỏ
result = np.zeros_like(image)
result[mask] = image[mask]

image_show = Image.fromarray(result)
display(image_show)
```

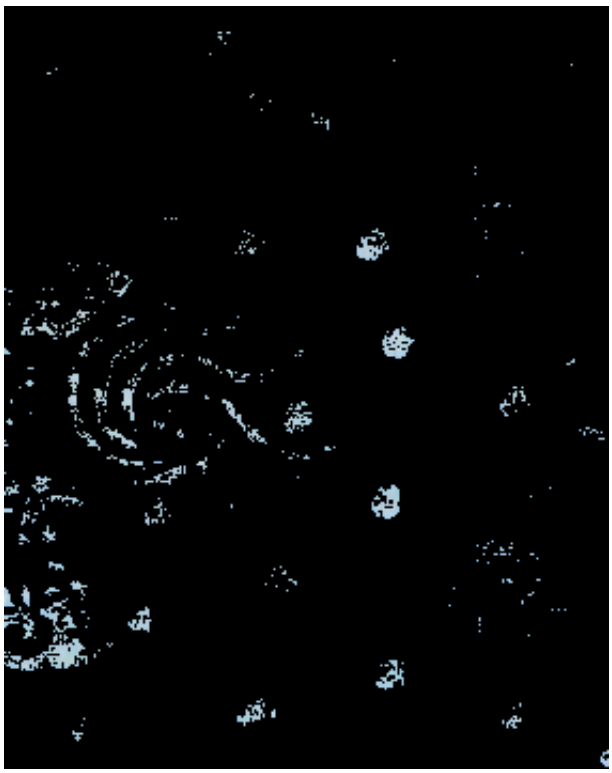
Sử dụng phân ngưỡng màu sắc (Color Thresholding):

In [13]:
```
import cv2
import numpy as np

image = cv2.imread('./data_process/a2.jpg')

# Chuyển đổi sang không gian màu HSV
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

# Thiết lập ngưỡng màu để tách nền
lower_color = np.array([0, 0, 0])
upper_color = np.array([179, 255, 100])

# Tạo mask dựa trên ngưỡng màu
mask = cv2.inRange(hsv, lower_color, upper_color)

# Áp dụng mask để loại bỏ nền
result = cv2.bitwise_and(image, image, mask=mask)

image_show = Image.fromarray(result)
display(image_show)
```



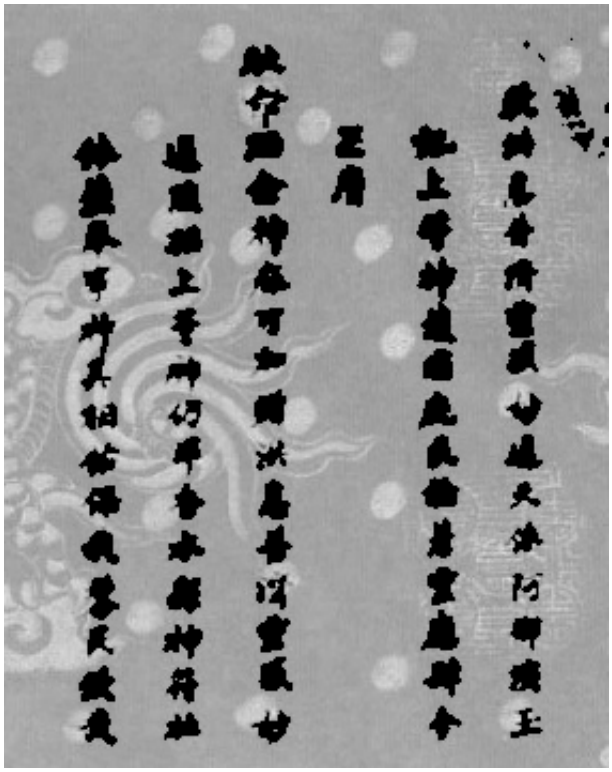
Sử dụng phân ngưỡng độ sáng (Brightness Thresholding):

In [12]:
```
import cv2

image = cv2.imread('./data_process/a2.jpg', cv2.IMREAD_COLOR)

# Chuyển đổi sang ảnh xám
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Áp dụng phân ngưỡng độ sáng để tách nền
_, thresh = cv2.threshold(gray_image, 200, 255, cv2.THRESH_BINARY)

# Lọc ảnh để giữ lại văn bản và loại bỏ nền
result = cv2.bitwise_and(image, image, mask=thresh)

image_show = Image.fromarray(result)
display(image_show)
```



Phân ngưỡng dựa trên kết cấu (Texture-based Thresholding) - Sử dụng Gaussian Blur

In [18]:
```
import cv2


image = cv2.imread('./data_process/a2.jpg', cv2.IMREAD_GRAYSCALE)

# Áp dụng Gaussian Blur để làm mờ ảnh
blurred = cv2.GaussianBlur(image, (5, 5), 0)

# Áp dụng phân ngưỡng độ sáng
_, thresh = cv2.threshold(blurred, 0, 255, cv2.THRESH_BINARY + cv2.

# Lọc ảnh để giữ lại văn bản và loại bỏ nền
result = cv2.bitwise_and(image, image, mask=thresh)

image_show = Image.fromarray(result)
display(image_show)
```



Áp dụng các phép biến đổi hình thái học, kết hợp mở rộng (dilation) và thu nhỏ (erosion), để loại bỏ nhiễu và tinh chỉnh mask sau khi áp dụng phân ngưỡng

In [20]:
```
import cv2
import numpy as np


image = cv2.imread('./data_process/a2.jpg', cv2.IMREAD_GRAYSCALE)

# Áp dụng phân ngưỡng
_, thresh = cv2.threshold(image, 200, 255, cv2.THRESH_BINARY)

# Sử dụng kernel cho phép biến đổi hình thái học
kernel = np.ones((5, 5), np.uint8)

# Áp dụng mở rộng (dilation) để loại bỏ nhiễu và nối các phần văn b
dilated = cv2.dilate(thresh, kernel, iterations=1)

# Áp dụng thu nhỏ (erosion) để tinh chỉnh mask
eroded = cv2.erode(dilated, kernel, iterations=1)

# Lọc ảnh để giữ lại văn bản và loại bỏ nền
result = cv2.bitwise_and(image, image, mask=eroded)

image_show = Image.fromarray(result)
display(image_show)
```