BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN – ĐHQG HCM
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN THỰC HÀNH CUỐI KỲ
TOÁN ỨNG DỤNG VÀ THỐNG KÊ

Giảng viên hướng dẫn: *Nguyễn Hữu Toàn*
Võ Nam Thục Đoan
Lớp: *21_2*
Sinh viên thực hiện: *21120050 – Trương Tấn Đạt*

HỌC KỲ II
NĂM HỌC 2022 – 2023

ĐỀ BÀI: *Ứng dụng của đại số tuyến tính trong bài toán minh họa các phép biến hình trong R^2*

I. Giới thiệu:

- Ứng dụng của đại số tuyến tính rất phong phú và đa dạng trong nhiều lĩnh vực khác nhau, và trong trường hợp này chúng ta có thể đánh giá ứng dụng của nó qua bài toán minh họa các phép biến hình trong không gian hai chiều R^2.
- Đại số tuyến tính cung cấp một cách hiệu quả để mô hình hóa và giải quyết các bài toán liên quan đến các phép biến hình trong không gian hai chiều. Dưới đây là một số phép biến hình được giải quyết bằng cách ứng dụng đại số tuyến tính:
  - ***Phép phản chiếu (đối xứng)***: Là phép biến hình lật hình học qua một đường thẳng hay một điểm.
  - ***Phép xoay*:** Là phép biến hình quay một hình học xung quanh một trục xoay (ở đây xem hình ảnh gốc là trục).
  - ***Phép co giãn:*** Là phép biến hình thay đổi kích thước của hình học bằng cách tăng hoặc giảm tỷ lệ của các độ dài.
  - ***Phép trượt nghiêng:*** Là phép biến hình thay đổi vị trí của các điểm trong không gian theo hướng ngang hoặc dọc bằng cách tăng hoặc giảm tỷ lệ của các độ dài.
- Nhờ vào đại số tuyến tính, chúng ta có thể bieru diễn và tính toán các phép biến hình trong R^2 một cách rõ ràng và hiệu quả. Điều này giúp chúng ta hiểu sâu hơn về cách các phép biến hình ảnh hưởng đến hình dạng và vị trí của các đối tượng trong không gian hai chiều. Phần tiếp theo đây, em sẽ minh họa kết quả các phép biến hình trong không gian R^2 (file mã nguồn được đính kèm với báo cáo).

II. Thư viện và các hàm hỗ trợ:

1. *Thư viện hỗ trợ:*

Đồ án sử dụng hai thư viện chính là ***matplotlib*** để phục vụ cho việc vẽ hình và ***math*** để thực hiện tính toán cos và sin của góc θ.

2. *Hàm* $\boldsymbol{mul\_matrix(A, B)}$*:*

Thực hiện nhân hai ma trận là ma trận A và ma trận B với nhau.

3. *Hàm* $\boldsymbol{illustrate(original, standard, color)}$*:*

- Nhận vào ba đối số là ma trận gốc `original`, ma trận chuẩn `standard` và màu `color` để vẽ.

- Đầu tiên, ta tạo ma trận điểm đơn vị `unit_points_matix`. Ma trận này có hai hàng, mỗi hàng chứa hai giá trị tương ứng với tọa độ `x` và `y` của một điểm đơn vị. Điểm đơn vị là một hình vuông có cạnh độ dài là 1.

- Sau đó, chúng ta tính toán các điểm biểu diễn từ ma trận gốc và ma trận điểm đơn vị. Điều này được thực hiện bằng cách nhân ma trận gốc với ma trận điểm đơn vị bằng hàm `mul_matrix(original, unit_points_matrix)`. Kết quả là ma trận `original_points` chứa các tọa độ `x` và `y` của các điểm biểu diễn.

- Tiếp theo, chúng ta trích xuất các tọa độ `x` và `y` từ `original_points` để vẽ. Đầu tiên, chúng ta khởi tạo hai danh sách rỗng `x_original` và `y_original`. Sau đó, chúng ta lặp qua các cột của `original_points` và thêm giá trị `x` vào `x_original` và giá trị `y` vào `y_original`.

- Sau đó, chúng ta lặp lại điểm đầu tiên bằng cách thêm giá trị `x` và `y` của điểm đầu tiên vào cuối `x_original` và `y_original`. Điều này để vẽ một đường thẳng đóng.

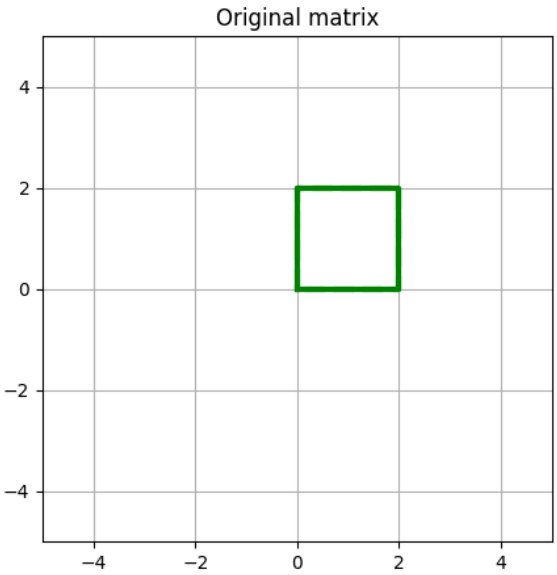
- Tiếp theo, chúng ta tính toán các điểm biểu diễn từ ma trận chuẩn và ma trận điểm đơn vị bằng cách nhân hai ma trận này với nhau bằng hàm `mul_matrix(standard, unit_points_matrix)`. Kết quả là ma trận `transformed_points_matrix` chứa các tọa độ `x` và `y` của các điểm biểu diễn đã được chuyển đổi.

- Sau đó, chúng ta trích xuất các tọa độ `x` và `y` từ `transformed_points_matrix` tương tự như trên và lặp lại điểm đầu tiên để vẽ đường thẳng đóng.

- Cuối cùng, chúng ta sử dụng thư viện `matplotlib.pyplot` để tạo một hình vẽ. Chúng ta tạo một đối tượng hình ảnh với kích thước 5x5 bằng `plt.figure(figsize=(5, 5))`. Chúng ta sử dụng `plt.plot` để vẽ đường thẳng màu xanh `('g--')` từ các điểm trong `x_original` và `y_original`, và vẽ đường thẳng từ các điểm trong `x` và `y` với màu sắc được chỉ định bởi đối số `color`. Chúng ta sử dụng `plt.axis` để thiết lập giới hạn trục `x` và `y` của hình vẽ. Cuối cùng, chúng ta sử dụng `plt.grid(True)` để hiển thị lưới trên hình vẽ.

III. Minh họa các phép biến hình:

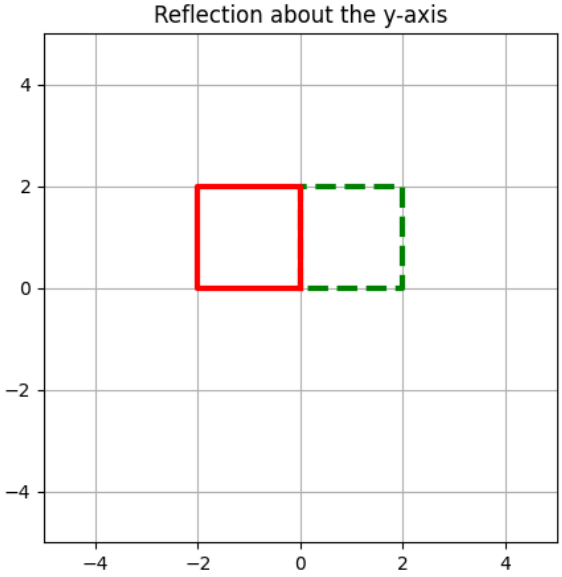
Trong bài làm, sử dụng ma trận $A = [[2, 0], [0, 2]]$ để minh họa các phép biến hình trên hệ trục tọa độ.



Hình 1: Hình ban đầu

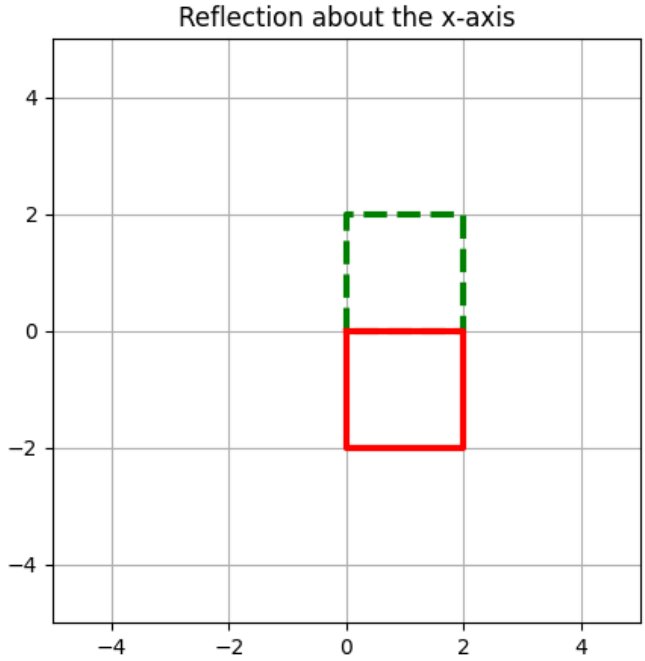
Ma trận ban đầu sẽ được minh họa bằng nét đứt, ma trận qua phép biến hình sẽ được minh họa bằng nét liền (khác màu). Trong trường hợp hai ma trận có những đoạn thẳng đè lên nhau thì chỉ thấy được những đoạn thẳng có màu của ma trận kết quả.

1. *Phép phản chiếu qua trục y –* $\boldsymbol{reflect\_y(matrix)}$:
   Sử dụng ma trận chuẩn là [[-2, 0], [0, 2]]



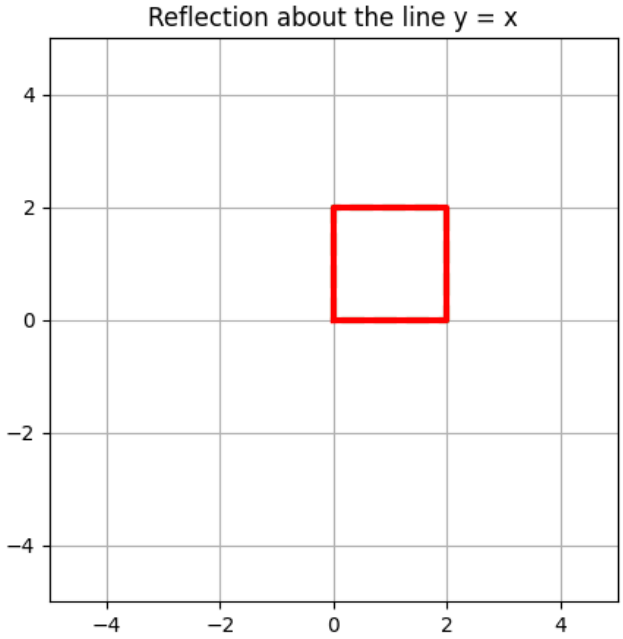
Hình 2: Hình minh họa phép phản chiếu qua trục Y

2. *Phép phản chiếu qua trục x –* $\boldsymbol{reflect\_x(matrix)}$*:*
   Sử dụng ma trận chuẩn là [[2, 0], [0, -2]]



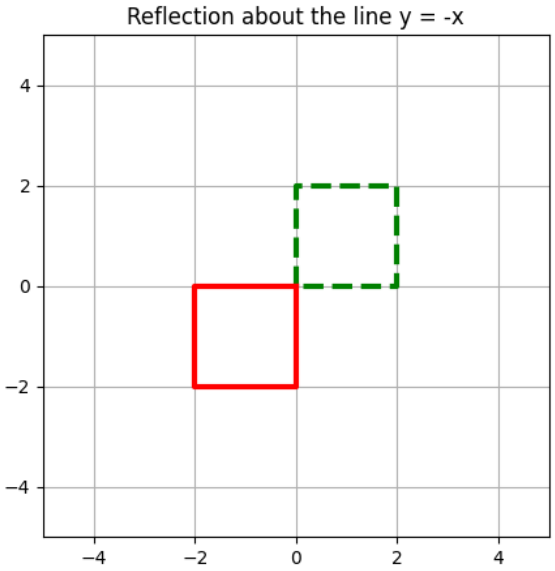
Hình 3: Hình minh họa phép phản chiếu qua trục X

3. *Phép phản chiếu qua đường thẳng* $y = x$ *–* $\boldsymbol{reflect\_y\_x(matrix)}$*:*
   Sử dụng ma trận chuẩn là [[2, 0], [0, 2]]



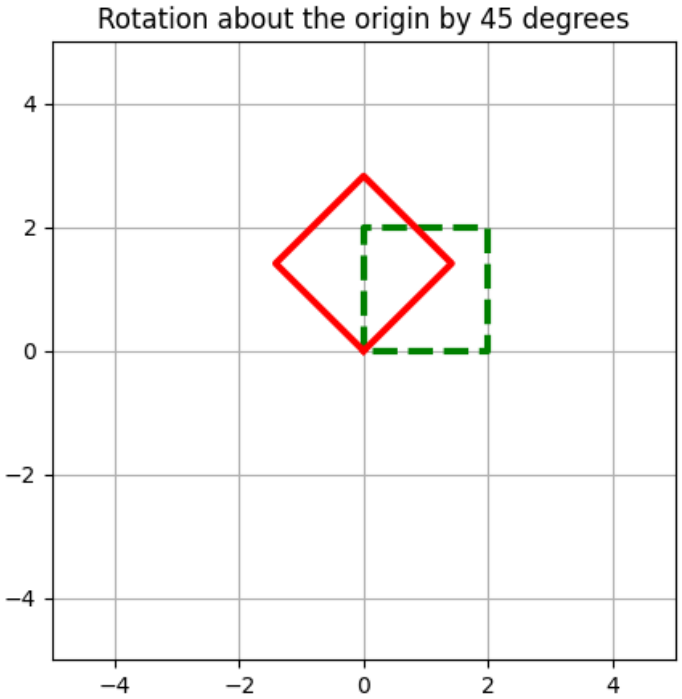
Hình 4: Hình minh họa phép phản chiếu qua đường thẳng $y = x$

4. *Phép phản chiếu qua đường thẳng* $y = -x$ – $\boldsymbol{reflect\_y\_neg\_x(matrix)}$*:*
   Sử dụng ma trận chuẩn là [[-2, 0], [0, -2]]



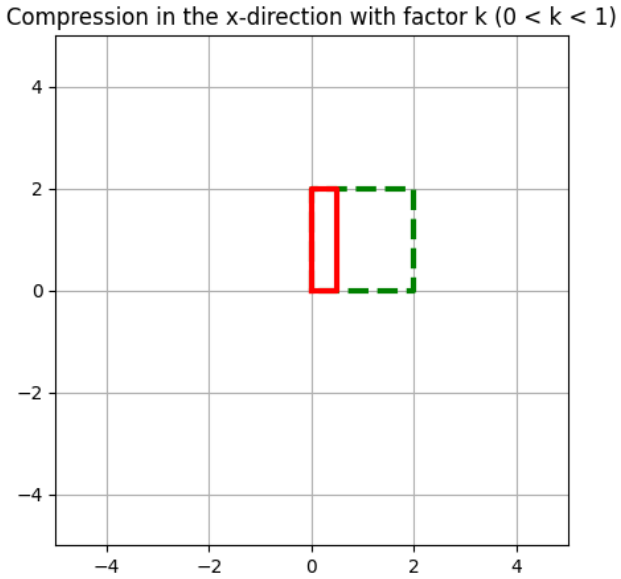
Hình 5: Hình minh họa phép phản chiếu qua đường thẳng $y = -x$

5. *Phép xoay quanh hình ảnh gốc theo góc dương* θ – $\boldsymbol{rotate(matrix, theta)}$*:*
   Sử dụng ma trận chuẩn là [[2cos(θ), -2sin(θ)], [2sin(θ), 2cos(θ)]] (với θ được tính theo đơn vị *radians*)
   Giả sử $\theta = 45°$



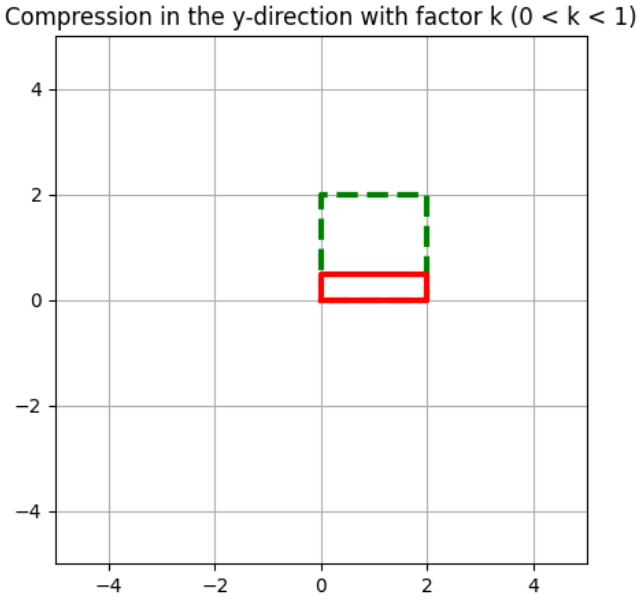
Hình 6: Hình minh họa phép xoay quanh hình ảnh gốc theo góc dương θ

6. *Phép nén theo hướng trục x với hệ số k* $(0 < k < 1)$ – ***compress_x_direction(matrix, k)***:
   Sử dụng ma trận chuẩn là [[k, 0], [0, 2]]
   Giả sử $k = 0.5$



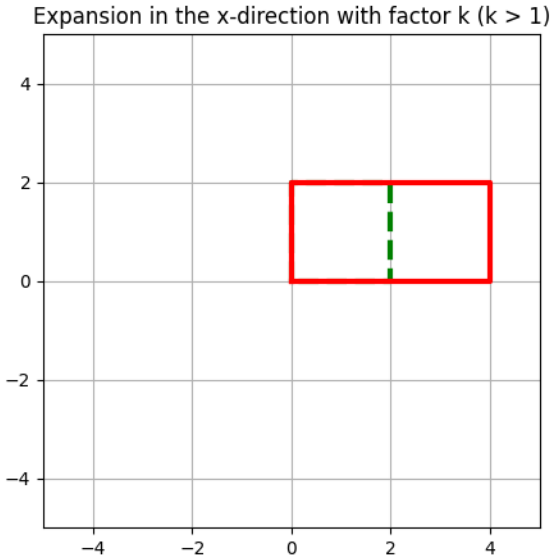
Hình 7: Hình minh họa phép nén theo hướng trục x với hệ số k

7. *Phép nén theo hướng trục y với hệ số k* $(0 < k < 1)$ – ***compress_y_direction(matrix, k)***:
   Sử dụng ma trận chuẩn là [[2, 0], [0, k]]
   Giả sử $k = 0.5$



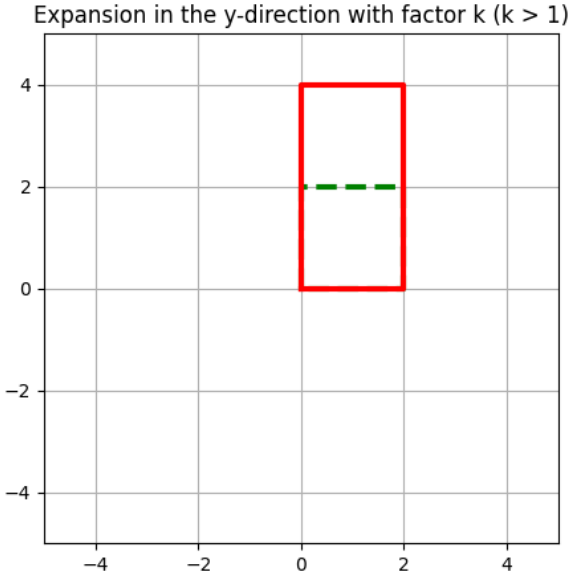
Hình 8: Hình minh họa phép nén theo hướng trục y với hệ số k

8. *Phép mở rộng theo hướng trục x với hệ số k* $(k > 1)$ – ***expansion_x_direction*(*matrix*, *k*)**:
   Sử dụng ma trận chuẩn là [[k, 0], [0, 2]]
   Giả sử $k = 4$



Hình 9: Hình minh họa phép mở rộng theo hướng trục x với hệ số k

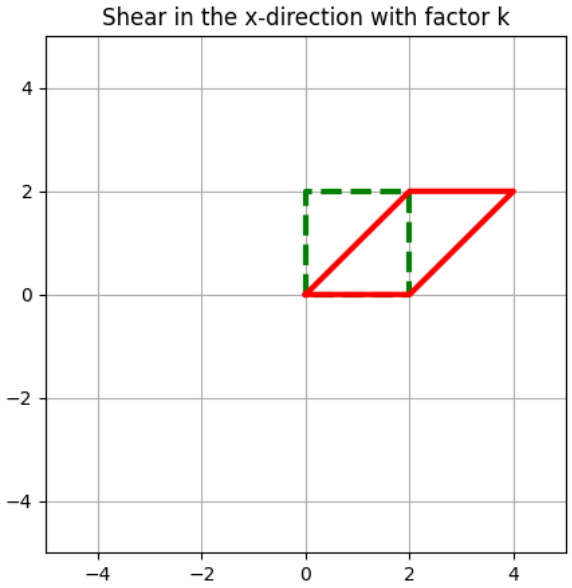
9. *Phép mở rộng theo hướng trục y với hệ số k* $(k > 1)$ – ***expansion_y_direction*(*matrix*, *k*)**:
   Sử dụng ma trận chuẩn là [[2, 0], [0, k]]
   Giả sử $k = 4$



Hình 10: Hình minh họa phép mở rộng theo hướng trục y với hệ số k

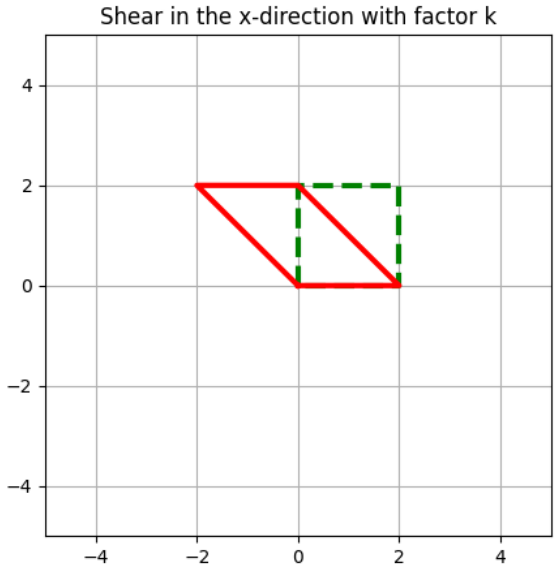
10. *Phép trượt nghiêng theo trục x với hệ số k –* ***shear_x_direction(matrix, k)****:*
Sử dụng ma trận chuẩn là [[2, k], [0, 2]]

a. Theo chiều dương của trục x ($k > 0$):
Giả sử $k = 2$



Hình 11.1: Hình minh họa phép cắt nghiêng theo chiều dương của trục x với hệ số k

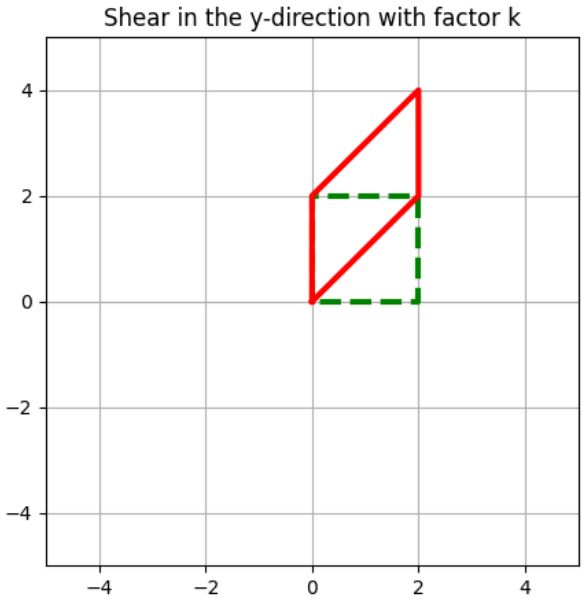
b. Theo chiều âm của trục x ($k < 0$):
Giả sử $k = -2$



Hình 11.2: Hình minh họa phép cắt nghiêng theo chiều âm của trục x với hệ số k

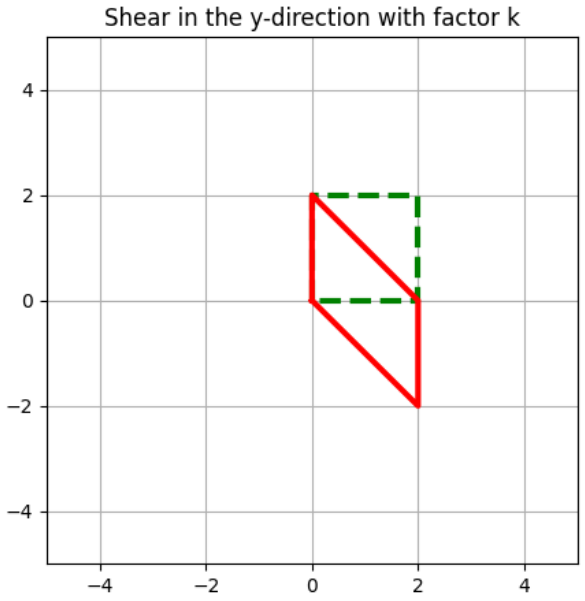
11. *Phép trượt nghiêng theo trục y với hệ số k –* ***shear_y_direction(matrix, k)****:*
Sử dụng ma trận chuẩn là [[2, 0], [k, 2]]

a. Theo chiều dương của trục y ($k > 0$):
Giả sử $k = 2$



Hình 12.1: Hình minh họa phép cắt nghiêng theo chiều dương của trục y với hệ số k

b. Theo chiều âm của trục y ($k < 0$):
Giả sử $k = -2$



Hình 12.2: Hình minh họa phép cắt nghiêng theo chiều âm của trục y với hệ số k

IV.Kết luận:

- Đại số tuyến tính đóng vai trò quan trọng trong việc mô hình hóa và giải quyết các bài toán liên quan đến phép biến hình trong không gian hai chiều R^2.
- Ứng dụng của đại số tuyến tính trong các phép biến hình giúp chúng ta hiểu sâu hơn về cách chúng ảnh hưởng đến hình dạng và vị trí của các đối tượng. Nó cung cấp cho chúng ta một khung công cụ mạnh mẽ để thực hiện tính toán và tìm hiểu về các biến đổi không gian hai chiều.
- Qua đó, đại số tuyến tính không chỉ áp dụng trong lĩnh vực toán học mà còn có ứng dụng rộng rãi trong các lĩnh vực như đồ họa máy tính, địa lý học, điều khiển tự động, xử lý ảnh và thị giác máy tính. Việc áp dụng đại số tuyến tính trong các bài toán thực tế giúp chúng ta nắm bắt và giải quyết các vấn đề phức tạp một cách hiệu quả và chính xác.
- Tóm lại, đại số tuyến tính mang lại lợi ích đáng kể trong việc hiểu và ứng dụng các phép biến hình trong nhiều lĩnh vực thực tế.