Final Praktikum

Dasar Pemrograman 2024

Fourth Wall

Soal oleh: Rakha Fathin

Ilustrasi oleh: cherii & Rakha

Description

Suatu hari saat Furina sedang jalan-jalan, tiba-tiba di suatu jalan ada semacam kerangka persegi panjang (*Bagi kita, player, itu adalah kamera*). Furina penasaran dan akhirnya dia membentuk suatu bangun ruang dengan kekuatannya dan mengarahkannya ke persegi panjang itu. Dia mencoba mentransformasikan (merotasi, translasi, refleksi, dan skalasi) bangun tersebut. Dari sudut pandang kita, bangun-bangun tersebut bisa terlihat oleh kamera jika bangun tersebut berada di depan kamera dan masuk bingkai kamera.

Akan terdapat 4 titik koordinat kamera. Lalu **M** buah bangun ruang pada koordinat ruang (x, y, z) dengan titik sudutnya masing-masing sejumlah **N** (Tergantung jenis bangun ruangnya). setelah itu, akan ada **T** perintah transformasi dari Furina kepada bangun dengan index tertentu. Tentukan apakah setiap bangun terlihat dari kamera atau tidak.

Berikut adalah perintah transformasi yang mungkin:

1. $TRANSLATE\ I\ A\ B\ C$

   Memindahkan semua titik sudut bangun ke-I sebanyak A sepanjang sumbu X, B sepanjang sumbu Y, dan C sepanjang sumbu Z.

2. $ROTATE\ I\ S\ D$

   Merotasikan semua titik sudut bangun ke-I sebanyak D derajat terhadap sumbu S (Bisa X, Y, atau Z). Gunakan $\pi = 3.14159265358979323846$

3. $REFLECT\ I\ S$

   Mencerminkan semua titik sudut bangun ke-I terhadap sumbu S (Bisa X, Y, atau Z).

4. $SCALE\ I\ A\ B\ C$

   Mengalikan semua titik sudut bangun ke-I sebanyak A pada sumbu X, B pada sumbu Y, dan C pada sumbu Z.

Kamera berbentuk persegi panjang dalam ruang 3D dan posisinya berada pada bidang $X = 0$, bingkai kamera dapat didefinisikan oleh keempat titik sudutnya:

$$(0,\ y_{min}, z_{min}),\ (0,\ y_{max}, z_{min}),\ (0,\ y_{min}, z_{max}),\ (0,\ y_{max}, z_{max})$$

Syarat agar **suatu titik** (x_a, y_a, z_a) **terlihat oleh kamera** adalah:

1. $x_a > 0$ (Titik tersebut berada di depan kamera)
2. $y_{min} < y_a < y_{max}$

   $z_{min} < z_a < z_{max}$

   (Titik tersebut berada dalam bingkai kamera)

Berikut adalah ketentuan bangun dan visibilitasnya oleh kamera:

1. Jika semua titik sudut bangun tersebut terlihat oleh kamera, maka bangun tersebut **FULLY INFRAME**.
2. Jika terdapat minimal satu titik sudut dan tidak semua titik sudut bangun tersebut terlihat oleh kamera, maka bangun tersebut **PARTIALLY INFRAME**.
3. Jika tidak ada sama sekali titik sudut bangun tersebut terlihat oleh kamera, maka bangun tersebut **OUT OF FRAME**.

Jenis-jenis bangun ruang yang mungkin antara lain :

1. Kubus (“CUBE”)
2. Prisma Segitiga (“PRISM”)
3. Limas Segiempat (“PYRAMID”)

Perlu diketahui bahwa bangun-bangun tersebut memiliki jumlah titik sudut yang berbeda.

HINT: MATRIKS TRANSFORMASI

Input Format

4 Baris pertama merupakan 4 titik koordinat kamera. Baris selanjutnya merupakan sebuah integer M. Lalu, diikuti dengan M bangun beserta N titik-titik sudutnya. Setelah itu, sebuah integer T. Lalu, diikuti dengan T perintah transformasi.

$x_{C1}\ y_{C1}\ z_{C1}$

$x_{C2}\ y_{C2}\ z_{C2}$

$x_{C3}\ y_{C3}\ z_{C3}$

$x_{C4}\ y_{C4}\ z_{C4}$

M

$Shape1$

$x_{11}\ y_{11}\ z_{11}$

$x_{12}\ y_{12}\ z_{12}$

..

$x_{1N}\ y_{1N}\ z_{1N}$

$Shape2$

$x_{21}\ y_{21}\ z_{21}$

$x_{22}\ y_{22}\ z_{22}$

...

$ShapeM$

$x_{M1}\ y_{M1}\ z_{M1}$

...

$x_{MN}\ y_{MN}\ z_{MN}$

T

$Transformation_1$

$Transformation_2$

...

$Transformation_T$

Output Format

Keluarkan semua bangun beserta visibilitasnya oleh kamera setelah semua perintah transformasi dilakukan.

$Shape\ 1\colon\ TypeOfShape,\ Visibility$

$Shape\ 2\colon\ TypeOfShape,\ Visibility$

...

$Shape\ M\colon\ TypeOfShape,\ Visibility$

Dengan ketentuan string $Visibility$ sebagai berikut:

- Jika bangun **FULLY INFRAME**, $Visibility = "can\ be\ seen\ clearly\ in\ all\ its\ beauty!"$
- Jika bangun **PARTIALLY INFRAME**, $Visibility = "is...\ not\ looking\ complete?"$
- Jika bangun **OUT OF FRAME**, $Visibility = "is\ in\ your\ imagination\ only!"$

Constraint

$2 \leq M \leq 10$

$-\ 10^6 < Semua\ titik\ koordinat < 10^6$

$1 \leq T \leq 20$

$-\ 360 \leq D\ (derajat\ perintah\ rotasi) \leq 360$

$-\ 10^6 \leq A,\ B,\ C\ (perintah\ translasi\ dan\ skalasi) \leq 10^6$

$S\ (sumbu\ koordinat\ ruang) \in \{X,\ Y,\ Z\}$

Sample Input 0

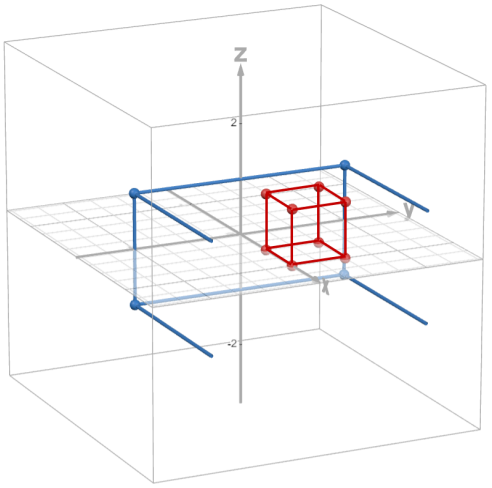
```
0 2 1
0 2 -1
0 -2 -1
0 -2 1
1
CUBE
1 0 0
1 1 0
1 0 1
1 1 1
2 0 0
2 1 0
2 0 1
2 1 1
3
ROTATE 1 X 120
SCALE 1 0.7 2 1.2
REFLECT 1 X
```

Sample Output 0

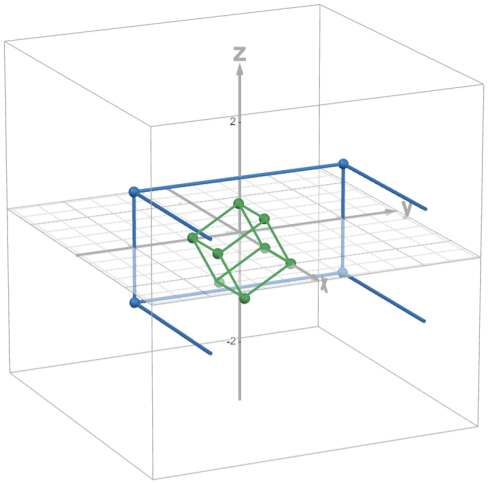
```
Shape 1: CUBE, is... not looking complete?
```

Penjelasan Output 0

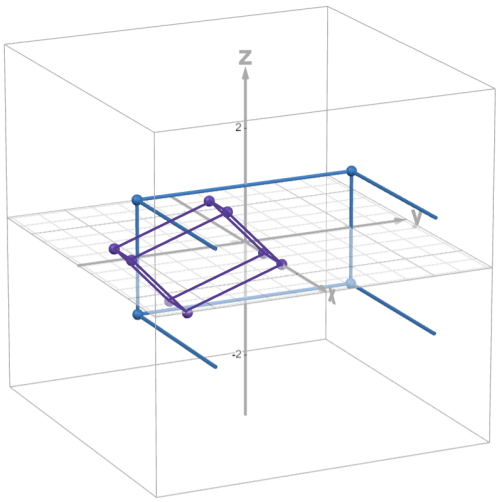
Berikut adalah visualisasi bangun ke-1 pada awalnya. Dengan kerangka biru sebagai kamera



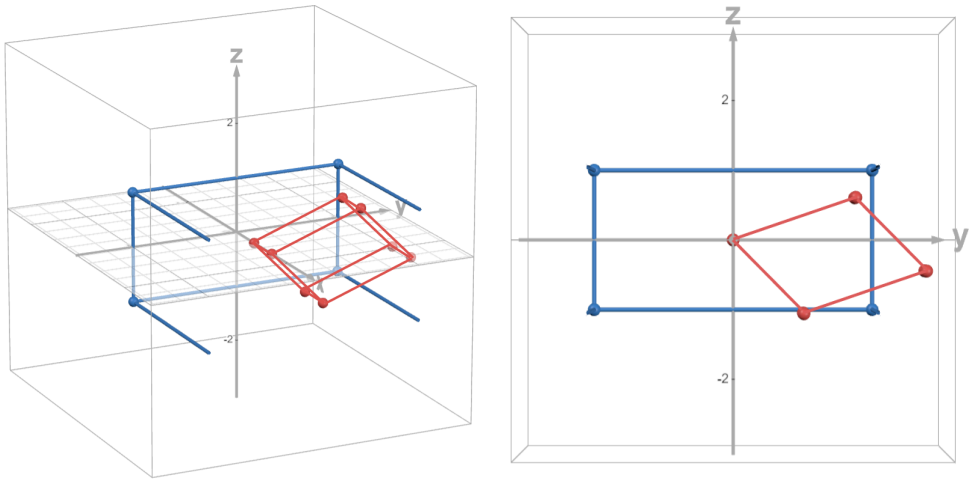
Berikut adalah visualisasi bangun ke-1 setelah dilakukan perintah transformasi ROTATE 120 derajat terhadap sumbu x.



Berikut adalah visualisasi bangun ke-1 setelah dilakukan perintah transformasi SCALE 0.7 terhadap sumbu X, 2 terhadap sumbu Y, dan 1.2 terhadap sumbu Z



Berikut adalah visualisasi bangun ke-1 setelah dilakukan perintah transformasi REFLECT terhadap sumbu X



Karena ada titik yang tidak terlihat oleh kamera, maka bangun tersebut **PARTIALLY INFRAME**.

Sample Input 1

```
0 5.6 3.1
0 5.6 -3.1
0 -5.6 -3.1
0 -5.6 3.1
2
PRISM
2 0 2.4
2 1.3 0
2 1 1.5
3.5 1 1.5
3.5 1.3 0
3.5 0 2.4
PYRAMID
-1 -3 3
-1 -3 -3
```

```
-1 3 3
-1 3 -3
-6 0 0
5
REFLECT 1 Z
SCALE 1 -4 10 9
TRANSLATE 2 6.1 0 0
ROTATE 1 Y 134
TRANSLATE 1 1 -0.5 1.6
```

Sample Output 1

```
Shape 1: PRISM, is in your imagination only!
Shape 2: PYRAMID, can be seen clearly in all its beauty!
```

Selamat karena telah sampai disini! Kamu telah menamatkan Dasar Pemrograman, kelas paling menyenangkan di Teknik Informatika! Semangat kedepannya!

Practicum Finale

Fundamental Programming 2024

Fourth Wall

Problem by: Rakha Fathin

Illustration by: cherii & Rakha

Description

One day, while taking a walk, Furina came across a rectangular frame (*for us, the players, it represents a camera*). Curious, she used her powers to create a three-dimensional shape and directed it toward the rectangular frame. She experimented with transforming (rotating, translating, reflecting, and scaling) the shape. From our perspective, the shapes are visible through the camera if they are positioned in front of the camera and within its frame.

There will be 4 camera points in space coordinates. Then **M** three-dimensional shapes located in 3D space coordinates (x, y, z), each with **N** vertices (depending on the type of shape). Afterward, there will be **T** transformation commands from Furina targeting specific shapes by their index. Determine whether each shape is visible through the camera or not.

The possible transformation commands are as follows:

1. $TRANSLATE\ I\ A\ B\ C$

   Moves all vertices of the I-th shape by A units along the X-axis, B units along the Y-axis, and C units along the Z-axis.

2. $ROTATE\ I\ S\ D$

   Rotates all vertices of the I-th shape by D degrees about axis S (can be X, Y, or Z). Use $\pi = 3.14159265358979323846$

3. $REFLECT\ I\ S$

   Reflects all vertices of the I-th shape across axis S (can be X, Y, or Z).

4. $SCALE\ I\ A\ B\ C$

   Scales all vertices of the I th shape by a factor of A along the axis X, B along the axis Y, and C along the axis Z.

The camera is a rectangular frame in 3D space, positioned on the plane $X = 0$. The camera frame can be defined by its four corner points:

$$(0,\ y_{min}, z_{min}),\ (0,\ y_{max}, z_{min}),\ (0,\ y_{min}, z_{max}),\ (0,\ y_{max}, z_{max})$$

For **a point** (x_a, y_a, z_a) **to be visible through the camera**, it must satisfy the following conditions:

1. $x_a > 0$ (The point is in front of the camera)
2. $y_{min} < y_a < y_{max}$

   $z_{min} < z_a < z_{max}$

   (The point is within the camera's frame)

Here are the visibility conditions for shapes:

1. If all vertices of a shape are visible through the camera, the shape is classified as **FULLY INFRAME**.
2. If at least one vertex but not all vertices of a shape are visible through the camera, the shape is classified as **PARTIALLY INFRAME**.
3. If no vertices of a shape are visible through the camera, the shape is classified as **OUT OF FRAME**.

Possible types of 3D shapes:

1. Cube (“CUBE”)
2. Triangular Prism (“PRISM”)
3. Square Pyramid (“PYRAMID”)

It should be noted that those shapes have a different number of vertices.

HINT: TRANSFORMATION MATRIX

Input Format

The first 4 lines represent the 4 corner points of the camera. The next line contains an integer M, representing the number of 3D shapes. This is followed by M shapes, each defined by their N corner points. Afterward, an integer T is given, representing the number of transformation commands. This is followed by T transformation commands.

$x_{C1}\ y_{C1}\ z_{C1}$

$x_{C2}\ y_{C2}\ z_{C2}$

$x_{C3}\ y_{C3}\ z_{C3}$

$x_{C4}\ y_{C4}\ z_{C4}$

M

$Shape1$

$x_{11}\ y_{11}\ z_{11}$

$x_{12}\ y_{12}\ z_{12}$

..

$x_{1N}\ y_{1N}\ z_{1N}$

$Shape2$

$x_{21}\ y_{21}\ z_{21}$

$x_{22}\ y_{22}\ z_{22}$

…

$ShapeM$

$x_{M1}\ y_{M1}\ z_{M1}$

…

$x_{MN}\ y_{MN}\ z_{MN}$

T

$Transformation_1$

$Transformation_2$

…

$Transformation_T$

Output Format

For each shape, output its visibility from the camera after all transformation commands are applied.

$Shape\ 1\colon\ TypeOfShape,\ Visibility$

$Shape\ 2\colon\ TypeOfShape,\ Visibility$

...

$Shape\ M\colon\ TypeOfShape,\ Visibility$

With the string $Visibility$ are as follows

- If a shape is **FULLY INFRAME**, $Visibility = "can\ be\ seen\ clearly\ in\ all\ its\ beauty!"$
- If a shape is **PARTIALLY INFRAME**, $Visibility = "is...\ not\ looking\ complete?"$
- If a shape is **OUT OF FRAME**, $Visibility = "is\ in\ your\ imagination\ only!"$

Constraint

$2 \leq M \leq 10$

$-\ 10^6 < All\ coordinate\ points < 10^6$

$1 \leq T \leq 20$

$-\ 360 \leq D\ (degree\ for\ rotation) \leq 360$

$-\ 10^6 \leq A,\ B,\ C\ (for\ translation\ and\ scale) \leq 10^6$

$S\ (coordinate\ axes) \in \{X,\ Y,\ Z\}$

Sample Input 0

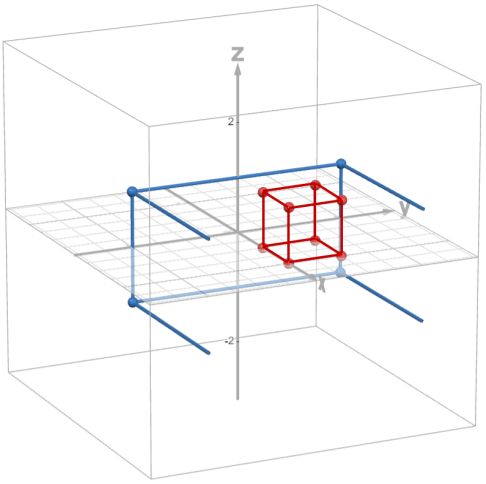
```
0 2 1
0 2 -1
0 -2 -1
0 -2 1
1
CUBE
1 0 0
1 1 0
1 0 1
1 1 1
2 0 0
2 1 0
2 0 1
2 1 1
3
ROTATE 1 X 120
SCALE 1 0.7 2 1.2
REFLECT 1 X
```

Sample Output 0

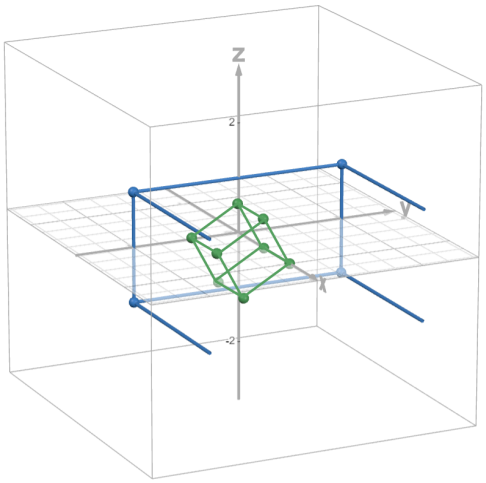
```
Shape 1: CUBE, is... not looking complete?
```

Output 0 Explanation

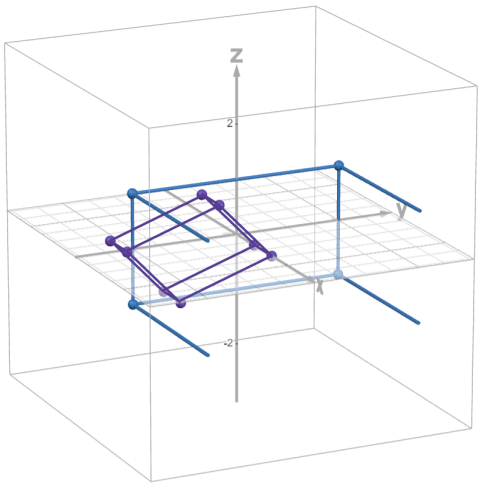
Initially, the visualization of the first shape is shown. With the blue frame as the camera.



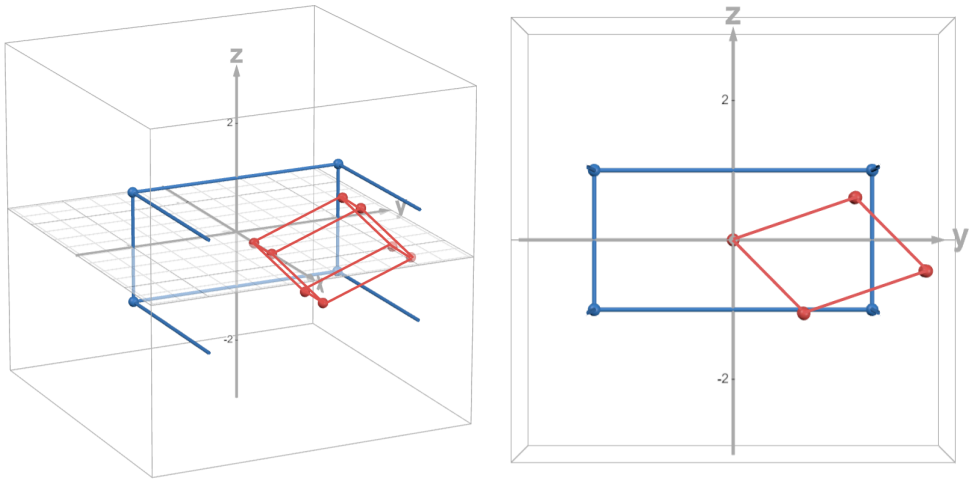
After applying the transformation command to ROTATE 120 degrees about the X-axis, the updated visualization of the first shape is shown.



After applying the transformation command to SCALE by 0.7 along the X-axis, 2 along the Y-axis, and 1.2 along the Z-axis, the updated visualization of the first shape is shown.



After applying the transformation command to REFLECT over the X-axis, the updated visualization of the first shape is shown.



Because there are vertices that can't be seen by the camera, then that shape is **PARTIALLY INFRAME**.

Sample Input 1

```
0 5.6 3.1
0 5.6 -3.1
0 -5.6 -3.1
0 -5.6 3.1
2
PRISM
2 0 2.4
2 1.3 0
2 1 1.5
3.5 1 1.5
3.5 1.3 0
3.5 0 2.4
PYRAMID
-1 -3 3
-1 -3 -3
```

```
-1 3 3
-1 3 -3
-6 0 0
5
REFLECT 1 Z
SCALE 1 -4 10 9
TRANSLATE 2 6.1 0 0
ROTATE 1 Y 134
TRANSLATE 1 1 -0.5 1.6
```

Sample Output 1

```
Shape 1: PRISM, is in your imagination only!
Shape 2: PYRAMID, can be seen clearly in all its beauty!
```

Congratulations for making it this far! You have finally finished Fundamental Programming, the most fun class in Informatics! Good luck going forward!