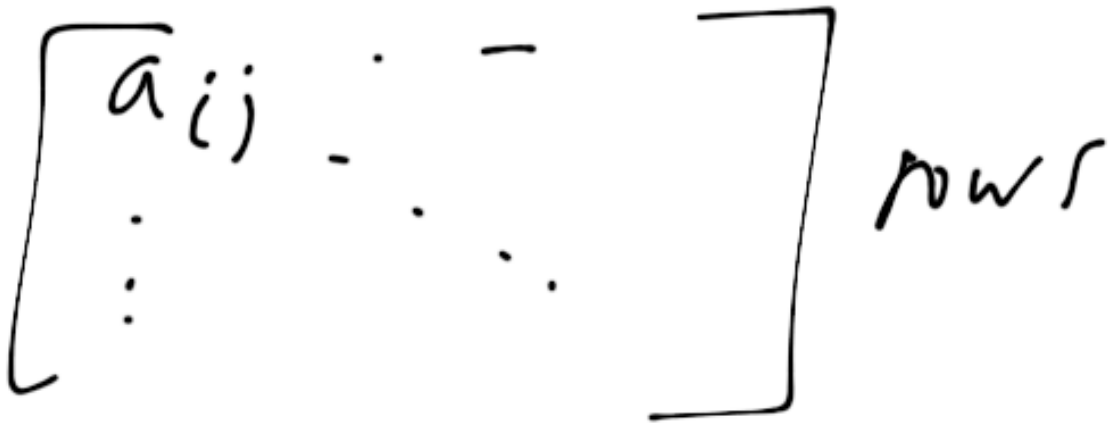
Data Structures thru Jar:

- array { own & STL vector<>
- linked list } own & STL list<>
- stack (queue)
  (binheap
- binary search tree (BST)
- AVL tree (balanced BST)
- hash tables
- associative array (STL map<>)
- kd-tree

- matrix:



- particularly useful for 3D rotations (in computer graphics)
  - ↳ 4x4 matrix

e.g.

3D point (x, y, vertex)

$$\begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

1×4 $\quad$ 4×4

$=$ $\quad$ $=$

$$= \begin{bmatrix} x' & y' & z' & 1 \end{bmatrix}$$

1×4

rotates point about z-axis by θ°

Rot by y:

$$\begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Similarly for rot by x

4x4 matrix

$[x\ y\ z\ 1]$: 4-vector

q

a 'one', representing

(3)

point using

homogeneous coords

- matrix in C++?
- use private:

std::vector<std::vector<T>>
a[1];

Vector of vectors

for any m x n matrix

- Constructor:

```
matrix(int r = 4, int c = 4):
    arr(r)
{ for (int i = 0; i < r; i++)
    arr[i].resize(c); }
```

- accessor / mutator operator

```
const std::vector<T>&
    operator[](int r) const
{ return arr[r]; }

std::vector<T>&
    operator[](int r)
{ return arr[r]; }
```

- other functions:

e.g. operator- (void)

operator! (void),

matrix inverse

e.g. A, A^{-1} is inverse of A $AA^{-1} = I$

many operators, negate, cube

e.g., operator/(void):

// compute matrix inverse

e.g., look up "Num. Recipes in C" (or C++) & use something like LU decomposition

- idea for math lib
(& code development
in general):
implement desired
functionality & write
test suite to validate
code
↳ test driver.