Example 1

1. Create a window (InitWindow)
2. Close a window (CloseWindow)
3. Drawing in window (BeginDrawing, EndDrawing)
4. Change background color every frame.
5. Draw circle.
6. Why is it in the top left corner?
7. Whiteboard.
8. Screen Space to World Space.

```
#include <math.h>
#include <raylib.h>

#define WORLD_UNIT (100)
#define SCREEN_WIDTH (640)
#define SCREEN_HEIGHT (480)

double screen_x_to_world(double x, double unit) {
	return (x * unit) + GetScreenWidth()/2;
}

int main(void) {
	InitWindow(SCREEN_WIDTH, SCREEN_HEIGHT, "RaylibExample");

	double x = 0;

	while (!WindowShouldClose()) {
		BeginDrawing();
		ClearBackground(BLACK);
		DrawCircle(screen_x_to_world(x, GetScreenWidth()/2), GetScreenHeight()/2, 15, GREEN);
		EndDrawing();
		x = sinf(GetTime());
	}

	CloseWindow();

	return 0;
}
```

Example 2

1. The structure of a matrix is similar to an excel spread sheet. Rows and columns.
2. Operations with a matrix and a vector,
3. What is happening during matrix multiplication, product and sum of components
4. Identity matrix doesn’t affect the final vector.
5. How to change the position of point using matrix. 3X3 matrix
6. Lets try some things in code.

Example 3

1. Define geometry on whiteboard
2. Go into math.h and explain code

```
#include "scratch.h"

#include "draw.h"
#include "math.h"

// draw on whiteboard
static vec2 verts[4] = {
	{-1, 1},
	{ 1, 1},
	{ 1, -1},
	{-1, -1},
};

static int indices[8] = {
	0, 1,
	1, 2,
	2, 3,
	3, 0
};

mat3x3 model;

void setup(void) {
	model = mat3x3_identity();
}

void update(void) {
	// translation
	// scaling
	// skew then rotation
}

void draw(void) {
	draw_shape(&model, verts, 4, indices, 8);
}
```

Example 3

1. Define a cube [Whiteboard?]
2. Look at perspective matrix implementation.

```
#include "scratch.h"
#include "draw.h"
#include "math.h"

vec3 cube[8] = {
	(vec3){ 1, -1, 1},
	(vec3){ 1, 1, 1},
	(vec3){ 1, -1, -1},
	(vec3){ 1, 1, -1},
	(vec3){-1, -1, 1},
	(vec3){-1, 1, 1},
	(vec3){-1, -1, -1},
	(vec3){-1, 1, -1},
};

int indices[24] = {
	0, 1,
	0, 2,
	1, 3,
	2, 3,
	4, 5,
	5, 7,
	6, 7,
	4, 6,
	0, 4,
	1, 5,
	2, 6,
	3, 7,
};

mat4x4 model;
mat4x4 perspective;
mat4x4 MP;
void setup(void) {
	model = mat4x4_identity();
	perspective = mat4x4_perspective(100, 0.001, 70, 1);
}

void update(void) {
	model = mat4x4_translate((vec3){sinf(GetTime()), 0, -5});
	model = mat4x4_mul(model, mat4x4_rotate_x(GetTime()));
	model = mat4x4_mul(model, mat4x4_rotate_z(GetTime()));
	model = mat4x4_mul(model, mat4x4_scale((vec3){1, 1, 1}));

	MP = mat4x4_mul(perspective, model);
}

void draw(void) {
	draw_shape(MP, cube, 8, indices, 24);
}
```