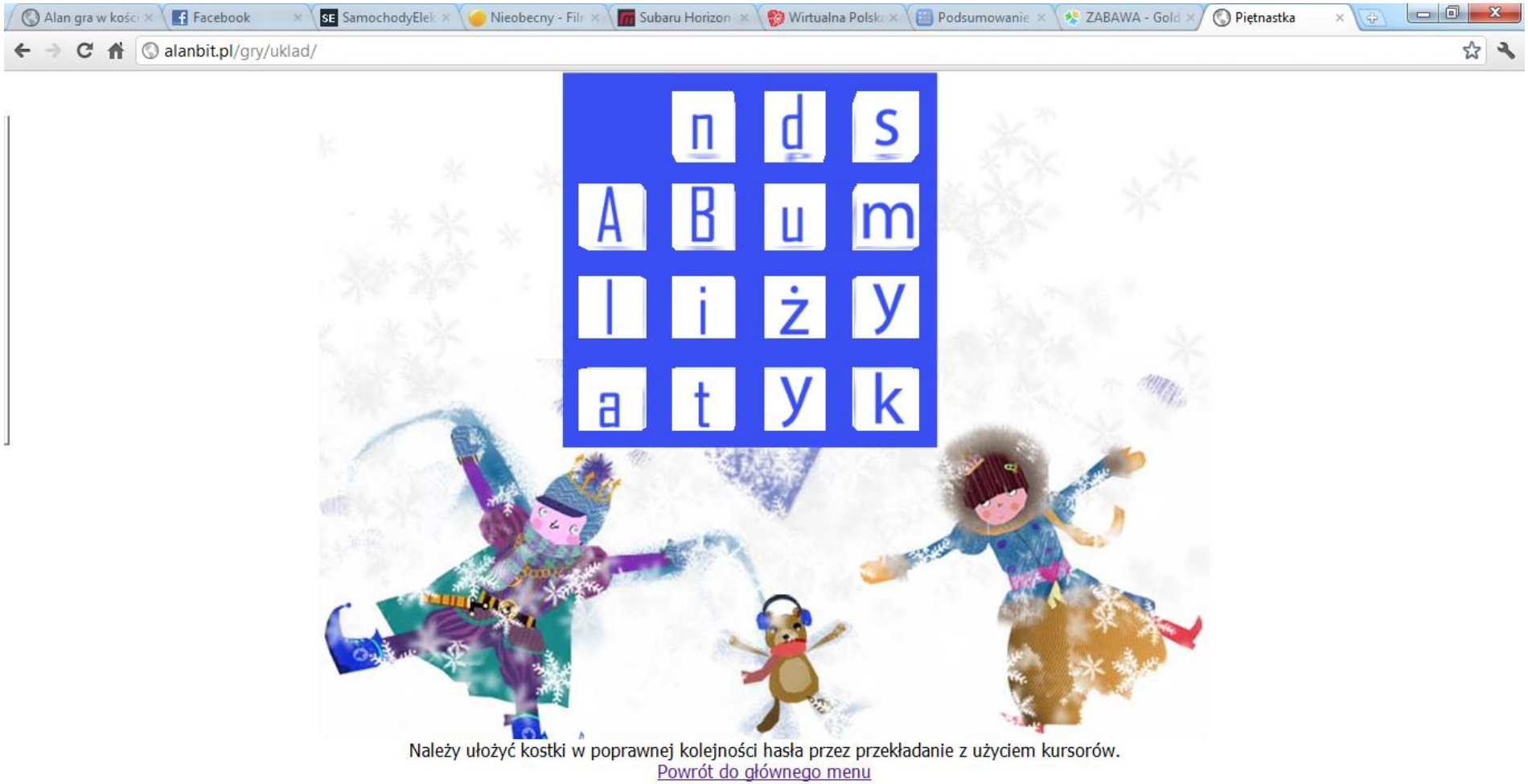
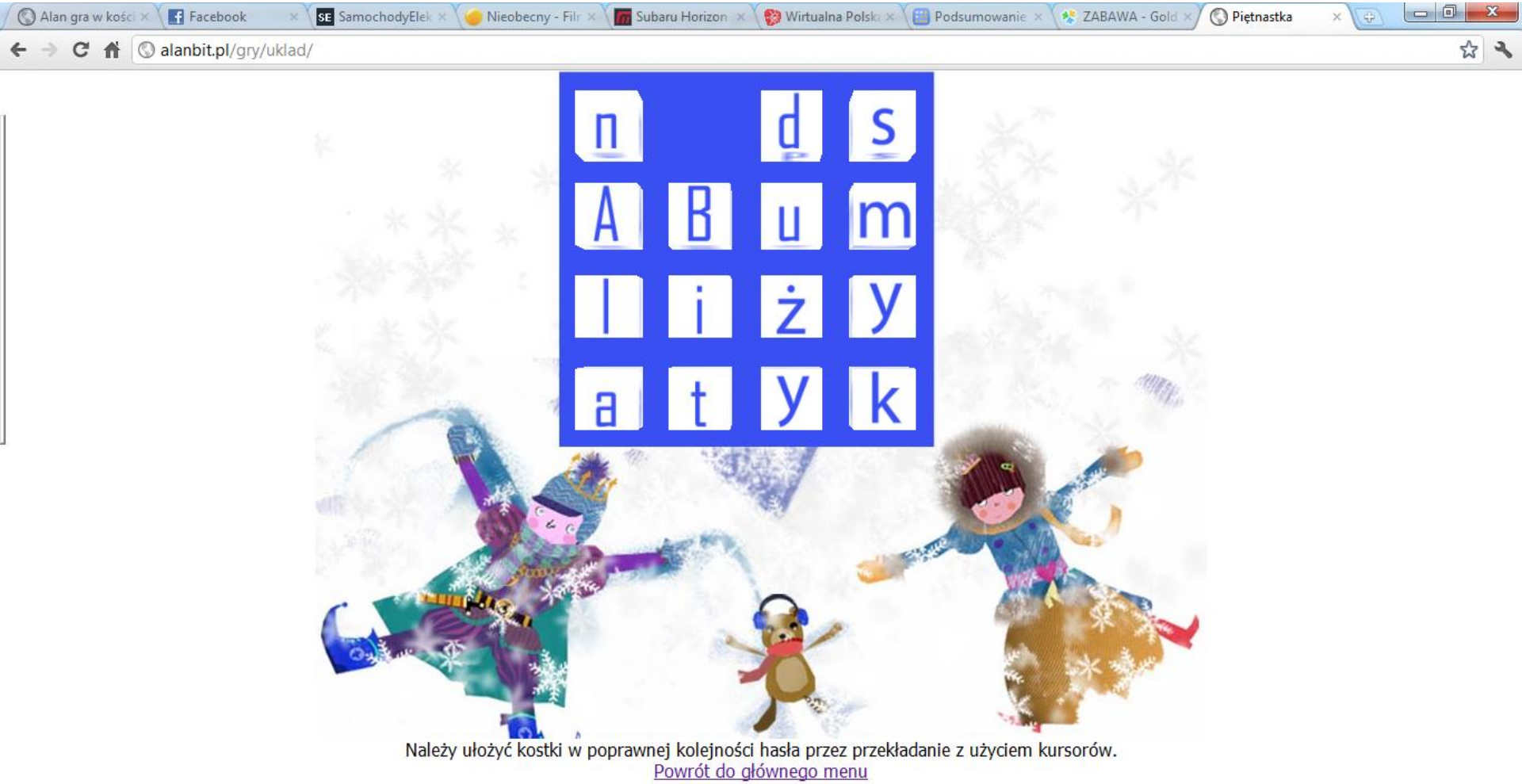
Gra wykorzystująca tekstury

Andrzej P.Urbański

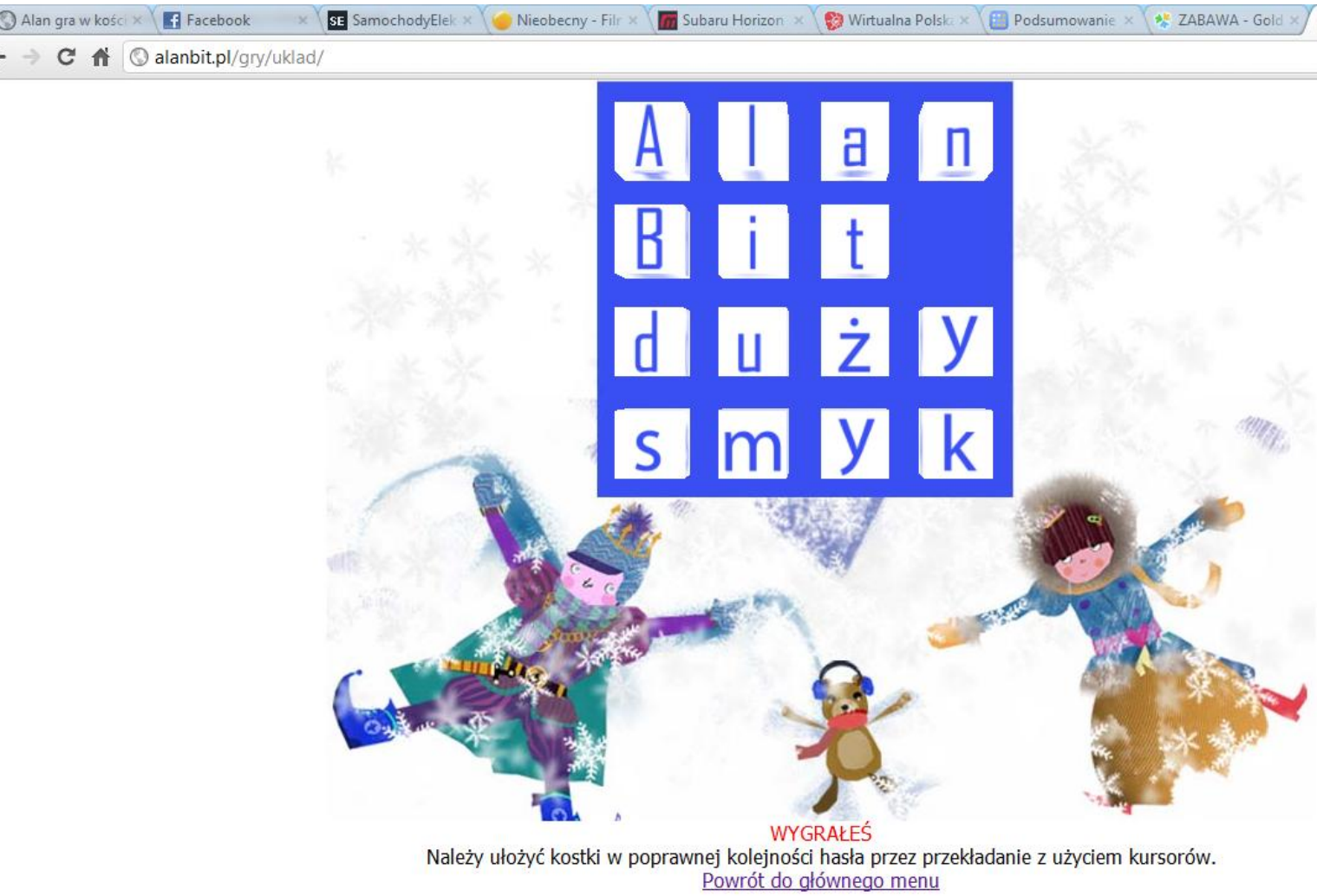
Piętnastka na początku



Przesunięcie strzałką w prawo



Stan końcowy gry



Dołączanie bibliotek

```
<html>
<head>
<title>Piętnastka</title>
<meta http-equiv="content-type" content="text/html;
   charset=utf-8">
 <script type="application/x-javascript"
   src="WGLtools.js"></script>
<script type="text/javascript" src="sylvester.js"></script>
<script type="text/javascript" src="glUtils.js"></script>
<link type="text/css" href="styles.css" rel="stylesheet" />
```

Fragment shader tekstur

```
<script id="shader-fs" type="x-shader/x-fragment">
  #ifdef GL_ES
     precision highp float;
  #endif
  varying vec2 vTextureCoord;

  uniform sampler2D uSampler;

  void main(void) {
    gl_FragColor = texture2D(uSampler, vec2(vTextureCoord.s, 1.0 -
     vTextureCoord.t));
  }
</script>
```

Fragment shader wierzchołków

```
<script id="shader-vs" type="x-shader/x-vertex">
  attribute vec3 aVertexPosition;
  attribute vec2 aTextureCoord;

  uniform mat4 uMVMatrix;
  uniform mat4 uPMatrix;

  varying vec2 vTextureCoord;


  void main(void) {
    gl_Position = uPMatrix * uMVMatrix * vec4(aVertexPosition, 1.0);
    vTextureCoord = aTextureCoord;
  }
</script>
```

Inicjowanie webGL

```
<script type="text/javascript">
var gl;
function initGL(canvas) {
  try {
    gl = canvas.getContext("experimental-webgl");
    gl.viewport(0, 0, canvas.width, canvas.height);
  } catch(e) {
  }
        if (!gl)
        {
                if( confirm("Could not initialise WebGL. \nDo you want to go to installation page?") )
                {
                        location.href="http://3dgames.pl/blog/2010/03/ustawienia-przegladarki/";
                }
        }
}
```

Wyszukiwanie shaderów

```
function getShader(gl, id) {
  var shaderScript = document.getElementById(id);
  if (!shaderScript) {
    return null;
  }

  var str = "";
  var k = shaderScript.firstChild;
  while (k) {
    if (k.nodeType == 3) {
      str += k.textContent;
    }
    k = k.nextSibling;
  }
```

Kompilacja shaderów

```
var shader;
  if (shaderScript.type == "x-shader/x-fragment") {
    shader = gl.createShader(gl.FRAGMENT_SHADER);
  } else if (shaderScript.type == "x-shader/x-vertex") {
    shader = gl.createShader(gl.VERTEX_SHADER);
  } else {
    return null;
  }

  gl.shaderSource(shader, str);
  gl.compileShader(shader);

  if (!gl.getShaderParameter(shader, gl.COMPILE_STATUS)) {
    alert(gl.getShaderInfoLog(shader));
    return null;
  }

  return shader;
}
```

Inicjowanie shadera 1

```
var shaderProgram;
function initShaders() {
  var fragmentShader = getShader(gl, "shader-fs");
  var vertexShader = getShader(gl, "shader-vs");

  shaderProgram = gl.createProgram();
  gl.attachShader(shaderProgram, vertexShader);
  gl.attachShader(shaderProgram, fragmentShader);
  gl.linkProgram(shaderProgram);

  if (!gl.getProgramParameter(shaderProgram, gl.LINK_STATUS)) {
    alert("Could not initialise shaders");
  }
```

Inicjowanie shadera 2

```
 gl.useProgram(shaderProgram);

  shaderProgram.vertexPositionAttribute = gl.getAttribLocation(shaderProgram,
      "aVertexPosition");
  gl.enableVertexAttribArray(shaderProgram.vertexPositionAttribute);

  shaderProgram.textureCoordAttribute = gl.getAttribLocation(shaderProgram,
      "aTextureCoord");
  gl.enableVertexAttribArray(shaderProgram.textureCoordAttribute);

  shaderProgram.pMatrixUniform = gl.getUniformLocation(shaderProgram,
      "uPMatrix");
  shaderProgram.mvMatrixUniform = gl.getUniformLocation(shaderProgram,
      "uMVMatrix");
  shaderProgram.samplerUniform = gl.getUniformLocation(shaderProgram,
      "uSampler");
}
```

Działania na macierzach video 1

```
var mvMatrix;
var mvMatrixStack = [];

function mvPushMatrix(m) {
  if (m) {
    mvMatrixStack.push(m.dup());
    mvMatrix = m.dup();
  } else {
    mvMatrixStack.push(mvMatrix.dup());
  }
}

function mvPopMatrix() {
  if (mvMatrixStack.length == 0) {
    throw "Invalid popMatrix!";
  }
  mvMatrix = mvMatrixStack.pop();
  return mvMatrix;
}
```

Działania na macierzach video 2

```
function loadIdentity() {
  mvMatrix = Matrix.I(4);
}


function multMatrix(m) {
    mvMatrix = mvMatrix.x(m);
}

function mvTranslate(v) {
    var m = Matrix.Translation($V([v[0], v[1], v[2]])).ensure4x4();
    multMatrix(m);
}
```

Działania na macierzach video 3

```
function mvRotate(ang, v) {
	var arad = ang * Math.PI / 180.0;
	var m = Matrix.Rotation(arad, $V([v[0], v[1], v[2]])).ensure4x4();
	multMatrix(m);
}

function mvRotateRad(rad,v)
{
	var m= Matrix.Rotation(rad, $V([v[0], v[1], v[2]])).ensure4x4();
	multMatrix(m);
}

var pMatrix;
	function perspective(fovy, aspect, znear, zfar) {
	pMatrix = makePerspective(fovy, aspect, znear, zfar);
}


function setMatrixUniforms() {
	gl.uniformMatrix4fv(shaderProgram.pMatrixUniform, false, new Float32Array(pMatrix.flatten()));
	gl.uniformMatrix4fv(shaderProgram.mvMatrixUniform, false, new Float32Array(mvMatrix.flatten()));
}
```

Obsługa klawiatury 1

```
var currentlyPressedKeys = {};

  function handleKeyDown(event) {
    currentlyPressedKeys[event.keyCode] = true;
  }


  function handleKeyUp(event) {
    currentlyPressedKeys[event.keyCode] = false;
  }


  function handleKeys() {
    if (currentlyPressedKeys[33]) {
      // Page Up
      swap(0,2);
                                    currentlyPressedKeys[33] = false;
    }
    if (currentlyPressedKeys[34]) {
      // Page Down
      swap(0,3);
                                    currentlyPressedKeys[34] = false;
    }
```

Obsługa klawiatury 2

```
if (currentlyPressedKeys[37]) {
        // Left cursor key
        move(-1,0);
                                currentlyPressedKeys[37] = false;
    }
    if (currentlyPressedKeys[39]) {
        // Right cursor key
        move(1,0);
                                currentlyPressedKeys[39] = false;
    }
    if (currentlyPressedKeys[38]) {
        // Up cursor key
        move(0,1);
                                currentlyPressedKeys[38] = false;
    }
    if (currentlyPressedKeys[40]) {
        // Down cursor key
        move(0,-1);
                                currentlyPressedKeys[40] = false;
    }
  }
```

Macierz 3d sześcianików

```
var Tab1 = [
        // front face
        -1.0, -1.0,  1.0,
        1.0, -1.0,  1.0,
        1.0,  1.0,  1.0,
        -1.0,  1.0,  1.0,

        // back face
        -1.0, -1.0, -1.0,
        -1.0,  1.0, -1.0,
        1.0,  1.0, -1.0,
        1.0, -1.0, -1.0,

        // top face
        -1.0,  1.0, -1.0,
        -1.0,  1.0,  1.0,
        1.0,  1.0,  1.0,
        1.0,  1.0, -1.0,

        // bottom face
        -1.0, -1.0, -1.0,
        1.0, -1.0, -1.0,
        1.0, -1.0,  1.0,
        -1.0, -1.0,  1.0,

        // right face
        1.0, -1.0, -1.0,
        1.0,  1.0, -1.0,
        1.0,  1.0,  1.0,
        1.0, -1.0,  1.0,

        // left face
        -1.0, -1.0, -1.0,
        -1.0, -1.0,  1.0,
        -1.0,  1.0,  1.0,
        -1.0,  1.0, -1.0,
];
```

Macierz 2D tekstur do sześcianów

```
var Tab2 = [
        // front face
        0.0, 0.0,
        1.0, 0.0,
        1.0, 1.0,
        0.0, 1.0,

        // back face
        1.0, 0.0,
        1.0, 1.0,
        0.0, 1.0,
        0.0, 0.0,

        // top face
        0.0, 1.0,
        0.0, 0.0,
        1.0, 0.0,
        1.0, 1.0,

        // bottom face
        1.0, 1.0,
        0.0, 1.0,
        0.0, 0.0,
        1.0, 0.0,

        // right face
        1.0, 0.0,
        1.0, 1.0,
        0.0, 1.0,
        0.0, 0.0,

        // left face
        0.0, 0.0,
        1.0, 0.0,
        1.0, 1.0,
        0.0, 1.0,
];
```

Macierz podziału na trójkąty

```
var Tab3 = [
0, 1, 2,      0, 2, 3,    // front face
4, 5, 6,      4, 6, 7,    // back face
8, 9, 10,     8, 10, 11,  // top face
12, 13, 14,   12, 14, 15, // bottom face
16, 17, 18,   16, 18, 19, // right face
20, 21, 22,   20, 22, 23  // left face
];
```

Pętla obsługi zdarzeń

```
//time variables
var lastTime = 0;
var elapsed=0;
var elapsed_s=0;
function timeTick()
{
	var timeNow = (new Date).getTime();
	if (lastTime != 0)
	{
		elapsed = timeNow - lastTime;
		elapsed_s=elapsed/1000;
	}
	lastTime = timeNow;
}
function tick()
{
	timeTick();
   handleKeys();
	animate();
	drawScene();
}
```

Wykrywanie warunku zakończenia

```
function animate()
{
	var thend=true;
	var s=0;
	for( var j=0; j<4; j++){
		for( var i=0; i<4; i++){
			if(i+4*j<7)if((Objects[i+4*j+1].Position_x == i*3)
			&&(Objects[i+4*j+1].Position_y == (3-j)*3))
			thend=thend;
			else thend=false;
			if((i+4*j>7)&&(i+4*j!=11)&&(i+4*j!=14))if((Objects[i+4*j].Position_x == i*3)
			&&(Objects[i+4*j].Position_y == (3-j)*3))
			thend=thend;
			else thend=false;			}
	}
	if(thend)document.getElementById("alert").innerHTML='<font color="red">WYGRAŁEŚ</font>';
	else document.getElementById("alert").innerHTML='   ';
}
```

Rysowanie sceny

```
function drawScene()
{
    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT)

    perspective(25, 1.333, 0.1, 100.0);
    loadIdentity();

    mvTranslate([-5.0, -2.0, -40]);
    for(var i=1; i<Objects_amount; i++)
    {
        mvPushMatrix();
        mvTranslate([Objects[i].Position_x, Objects[i].Position_y, 0.0]);
        Objects[i].display();
        mvPopMatrix();
    }
}
```

Sześcianiki

```
function c_Cube()
{
	//position
	this.Position_x = 0;
	this.Position_y = 0;

	//rot
	this.Rot_x = 0;
	this.Rot_y = 0;
	this.Rot_z = 0;

	//rot speed
	this.X_speed;
	this.Y_speed;
	this.Z_speed
}
c_Cube.prototype = new Model;

var Objects = new Array();
var Objects_amount = 16;
```

Uruchamianie webGL

```
function webGLStart() {
	var canvas = document.getElementById("canvas");
	initGL(canvas);
	initShaders();
	for(var i=0; i<Objects_amount; i++)
	{
		var tex_num = i % 16;

		Objects[i] = new c_Cube();
		Objects[i].init(tex_num+".gif",gl.LINEAR,gl.LINEAR_MIPMAP_LINEAR,Tab1,Tab2,Tab3);
		 //Objects[i].Position_y = Objects[i].Position_y.toFixed(0);
		//if(i % 2>0)Objects[i].Position_x = Objects[i].Position_x-2.0;
		Objects[i].X_speed = 0;
		Objects[i].Y_speed = 0;
		Objects[i].Z_speed = 0;
	}
	for( var j=0; j<4; j++){
		for( var i=0; i<4; i++){
			Objects[4*i+j].Position_x = i*3;
			Objects[4*i+j].Position_y = (3-j)*3;
		}
	}
    gl.clearColor(0.0, 0.0, 0.0, 0.0);
    gl.enable(gl.DEPTH_TEST);
    document.onkeydown = handleKeyDown;
    document.onkeyup = handleKeyUp;
	setInterval(tick, 15);
}
```

Zamiany i przesunięcia sześcianików

```
function swap(i,j){
	var x = Objects[i].Position_x;
	var y = Objects[i].Position_y;
	Objects[i].Position_x=Objects[j].Position_x;
	Objects[i].Position_y=Objects[j].Position_y;
	Objects[j].Position_x=x;
	Objects[j].Position_y=y;
}

function move(dx,dy){
	var x = Objects[0].Position_x + 3*dx;
	var y = Objects[0].Position_y + 3*dy;
	var j=0;
	for( var i=1; i<Objects_amount; i++){
		if((Math.abs(x-Objects[i].Position_x)<1.5)&&(Math.abs(y-Objects[i].Position_y)<1.5)){ j=i; }
	//document.getElementById("alert").innerHTML='Znaleziono '+j; }
	}
	if(j>0)swap(j,0);//else document.getElementById("alert").innerHTML='Nie znaleziono '+x+','+y;
}
```

Segment główny

```
</script>


</head>


<body onload="webGLStart();">
      <center>                <div id="game">
 <canvas id="canvas" style="border: none;" width="640" height="480"></canvas>
</div>
  <div id="alert"></div><div>Należy ułożyć kostki w poprawnej kolejności hasła przez przekładanie z
      użyciem kursorów.</div>
  <a href="http://alanbit.pl">Powrót do głównego menu</a>
 </center>
</body>

</html>
```