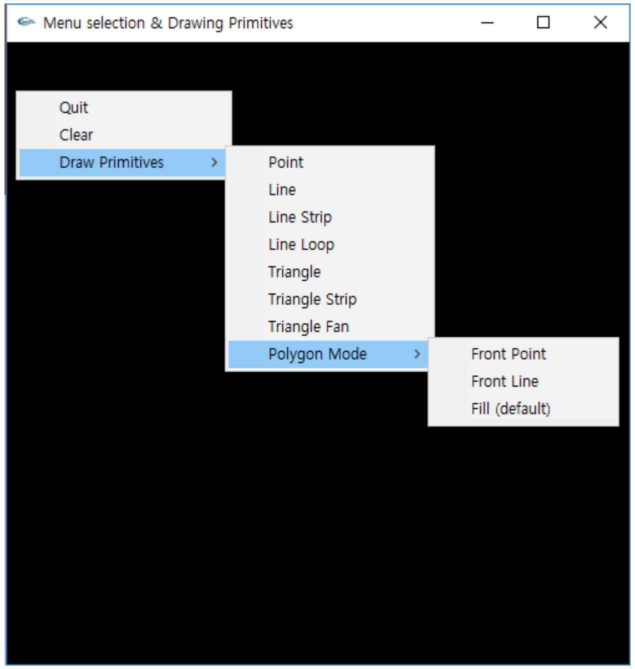
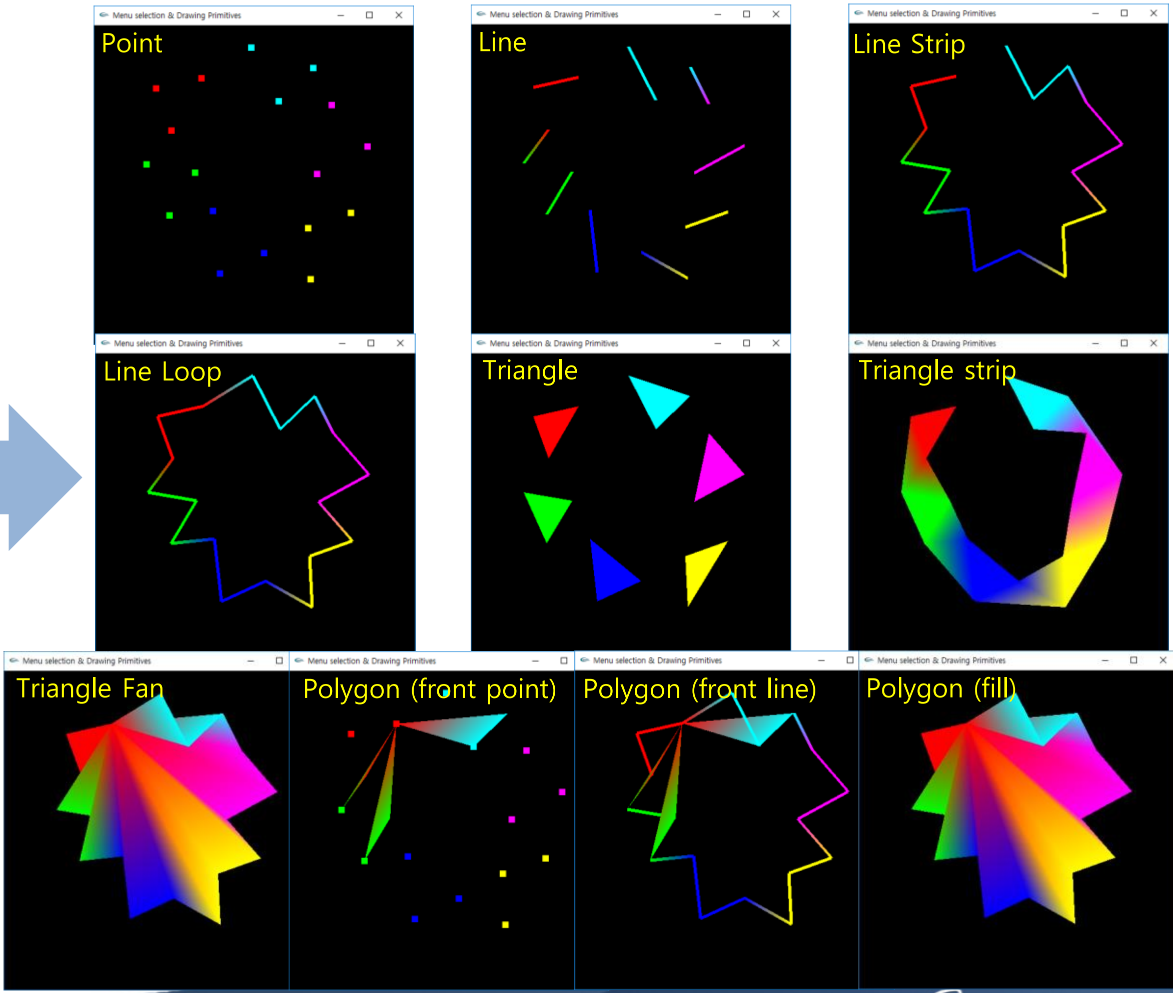
LAB I
Week 04

Seoul National University
Graphics & Media Lab
HyeonSeung Shin

Today's Mission

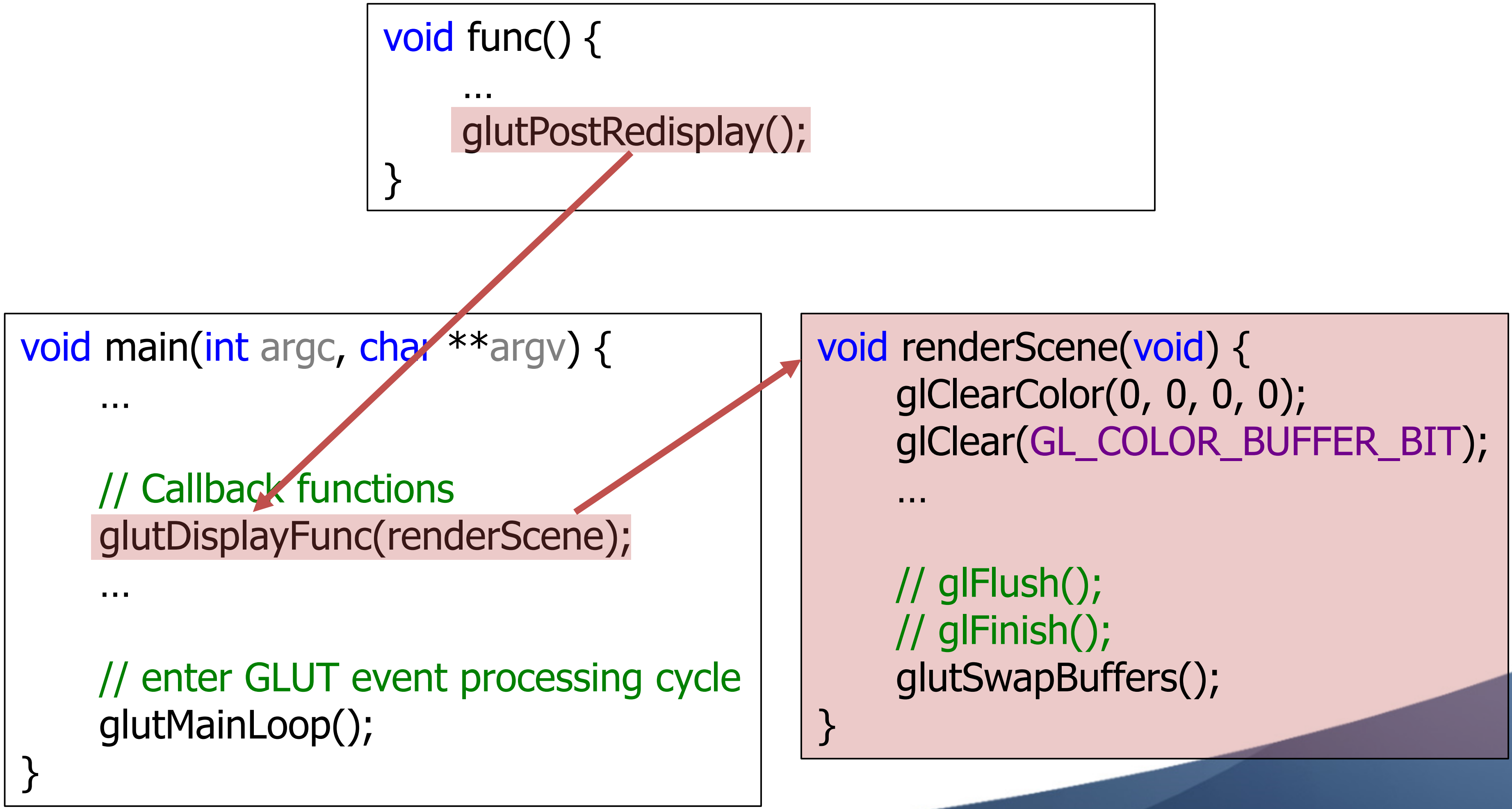
- Draw primitives using popup menu & pressing left mouse button
- Change vertex color using 1~7 keyboard buttons





glutPostRedisplay()

- marks the current window as needing to be redisplayed.

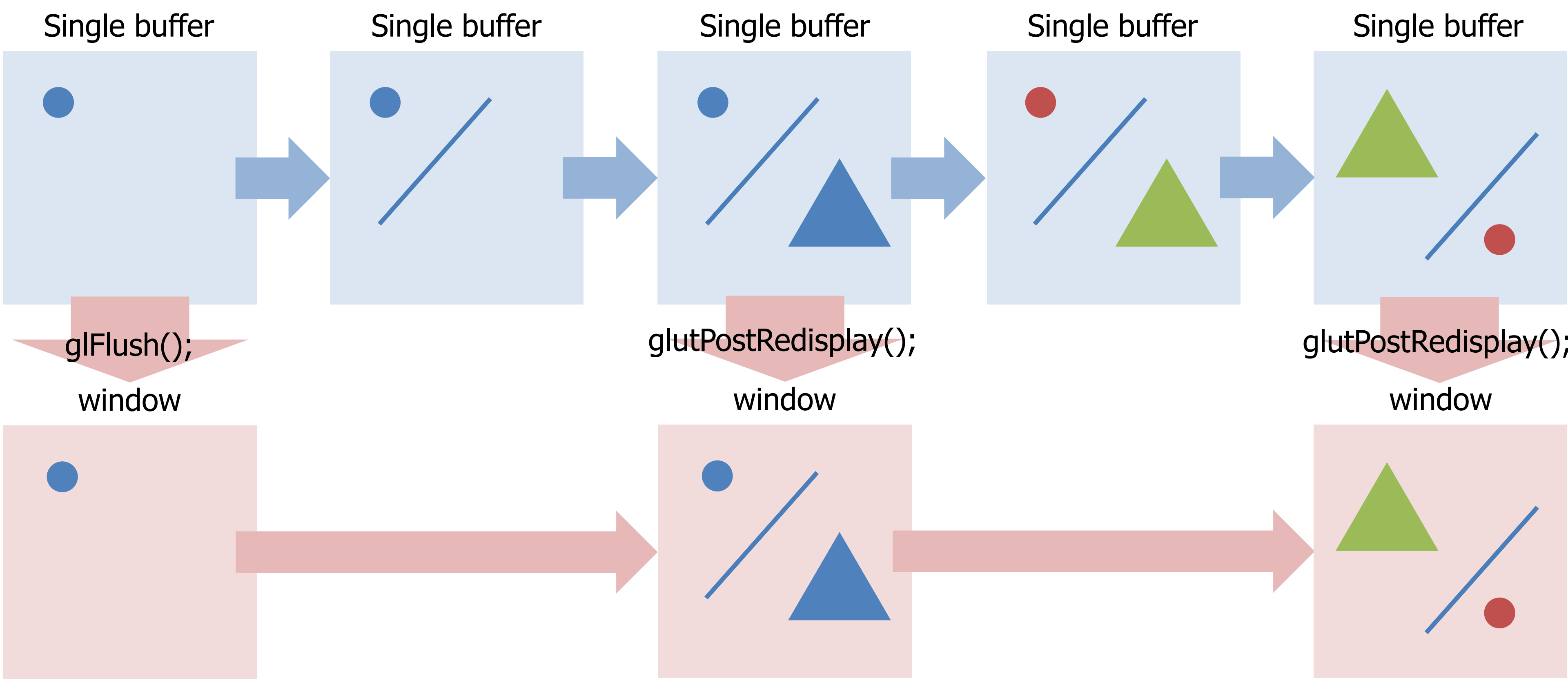


```
void func() {
        …
        glutPostRedisplay();
}
```

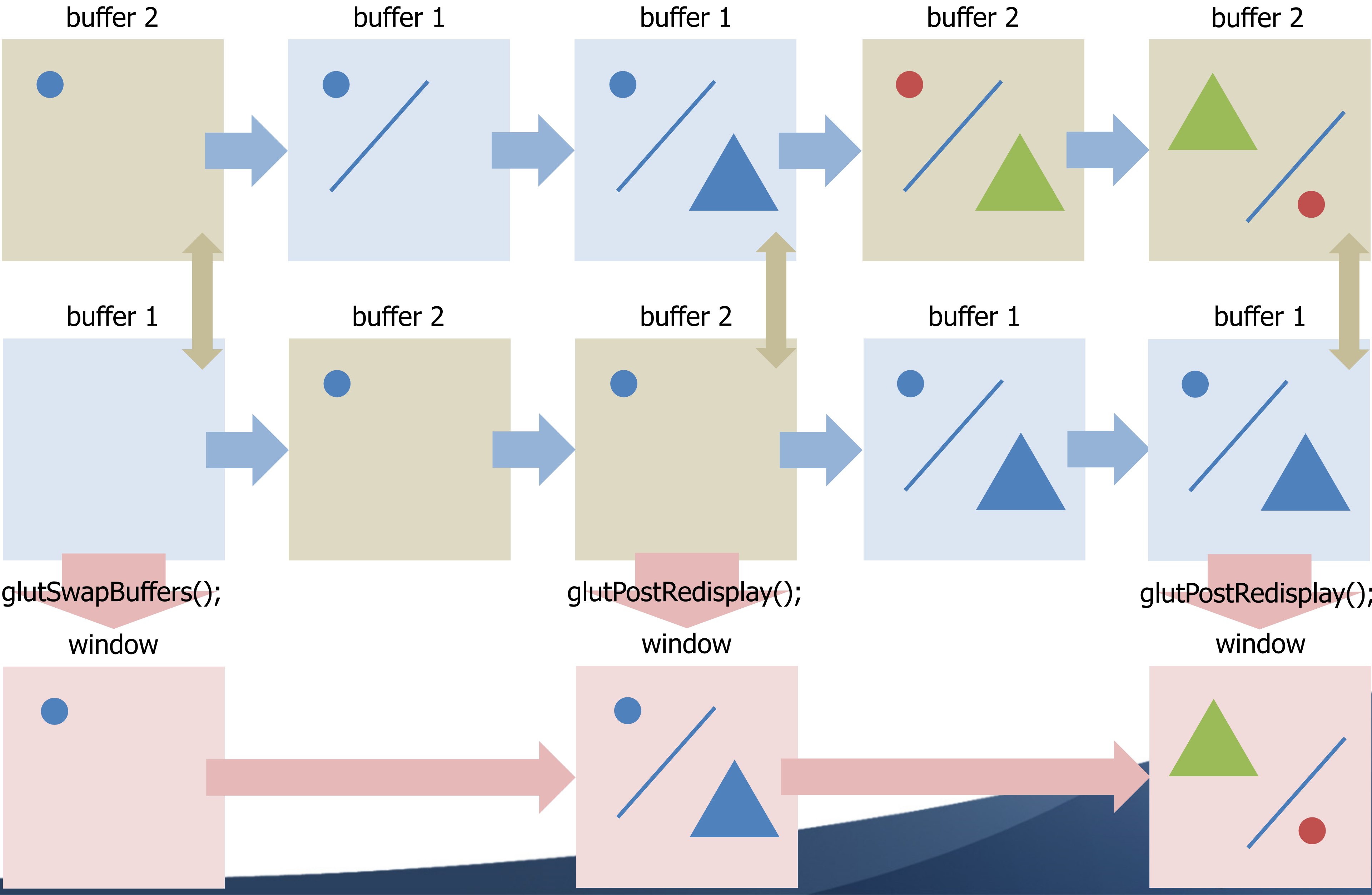
```
void main(int argc, char **argv) {
        …

        // Callback functions
        glutDisplayFunc(renderScene);
        …

        // enter GLUT event processing cycle
        glutMainLoop();
}
```

```
void renderScene(void) {
        glClearColor(0, 0, 0, 0);
        glClear(GL_COLOR_BUFFER_BIT);
        …

        // glFlush();
        // glFinish();
        glutSwapBuffers();
}
```

glutPostRedisplay()



glutPostRedisplay()



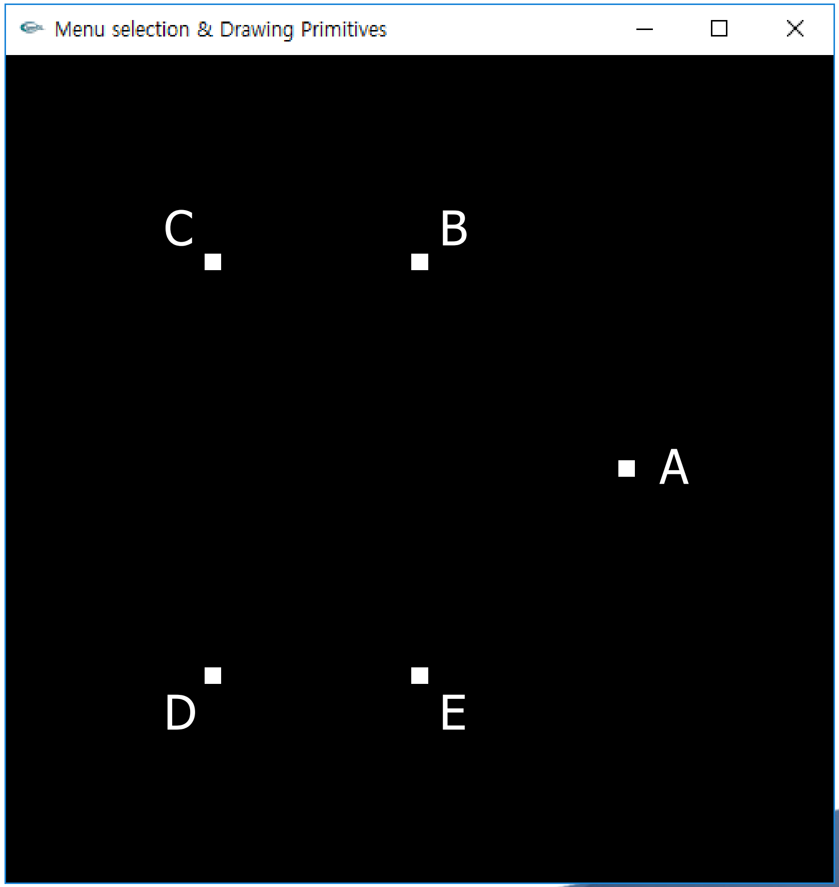
Displaying primitives

- glBegin(GLenum mode)
  - GL_POINTS
  - GL_LINES
  - GL_LINE_STRIP
  - GL_LINE_LOOP
  - GL_TRIANGLES
  - GL_TRIANGLE_STRIP
  - GL_TRIANGLE_FAN
  - GL_QUADS
  - GL_POLYGON
- glEnd()

Point

- glPointSize(GLfloat size)
  - size: must exceed 0.0 (default = 1.0)

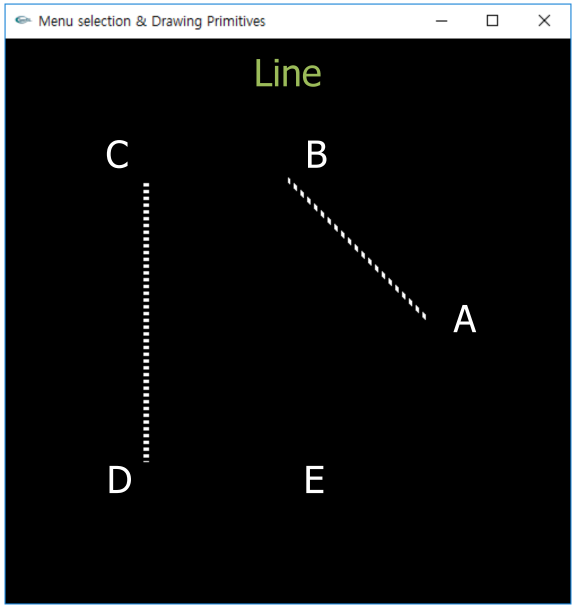
```
void drawPoint() {
	glColor3f(1, 1, 1);
	glPointSize(10.0f);
	glBegin(GL_POINTS);
		glVertex2f(0.5, 0.0);		// A
		glVertex2f(0.0, 0.5);		// B
		glVertex2f(-0.5, 0.5);		// C
		glVertex2f(-0.5, -0.5);	// D
		glVertex2f(0.0, -0.5);		// E
	glEnd();
}
```

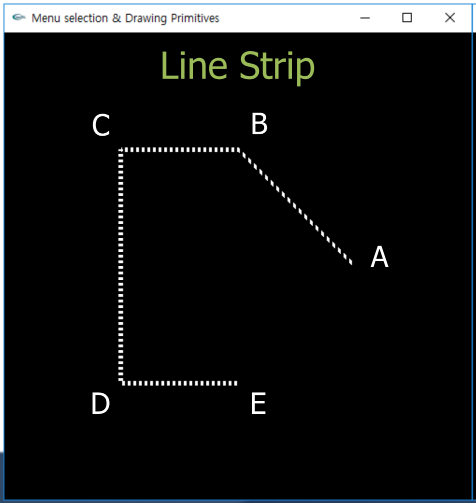


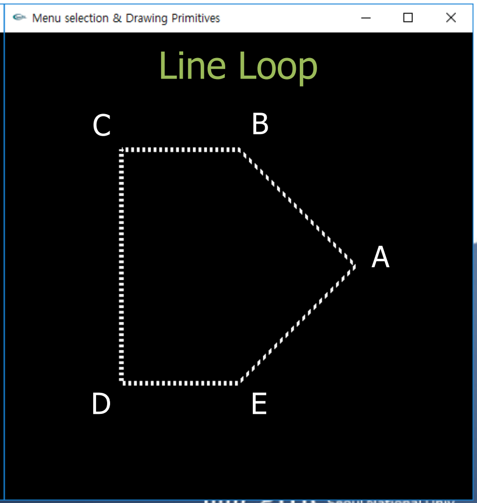
Line

- glLineStipple(GLint factor, GLushort pattern)
- glLineWidth(GLfloat width)
  - width: must exceed 0.0 (default = 1.0)

```
void drawLine() {
    glColor3f(1, 1, 1);
    glLineWidth(5.0f);
    glEnable(GL_LINE_STIPPLE);
    glLineStipple(3, 0xAAAA);
    glBegin(GL_LINES);
    //glBegin(GL_LINE_STRIP)
    //glBegin(GL_LINE_LOOP)
        glVertex2f(0.5, 0.0);       // A
        glVertex2f(0.0, 0.5);       // B
        glVertex2f(-0.5, 0.5);      // C
        glVertex2f(-0.5, -0.5);     // D
        glVertex2f(0.0, -0.5);      // E
    glEnd();
}
```



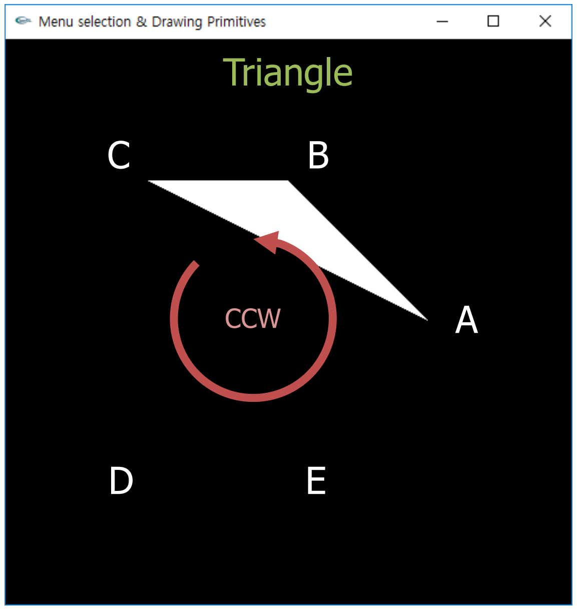


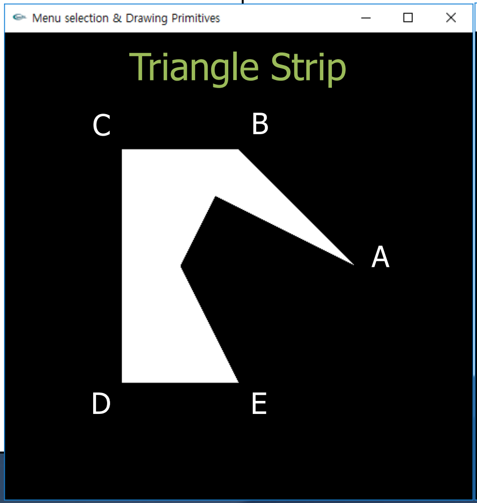


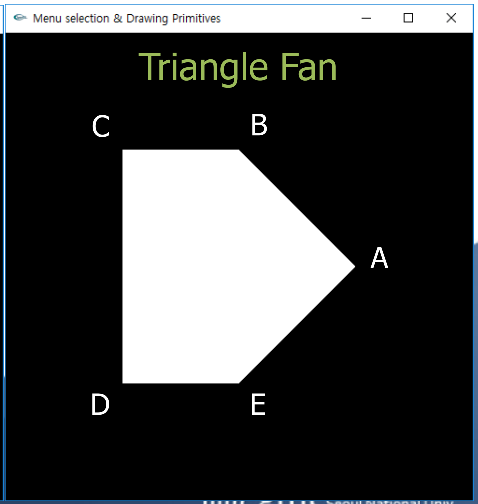
Polygon

- glPolygonMode(GLenum face, GLenum mode)
- glFrontFace(GLenum mode)

```
void drawPolygon() {
    glFrontFace(GL_CCW) ;
    glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
    glColor3f(1, 1, 1);
    glBegin(GL_TRIANGLES);
    //glBegin(GL_TRIANGLE_STRIP);
    //glBegin(GL_TRIANGLE_FAN);
    //glBegin(GL_QUADS);
    //glBegin(GL_POLYGON);
        glVertex2f(0.5, 0.0);       // A
        glVertex2f(0.0, 0.5);       // B
        glVertex2f(-0.5, 0.5);      // C
        glVertex2f(-0.5, -0.5);     // D
        glVertex2f(0.0, -0.5);      // E
    glEnd();
}
```



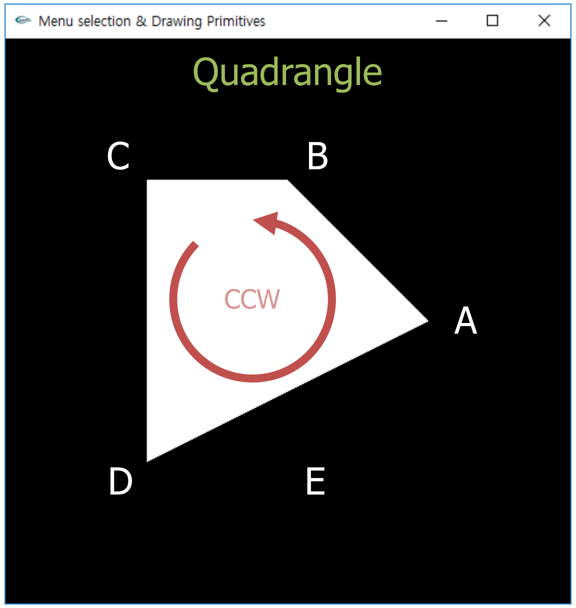


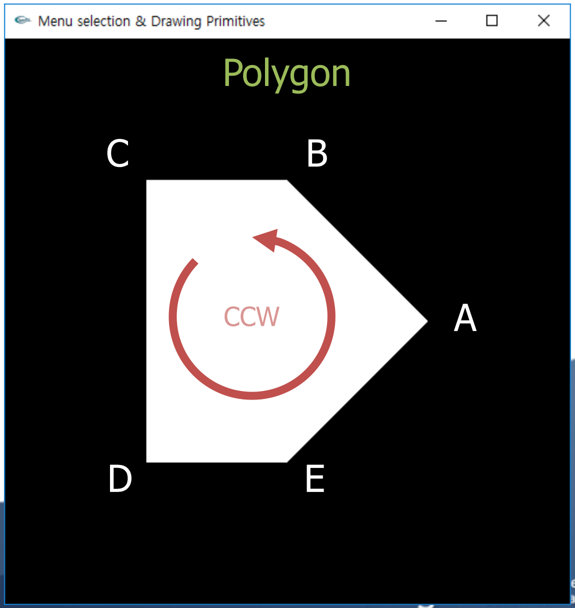


Polygon

- glPolygonMode(GLenum face, GLenum mode)
- glFrontFace(GLenum mode)

```
void drawPolygon() {
    glFrontFace(GL_CCW) ;
    glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
    glColor3f(1, 1, 1);
    //glBegin(GL_TRIANGLES);
    //glBegin(GL_TRIANGLE_STRIP);
    //glBegin(GL_TRIANGLE_FAN);
    glBegin(GL_QUADS);
    //glBegin(GL_POLYGON);
        glVertex2f(0.5, 0.0);          // A
        glVertex2f(0.0, 0.5);          // B
        glVertex2f(-0.5, 0.5);         // C
        glVertex2f(-0.5, -0.5);        // D
        glVertex2f(0.0, -0.5);         // E
    glEnd();
}
```





Polygon

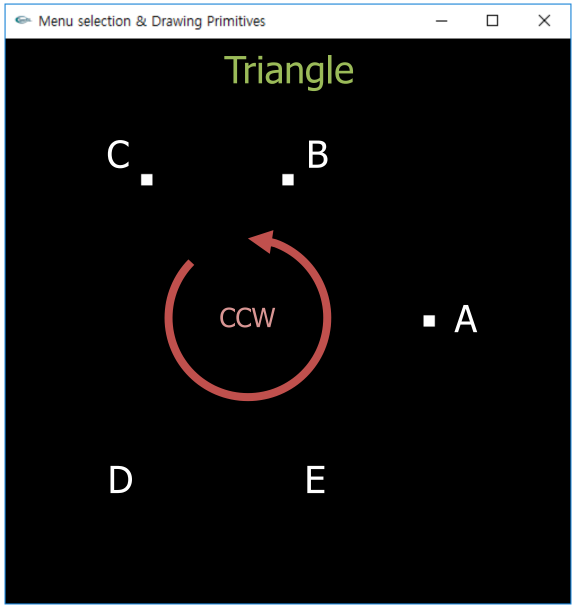
- glPolygonMode(GLenum face, GLenum mode)
  - face
    - GL_FRONT
    - GL_BACK
    - GL_FRONT_AND_BACK
  - mode
    - GL_POINT
    - GL_LINE
    - GL_FILL
  - Default mode: glPolygonMode(GL_FRONT_AND_BACK, GL_FILL)

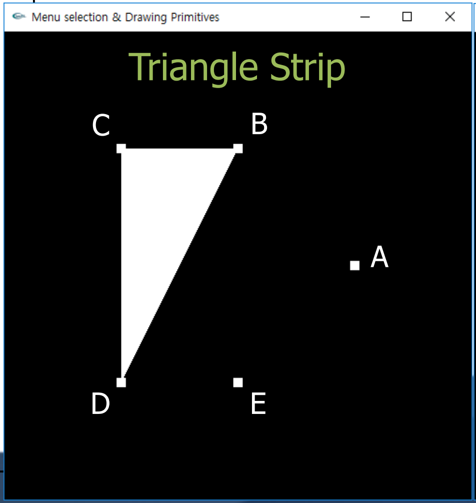
- glFrontFace(GLenum mode)
  - GL_CW
  - GL_CCW (default)

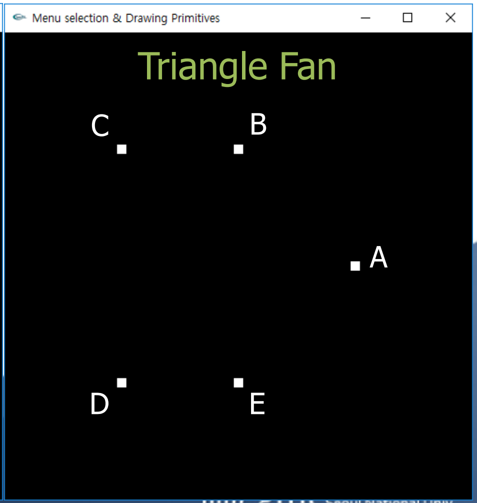
Polygon

- glPolygonMode(GLenum face, GLenum mode)
- glFrontFace(GLenum mode)

```
void drawPolygon() {
    glFrontFace(GL_CCW) ;
    glPolygonMode(GL_FRONT, GL_POINT);
    glPointSize(10.0f);
    glColor3f(1, 1, 1);
    glBegin(GL_TRIANGLES);
    //glBegin(GL_TRIANGLE_STRIP);
    //glBegin(GL_TRIANGLE_FAN);
    //glBegin(GL_QUADS);
    //glBegin(GL_POLYGON);
        glVertex2f(0.5, 0.0);       // A
        glVertex2f(0.0, 0.5);       // B
        glVertex2f(-0.5, 0.5);      // C
        glVertex2f(-0.5, -0.5);     // D
        glVertex2f(0.0, -0.5);      // E
    glEnd();
}
```



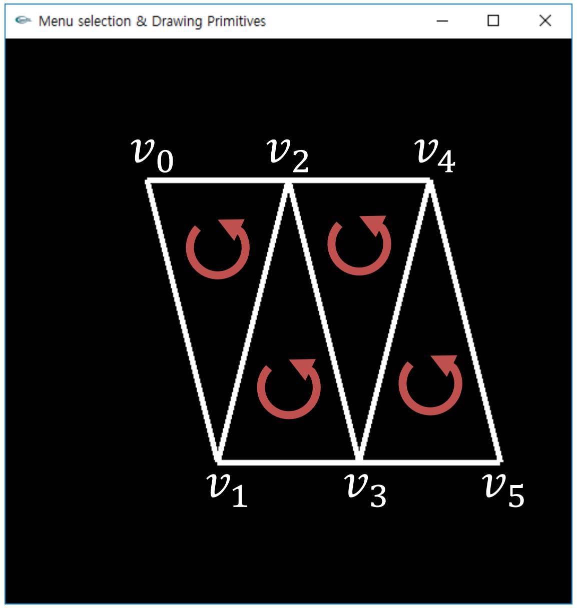




Polygon (Triangle Strip & Fan)

- Triangle Strip

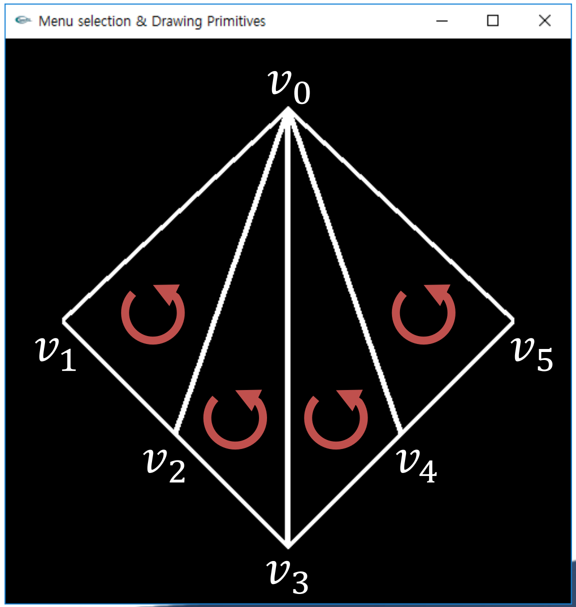
(v_0, v_1, v_2)
$\downarrow$
(v_2, v_1, v_3)
$\downarrow$
(v_2, v_3, v_4)
$\downarrow$
$\vdots$



```
void drawPolygon() {
        glFrontFace(GL_CCW) ;
        glPolygonMode(GL_FRONT, GL_LINE);
        glLineWidth(5.0f);
        glColor3f(1, 1, 1);
        glBegin(GL_TRIANGLE_STRIP);
                glVertex2f(-0.5, 0.5);          // v0
                glVertex2f(-0.25, -0.5);        // v1
                glVertex2f(0.0, 0.5);           // v2
                glVertex2f(0.25, -0.5);         // v3
                glVertex2f(0.5, 0.5);           // v4
                glVertex2f(0.75, -0.5);         // v5
        glEnd();
}
```

- Triangle Fan

(v_0, v_1, v_2)
$\downarrow$
(v_0, v_2, v_3)
$\downarrow$
(v_0, v_3, v_4)
$\downarrow$
$\vdots$

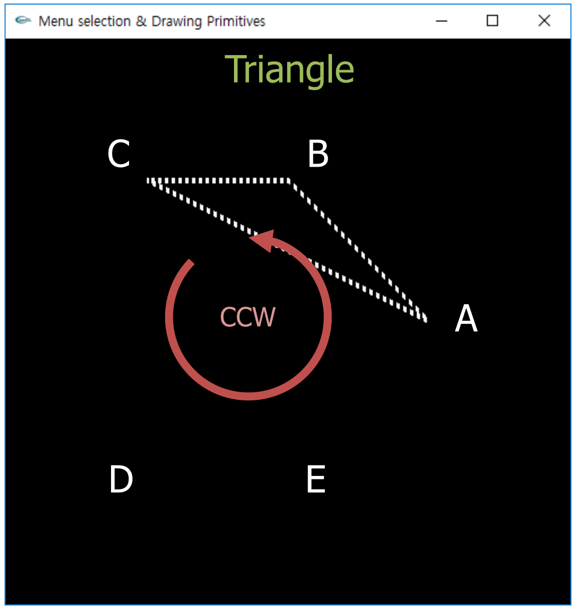


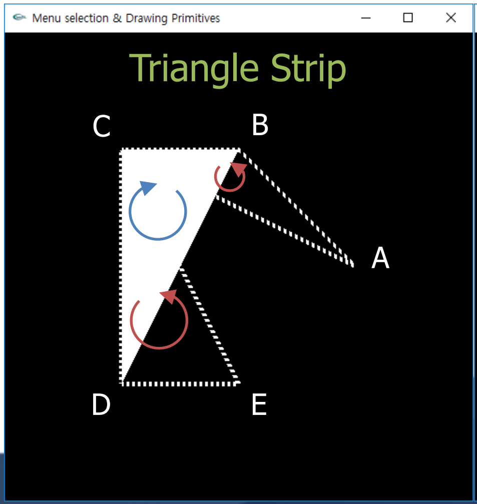
```
void drawPolygon() {
        glFrontFace(GL_CCW) ;
        glPolygonMode(GL_FRONT, GL_LINE);
        glLineWidth(5.0f);
        glColor3f(1, 1, 1);
        glBegin(GL_TRIANGLE_FAN);
                glVertex2f(0.0, 0.75);          // v0
                glVertex2f(-0.8, 0.0);          // v1
                glVertex2f(-0.4, -0.4);         // v2
                glVertex2f(0.0, -0.8);          // v3
                glVertex2f(0.4, -0.4);          // v4
                glVertex2f(0.8, 0.0);           // v5
        glEnd();
}
```

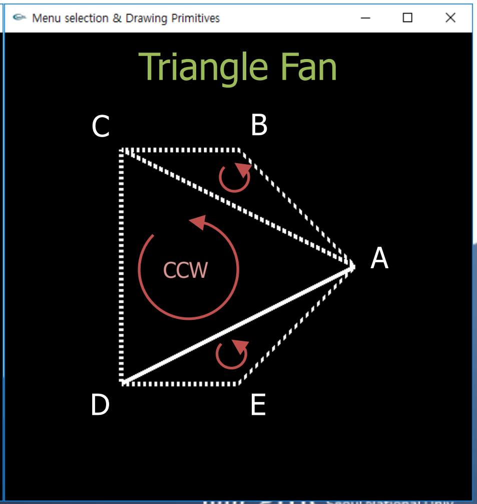
Polygon

- glPolygonMode(GLenum face, GLenum mode)
- glFrontFace(GLenum mode)

```
void drawPolygon() {
    glFrontFace(GL_CCW) ;
    glPolygonMode(GL_FRONT, GL_LINE);
    glLineWidth(5.0f);
    glEnable(GL_LINE_STIPPLE);
    glLineStipple(3, 0xAAAA);
    glColor3f(1, 1, 1);
    glBegin(GL_TRIANGLES);
    //glBegin(GL_TRIANGLE_STRIP);
    //glBegin(GL_TRIANGLE_FAN);
    //glBegin(GL_QUADS);
    //glBegin(GL_POLYGON);
        glVertex2f(0.5, 0.0);       // A
        glVertex2f(0.0, 0.5);       // B
        glVertex2f(-0.5, 0.5);      // C
        glVertex2f(-0.5, -0.5);     // D
        glVertex2f(0.0, -0.5);      // E
    glEnd();
}
```



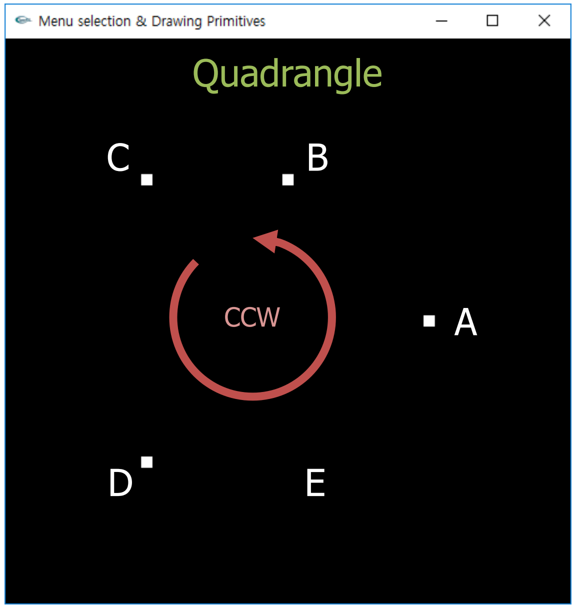


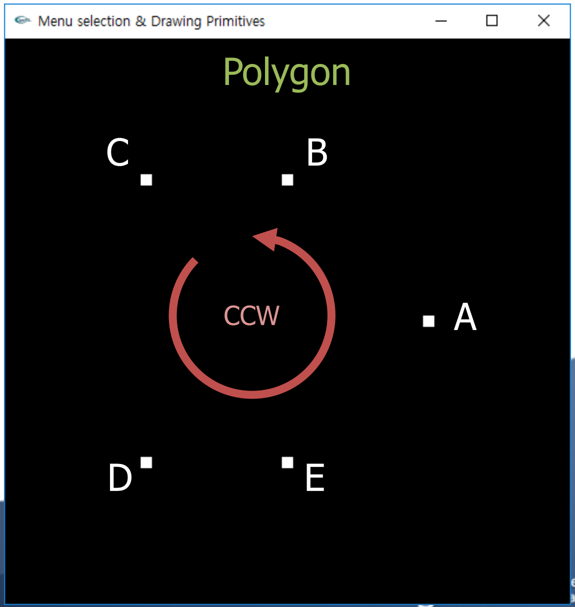


Polygon

- glPolygonMode(GLenum face, GLenum mode)
- glFrontFace(GLenum mode)

```
void drawPolygon() {
    glFrontFace(GL_CCW) ;
    glPolygonMode(GL_FRONT, GL_POINT);
    glPointSize(10.0f);
    glColor3f(1, 1, 1);
    //glBegin(GL_TRIANGLES);
    //glBegin(GL_TRIANGLE_STRIP);
    //glBegin(GL_TRIANGLE_FAN);
    glBegin(GL_QUADS);
    //glBegin(GL_POLYGON);
        glVertex2f(0.5, 0.0);        // A
        glVertex2f(0.0, 0.5);        // B
        glVertex2f(-0.5, 0.5);       // C
        glVertex2f(-0.5, -0.5);      // D
        glVertex2f(0.0, -0.5);       // E
    glEnd();
}
```

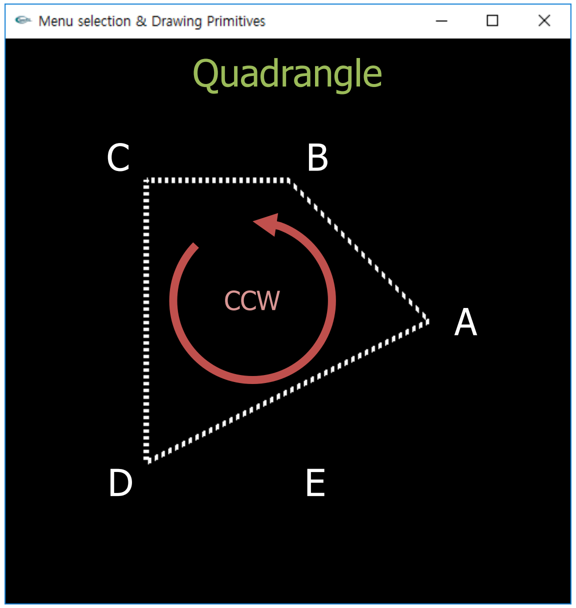


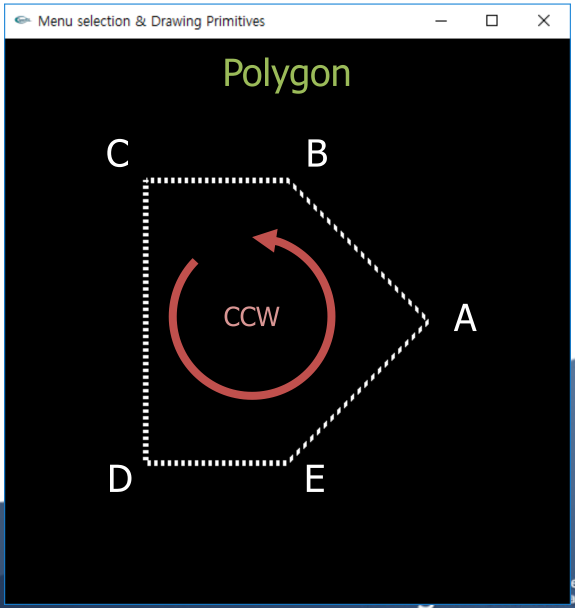


Polygon

- glPolygonMode(GLenum face, GLenum mode)
- glFrontFace(GLenum mode)

```
void drawPolygon() {
      glFrontFace(GL_CCW) ;
      glPolygonMode(GL_FRONT, GL_LINE);
      glLineWidth(5.0f);
      glEnable(GL_LINE_STIPPLE);
      glLineStipple(3, 0xAAAA);
      glColor3f(1, 1, 1);
      //glBegin(GL_TRIANGLES);
      //glBegin(GL_TRIANGLE_STRIP);
      //glBegin(GL_TRIANGLE_FAN);
      glBegin(GL_QUADS);
      //glBegin(GL_POLYGON);
            glVertex2f(0.5, 0.0);           // A
            glVertex2f(0.0, 0.5);           // B
            glVertex2f(-0.5, 0.5);          // C
            glVertex2f(-0.5, -0.5);         // D
            glVertex2f(0.0, -0.5);          // E
      glEnd();
}
```

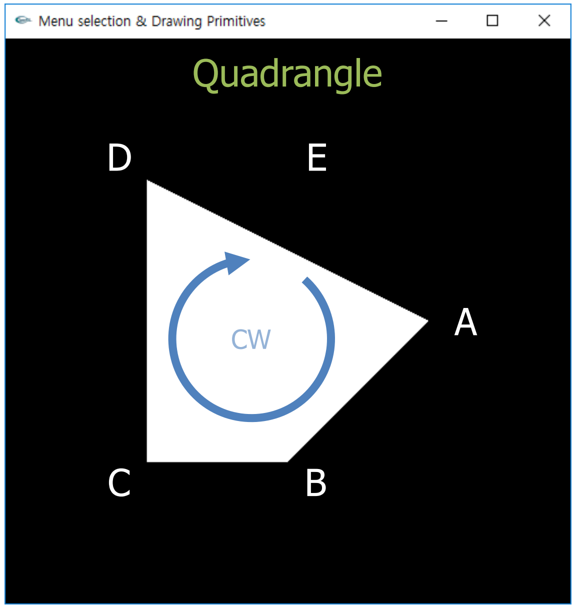


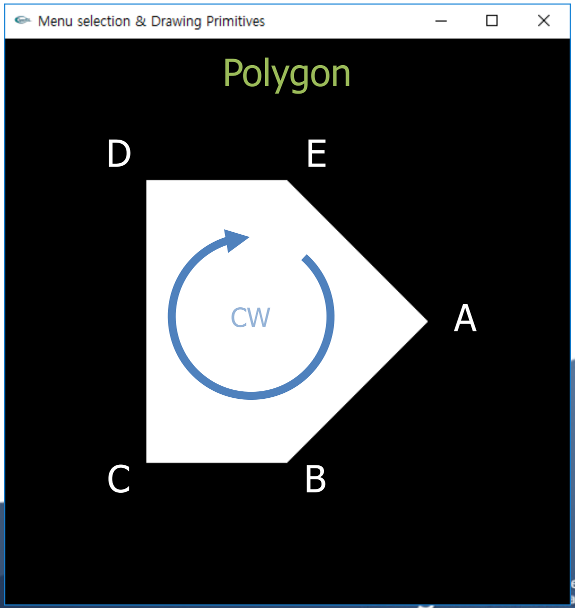


Polygon

- glPolygonMode(GLenum face, GLenum mode)
- glFrontFace(GLenum mode)

```
void drawPolygon() {
        glFrontFace(GL_CCW) ;
        glPolygonMode(GL_FRONT, GL_LINE);
        glLineWidth(5.0f);
        glEnable(GL_LINE_STIPPLE);
        glLineStipple(3, 0xAAAA);
        glColor3f(1, 1, 1);
        //glBegin(GL_TRIANGLES);
        //glBegin(GL_TRIANGLE_STRIP);
        //glBegin(GL_TRIANGLE_FAN);
        glBegin(GL_QUADS);
        //glBegin(GL_POLYGON);
                glVertex2f(0.5, 0.0);           // A
                glVertex2f(0.0, -0.5);          // B
                glVertex2f(-0.5, -0.5);         // C
                glVertex2f(-0.5, 0.5);          // D
                glVertex2f(0.0, 0.5);           // E
        glEnd();
}
```





Callback functions

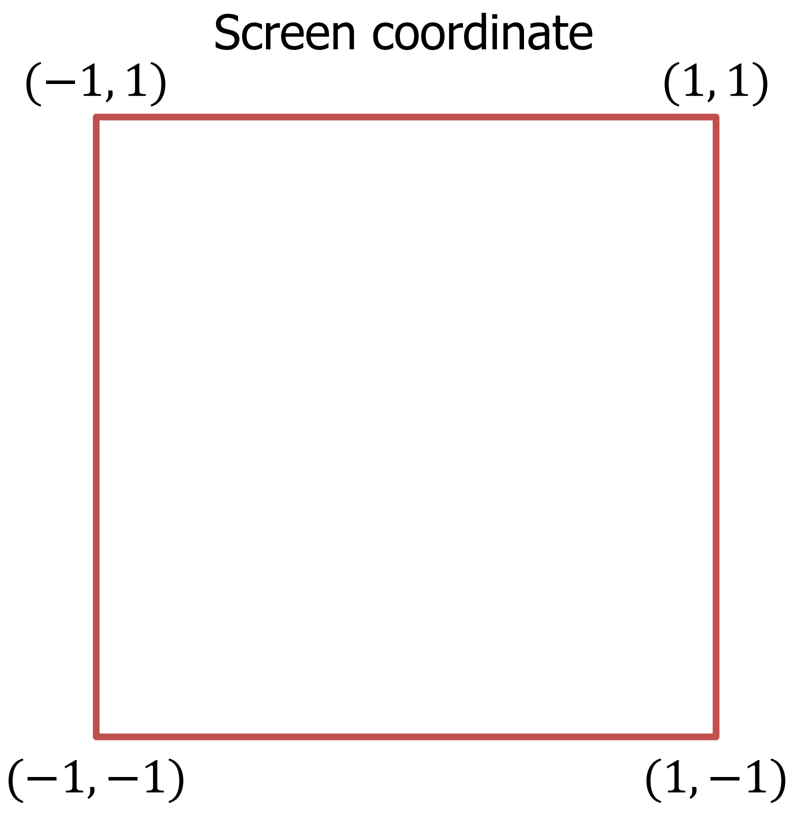
- glutDisplayFunc(…)
- glutKeyboardFunc(…)
- glutSpecialFunc(…)
- glutMouseFunc(…)
- glutMotionFunc(…)
- glutIdleFunc(…)
- glutReshapeFunc(…)

glutMouseFunc(processMouse)

```
void processMouse(int button, int state, int x, int y) {
	printf("Mouse button is clicked! (%d, %d, %d, %d)\n", button, state, x, y);
	if (button == GLUT_LEFT_BUTTON) {
		…
	}
	if (state == GLUT_UP) {
		…
	}
}
```

- button
  - GLUT_LEFT_BUTTON
  - GLUT_MIDDLE_BUTTON
  - GLUT_RIGHT_BUTTON
- state
  - GLUT_DOWN (press)
  - GLUT_UP (release)
- x, y: current mouse position

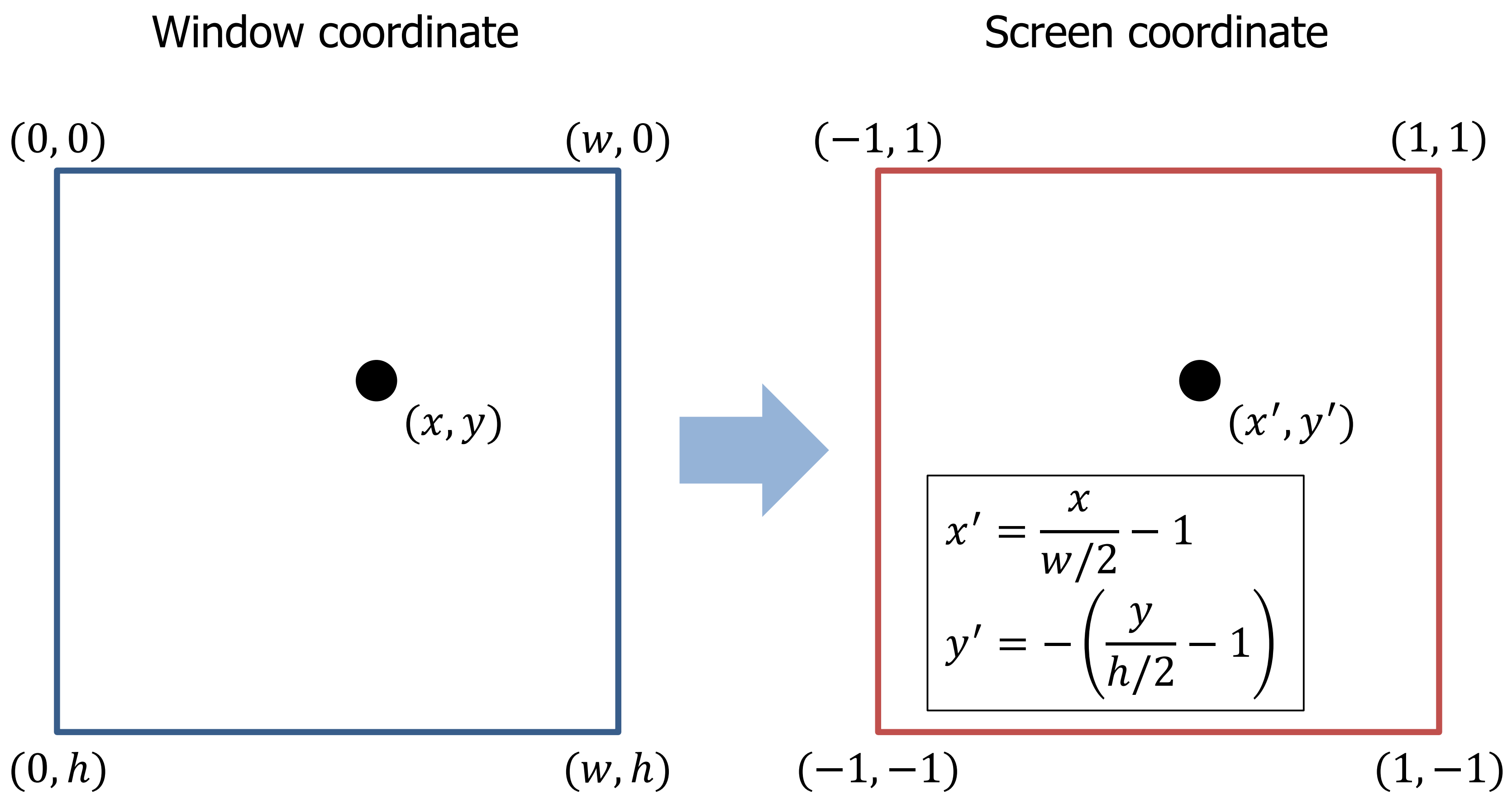
Mapping Coordinate



Default screen coordinate

```
glOrtho(-1.0f, 1.0f, -1.0f, 1.0f, -1.0f, 1.0f)
// gluOrtho2D(-1.0f, 1.0f, -1.0f, 1.0f)
```

Mapping Coordinate

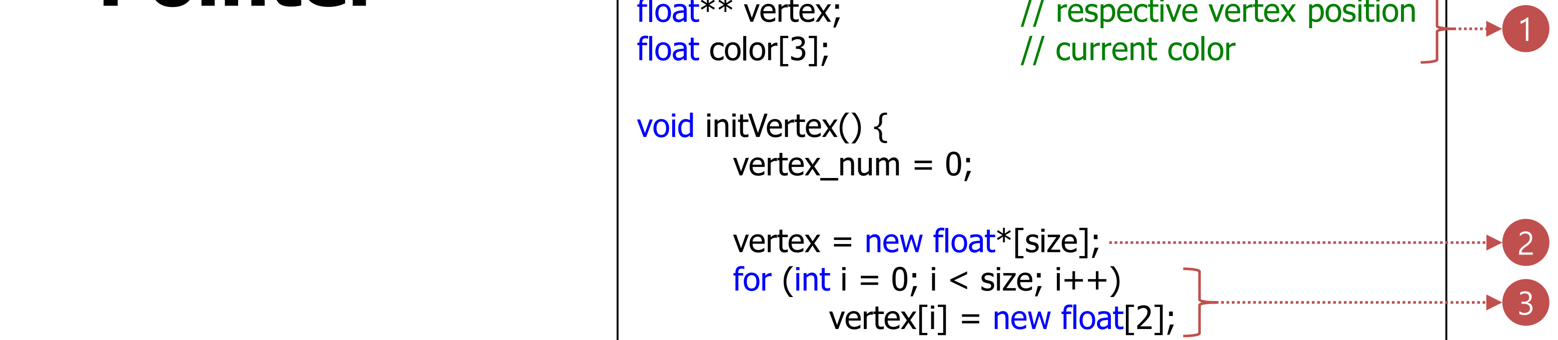


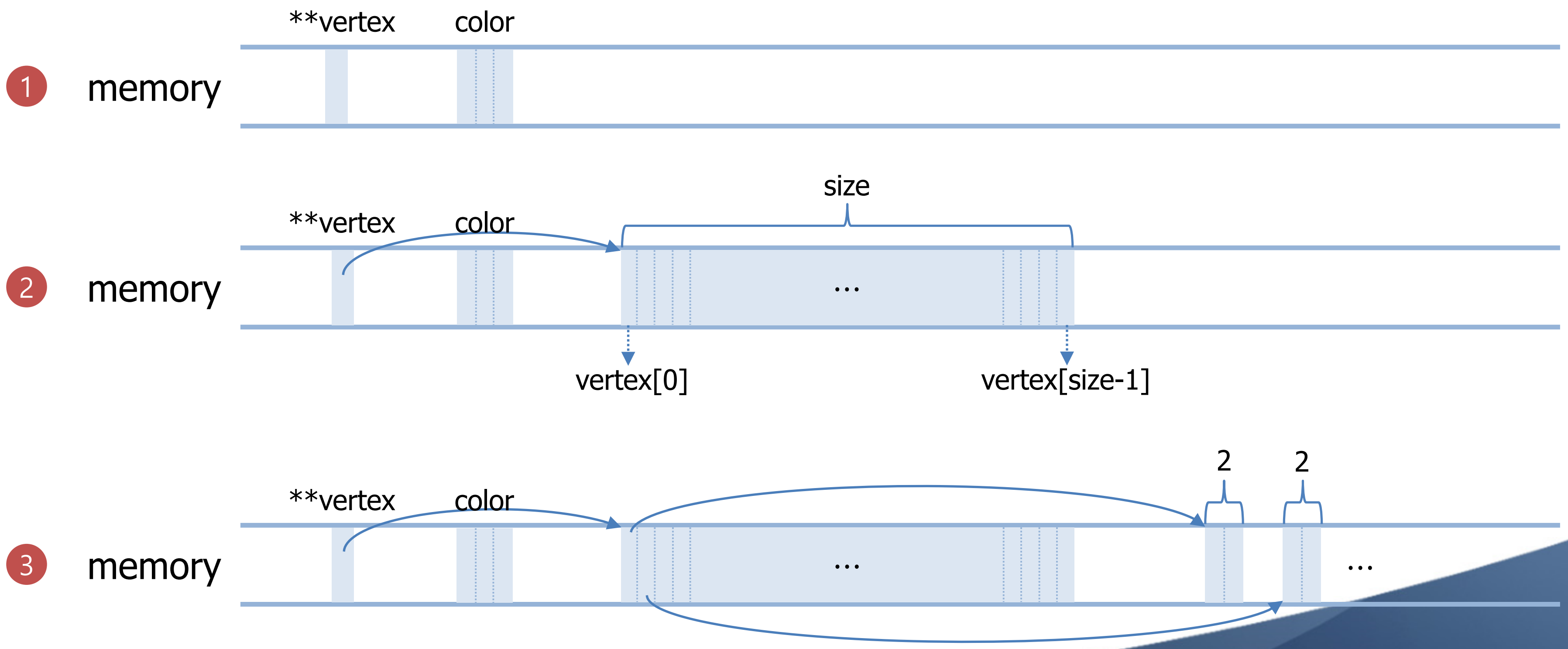
$$x' = \frac{x}{w/2} - 1$$

$$y' = -\left(\frac{y}{h/2} - 1\right)$$

Pointer

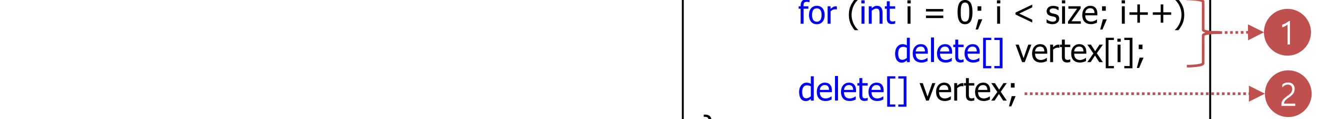
```
const int size = 1000;       // vertex size
int vertex_num;              // number of vertex
float** vertex;              // respective vertex position
float color[3];              // current color

void initVertex() {
        vertex_num = 0;

        vertex = new float*[size];
        for (int i = 0; i < size; i++)
                vertex[i] = new float[2];
}
```

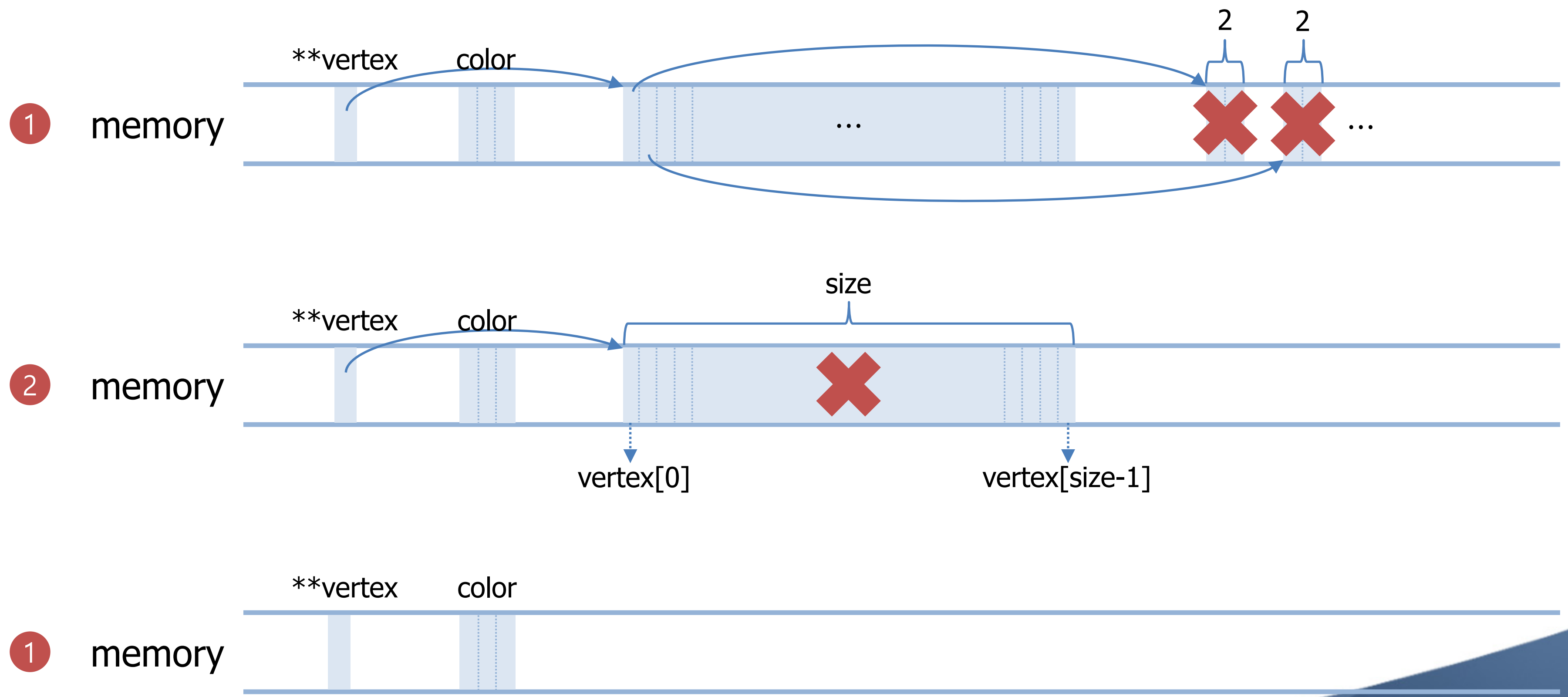




Pointer

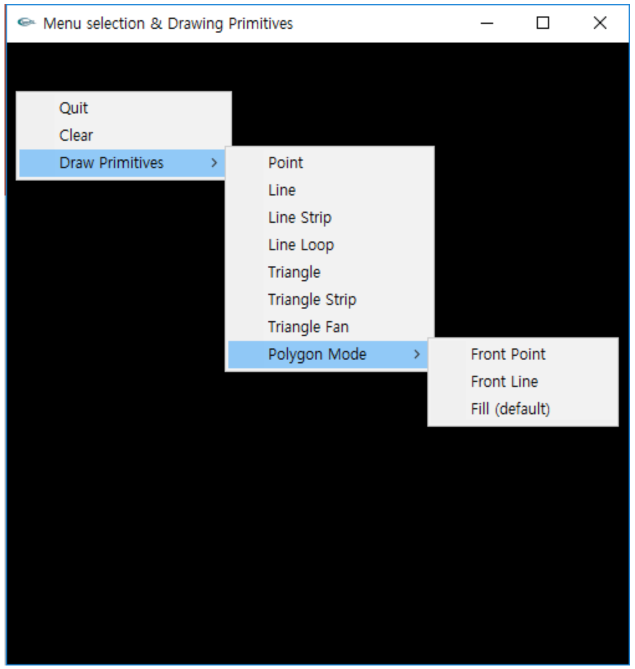
```
void clearVertex() {
        for (int i = 0; i < size; i++)
                delete[] vertex[i];
        delete[] vertex;
}
```

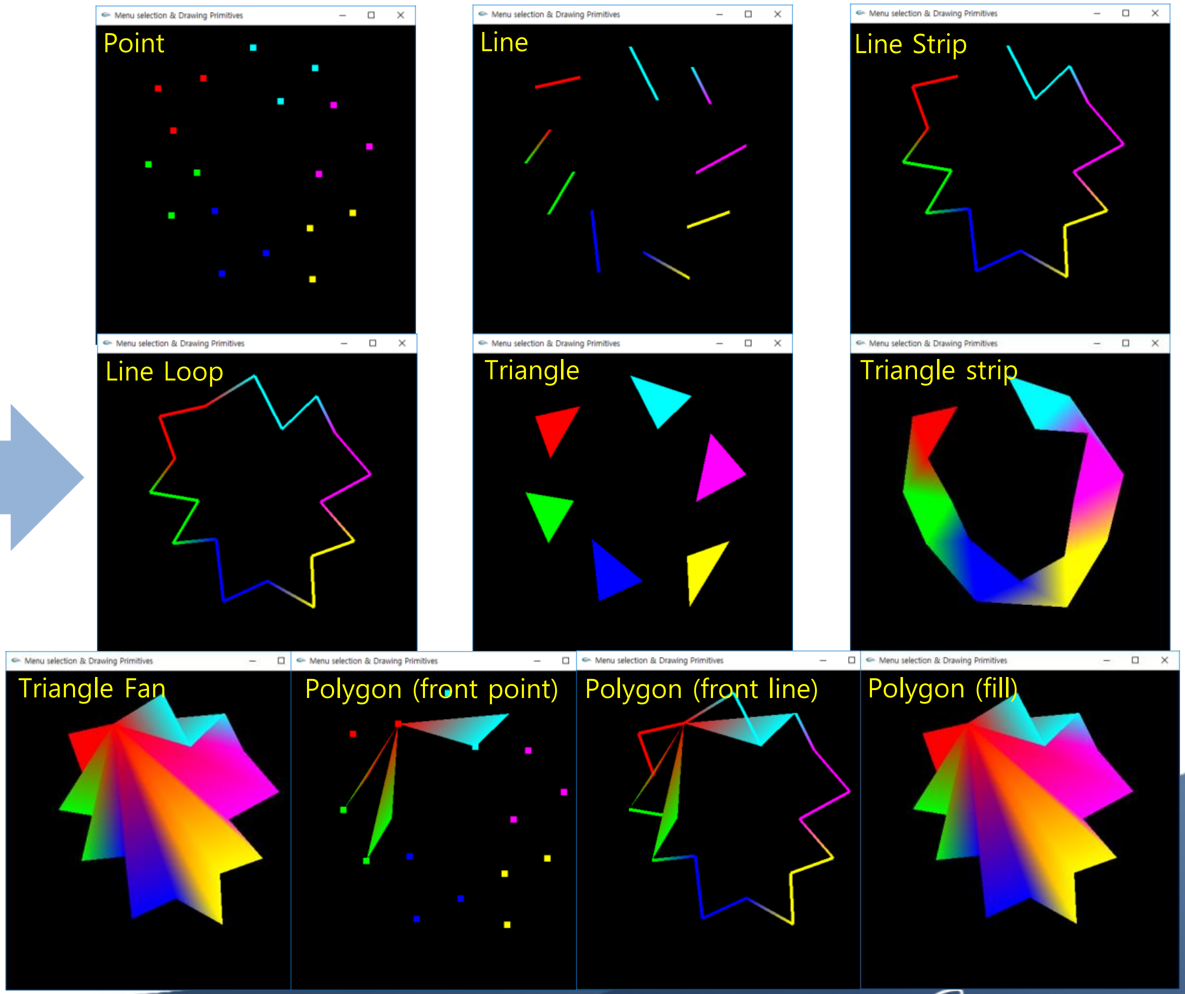




Today's Mission

- Draw primitives using popup menu & pressing left mouse button
- Change vertex color using 1~7 keyboard buttons



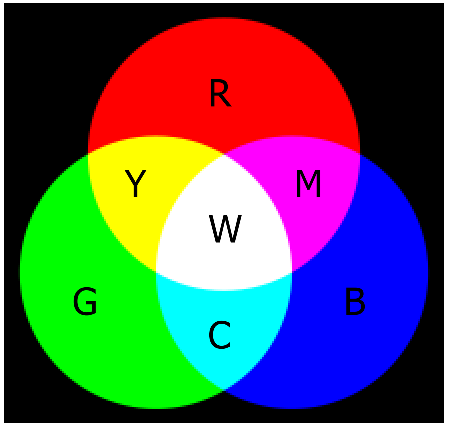


Today's Mission

- Given
  - Window's width (500) & height (500)
  - Size of vertex (1000)
  - Declaration of vertex & vertex_color: double pointer
  - Declaration of current color: array
  - Main_menu_function (quit & clear)
  - Sub_menu_function

Today's Mission

- Implementation
  - Register Callback function
  - Popup menu
    - Pop menu up using right mouse button
    - All menu options must work.
  - Draw Primitives
    - When left mouse button is pressed
    - Point : size (10.0)
    - Line: width (5.0)
  - GLUT Keyboard Input
    - Button 1 ~ 7: Color of vertex to be drawn is set to be R, G, B, Y, M, C, W immediately.
  - GLUT Mouse Input
    - Map 'window coordinate' to 'screen coordinate'



Today's Mission

- Implementation (details)

```
#define WIDTH 500    // window's width
#define HEIGHT 500   // window's height

const int size = 1000;  // vertex size
int vertex_num;         // number of vertex
float** vertex;         // respective vertex position
float** vertex_color;   // respective vertex color
float color[3];         // current color
int menu_number;        // option
```

```
void initVertex() {
        vertex_num = 0;

        /* Implement: Allocate vertex & vertex_color dynamically */
}

void clearVertex() {
        /* Implement: De-allocate vertex & vertex_color */
}
```

Today's Mission

- Implementation (details)
  - Register Callback function
  - Popup menu
    - Pop menu up using right mouse button

```
void main(int argc, char **argv) {
	// init GLUT and create Window
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGBA);
	glutInitWindowPosition(650, 300);
	glutInitWindowSize(WIDTH, HEIGHT);
	glutCreateWindow("Menu selection & Drawing Primitives");
	init();

	/* Implement: Create Popup Menu */

	/* Implement: Register Callback functions */

	// enter GLUT event processing cycle
	glutMainLoop();
}
```

```
void init() {
	initVertex();

	color[0] = color[1] = color[2] = 1;
	menu_number = 0;
}
```

Today's Mission

- Implementation (details)
  - Popup menu
    - Pop menu up using right mouse button

```
void sub_menu_function(int option) {
        printf("Sub menu %d has been selected□n", option);
        menu_number = option;
        /* implement if you need */
}

void main_menu_function(int option) {
        printf("Main menu %d has been selected□n", option);
        if (option == 999) {
                clearVertex();
                exit(0);
        }
        else if (option == 0) {
                clearVertex();
                initVertex();

                glClear(GL_COLOR_BUFFER_BIT);
                glutSwapBuffers();
        }
}
```

Today's Mission

- Implementation (details)
  - Draw Primitives
    - When left mouse button is pressed
    - Point : size (10.0)
    - Line: width (5.0)

```
void drawPoint() {
/* Implement: Display Point */
}

void drawLine() {
/* Implement: Display Line */
}

void drawTriangle() {
/* Implement: Display Triangle */
}

void drawPolygon() {
/* Implement: Display Polygon */
}
```

```
void renderScene(void) {
	glClearColor(0, 0, 0, 0);

	glClear(GL_COLOR_BUFFER_BIT);

	if (menu_number == 1)
		drawPoint();
	else if (menu_number == 2 || menu_number == 3 ||
	menu_number == 4)
		drawLine();
	else if (menu_number == 5 || menu_number == 6 ||
	menu_number == 7)
		drawTriangle();
	else if (menu_number == 8 || menu_number == 9 ||
	menu_number == 10)
		drawPolygon();

	glutSwapBuffers();
}
```

Today's Mission

- Implementation (details)
  - Draw Primitives
    - When left mouse button is pressed
    - Point : size (10.0)
    - Line: width (5.0)
  - GLUT Mouse Input
    - Map 'window coordinate' to 'screen coordinate'

```
void processMouse(int button, int state, int x, int y) {
        printf("Mouse button is clicked! (%d, %d, %d, %d)□n", button, state, x, y);
        /* Implement: Map window coordinate to screen coordinate */
        /* Implement: Store it (its color) into vertex (vertex_color) */
}
```

Today's Mission

- Implementation (details)
  - GLUT Keyboard Input
    - Button 1 ~ 7: Color of vertex to be drawn is set to be R, G, B, Y, M, C, W immediately.

```
void processNormalKeys(unsigned char key, int x, int y) {
	printf("You pressed %c at (%d, %d)□n", key, x, y);
	/* Implement: Change vertex color to be drawn next */
}
```