**기초 컴퓨터 그래픽스**

**HW1 README**

20192138 조명재

**1. [환경 명세]**

1) 본인 프로그램의 실제 구동 환경을 명시 할 것 (OS, CPU, GPU, Compiler 등등)

- OS : window10 64bit

- CPU : Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz

- GPU : intel(R) UHD Graphics 620

- Compiler : visual studio 2022 - win 64

**2. [요구사항] – 직접 실행하며 확인할 예정이니 화면 캡쳐를 올리지 않아도 됩니다.**

(a) 윈도우 화면

- 구현 여부: 예

- 작동 확인 방법: 프로그램 실행 시 화면에 뜬다.

- 구현 방법: 초기화 작업이 진행되고 glutMainLoop() 을 실행하면 윈도우 화면이 뜨면서 여러 작업들을 진행할 수 있는 상태가 된다.

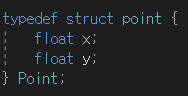
(b) 점 그리기

- 구현 여부: 예

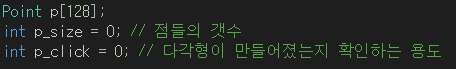
- 작동 확인 방법: shift키를 누르고 왼쪽 마우스 버튼을 클릭하면 파란색 점이 나타난다.

- 기타 요구사항에 명시된 내용: shift 키를 누르지 않은 상태에서는 작동하지 않으며 shift 키를 누른 상태에서만 점이 나타난다.

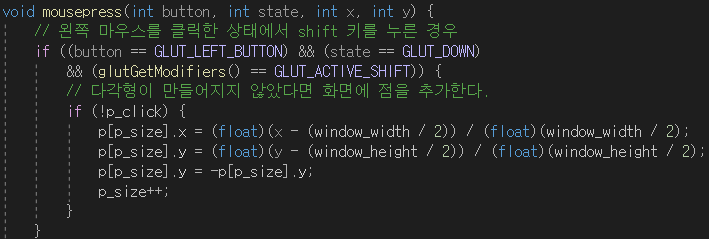
- 구현 방법: 점들을 담기 위한 구조체를 다음과 같이 선언하였다.



그리고 전역변수로 점들을 최대 128개를 담을 수 있도록 하며 점들의 개수, 다각형이 만들어진 여부 등을 알기 위해 다음과 같이 선언하였다.



왼쪽 마우스를 클릭한 상태에서 동시에 shift 키를 누른 경우, 다각형이 만들어지지 않을 경우에 화면에 점을 추가하도록 구현하였다.



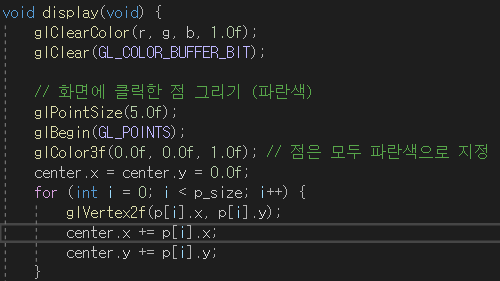
위 식에서 parameter로 받아온 int 형 (x, y)를 -1.0f ~ 1.0f 의 좌표 (x’, y’) 로 변환하기 위해 다음과 같이 식을 세워서 값을 변환시켰다.

x’ = (x - (window\_width / 2)) / (window\_width / 2)

y’ = (y - (window\_height / 2)) / (window\_height / 2)

이때, window\_width, window\_height 의 값은 현재 윈도우의 width, height 값이며 이를 이용하여 -1.0f ~ 1.0f 이내의 값이 만들어진다.

그러면 display() 함수에서 점들의 float 값을 가지고 다음과 같이 출력하도록 할 수 있다.



(c) 현재 점과 이전 점 사이에 선분 그리기

- 구현 여부: 예

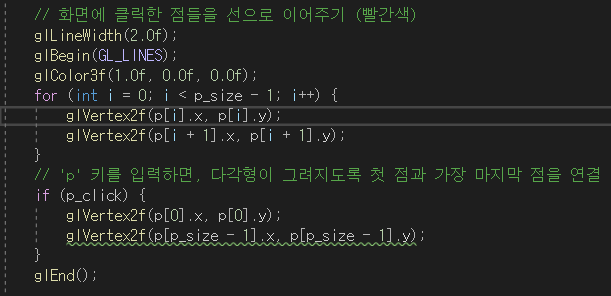
- 작동 확인 방법: shift키를 누르고 왼쪽 마우스 버튼을 클릭하면 파란색 점이 나타나면서 직전 점과 현재 점의 선분이 그려진다.

- 기타 요구사항에 명시된 내용: shift 키를 누르지 않은 상태에서는 작동하지 않으며 shift 키를 누른 상태에서만 선분이 그려진다.

- 구현 방법: 위에서 점들을 그려줬다면, 점과 이전 점과의 선분을 그려줘야 한다.

즉, display() 함수에서 인덱스 0부터 인덱스 p\_size - 2 까지 (0 ~ 1), (1 ~ 2), … , (p\_size - 2 ~ p\_size - 1) 점들 간의 선분을 그려주면 된다.

그리고 선분은 빨간색으로 지정하였다.



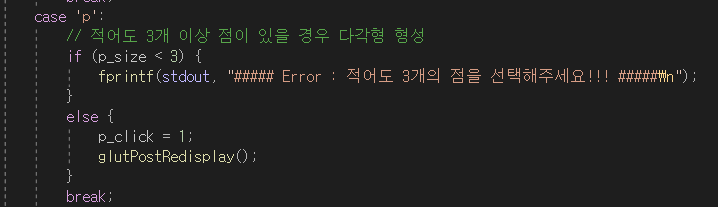
(d) 다각형 완성하기

- 구현 여부: 예

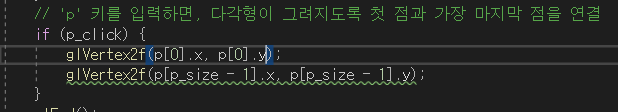
- 작동 확인 방법: ‘p’ 키를 누르면 가장 처음과 마지막에 선택한 점을 연결하는 선분이 그려지며 다각형이 만들어진다.

- 기타 요구사항에 명시된 내용: 최소 3개의 점을 선택한 상태일 때 허용된다.

- 구현 방법: keyboard() 함수에서 ‘p’ 에 대한 케이스에 대해 현재 점의 개수 p\_size 값이 3 이상일 경우 다각형이 만들어진 여부를 확인할 수 있는 변수 p\_click 의 값을 1로 변경시키며 glutPostRedisplay() 함수를 호출한다.



Display() 함수로 다시 넘어가서 p\_click 값이 1인 경우에 다각형이 그려지도록 첫 점과 마지막 점을 연결하도록 조건만 추가하여 선분을 그려주면 된다.



(e) 점과 선분들 또는 다각형 지우기

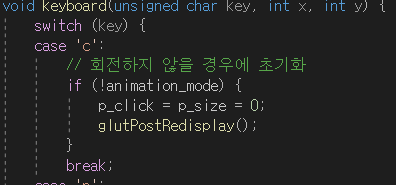
- 구현 여부: 예

- 작동 확인 방법: ‘c’ 키를 누르면 선택한 점, 선분들 또는 다각형이 모두 지워진다.

- 구현 방법: 우선 전역변수로 회전하는지 안하는지에 대해 판단하기 위한 변수를 다음과 같이 작성하였다.



keyboard() 함수에서 ‘c’ 에 대한 케이스에 대해 회전하지 않을 경우에 초기화가 이뤄져야 한다. (다각형이 아니도록 변경 및 점의 개수 0 초기화)

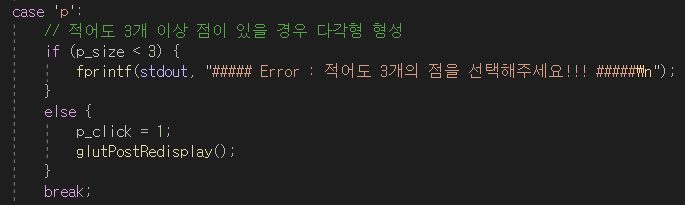


(f) 적절한 에러 메시지 출력

- 구현 여부: 예

- 작동 확인 방법: ‘p’ 키를 누를 때 최소 3개의 점을 선택하지 않은 경우 에러메시지가 뜬다.

- 구현 방법: keyboard() 함수에 ‘p’ 키를 누를 때 점의 개수가 3 미만일 경우 에러 문구를 출력하도록 하였다.

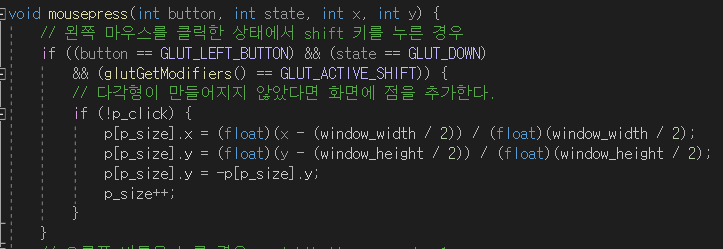


(g) 다각형을 완성하면 더 이상 점 및 선분을 그리지 않기

- 구현 여부: 예

- 작동 확인 방법: SHIFT 키를 누른 상태에서 왼쪽 마우스를 클릭하면 반응하지 않고 다각형이 완성된 상태일 때 해당한다.

- 구현 방법: mousepress 함수에서 다각형인지 판단하는 변수 p\_click 즉, p\_click 값이 1일 경우에는 반응시키지 않고 오직 다각형이 아닌 경우 p\_click 값이 0 인 경우에만 점들의 위치를 담으면서 점의 개수를 증가시키도록 구현하였다.



(h) 방향키를 누르면 적절한 양 만큼 해당 방향으로 다각형이 이동

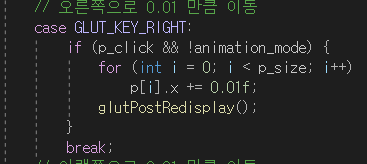
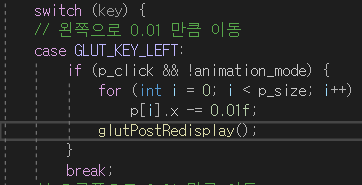
- 구현 여부: 예

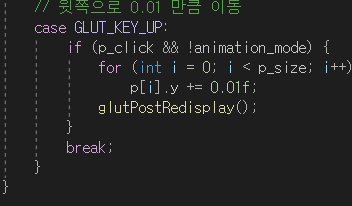
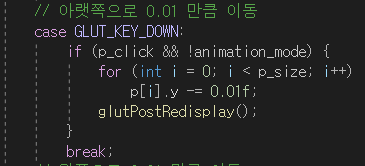
- 작동 확인 방법: 방향키를 누르면 다각형이 이동한다.

- 기타 요구사항에 명시된 내용: 다각형이 완성되지 않은 상태에서는 작동되지 않는다.

- 구현 방법: special() 함수에서 각 방향키별로 다각형이 만들어졌는지에 대한 여부와 회전하고 있지 않은 상태일 때 모든 좌표들을 0.01f 만큼씩 증가시키거나 감소시키도록 구현하였다.

그리고 좌표들을 전부 변화시키면 glutPostRedisplay() 를 호출하여 적용되도록 하였다.





(i) 오른쪽 마우스 버턴을 클릭하여 다각형 이동시키기

- 구현 여부: 예

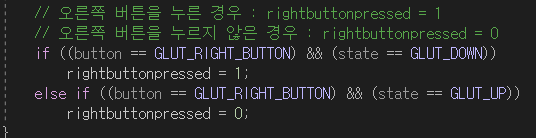
- 작동 확인 방법: 오른쪽 마우스 버튼을 클릭하여 다각혀잉 만들어진 경우에만 마우스가 움직인 양 만큼 해당 방향으로 이동한다.

- 기타 요구사항에 명시된 내용: 다각형이 완성되지 않은 상태에서는 이동시킬 수 없다.

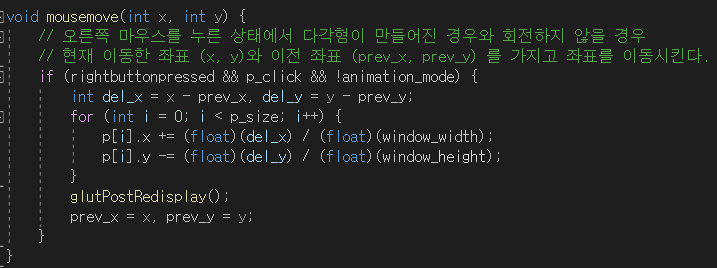
- 구현 방법: 오른쪽 마우스 버튼을 누르고 있는지를 확인하기 위한 전역변수를 다음과 같이 선언하였다.



그리고 오른쪽 마우스 버튼을 누르고 있는 경우 mousepress 함수에서 다음과 같이 rightbuttonpressed 값을 1로 선언하고 버튼을 땔 경우에는 0 으로 선언하였다.



이제 mousemove 함수에서 오른쪽 버튼이 눌린 상태에서 다각형이 만들어진 경우와 회전하고 있지 않은 이 모든 조건이 충족될 경우에 parameter로 받아온 (x, y) 즉, 현재 좌표에서 이전 좌표를 뺀 값에 대해서 그 값을 window 의 width, height 의 값으로 나눠서 현재 점들의 모든 좌표들ㄹ에 더해줘서 glutPostRedisplay() 를 호출하여 다시 화면에 그려지도록 하였다.



(j) 무게 중심 점 둘레로 다각형을 반시계 방향으로 회전시키기

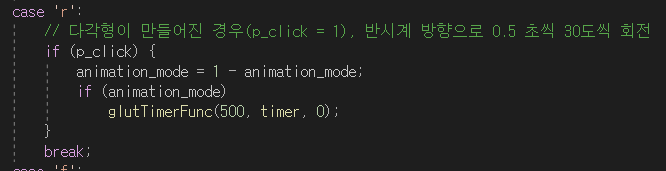
- 구현 여부: 예

- 작동 확인 방법: ‘r’ 키를 누르면 다각형은 자신의 무게 중심 점 둘레로 반시계 방향으로 적절한 속도로 회전하게 된다.

- 기타 요구사항에 명시된 내용: 다각형이 완성되지 않은 상태에서는 작동하지 않는다.

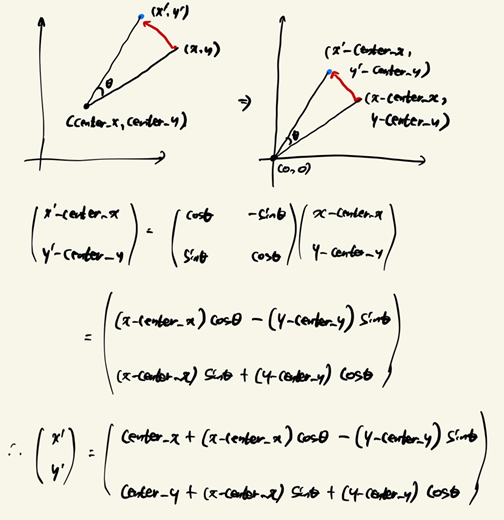
- 구현 방법: 교수님께서 올려주신 코드를 잘 참고하여 ‘r’ 키를 눌렀을 때 다각형인 경우에 animation\_mode 라는 전역 변수를 이용하여 값을 0에서 1로 변경시킬 경우 반시계 방향으로 0.5 초씩 30도씩 회전시키며 1에서 0으로 변경시킬 경우 회전을 중단하도록 구현하였다.

우선 ‘r’ 키를 눌렀을 때의 반응하는 코드는 다음과 같다.



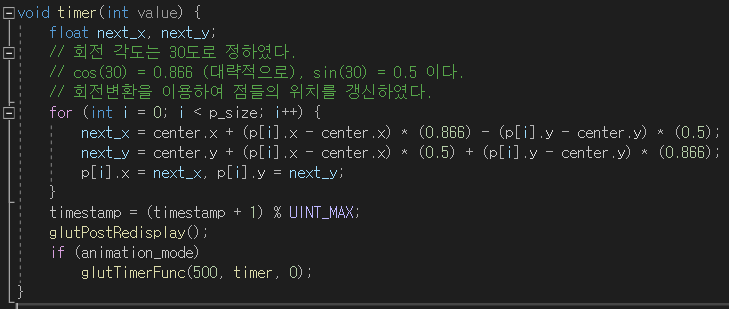
그리고 좌표들을 0.5 초씩 마다 변경시키기 위해 timer() 함수에서 점들의 좌표값들을 변경시켜주었다.

좌표값들의 변경을 위한 식에 대한 이해가 필요하기 때문에 아래 사진을 먼저 참고하신 후에 코드를 보시면 됩니다.



무게 중심 좌표를 (center\_x, center\_y), 현재 좌표를 (x, y) 라고 할 때, 반시계 방향으로 세타만큼 회전할 때 좌표를 (x’, y’) 라고 하면 위와 같은 식이 세팅된다.

세타에 대한 결정은 30 도로 결정하였으며 cos(30도) = 0.866 으로 대략적으로 잡았으며 sin(30) = 0.5 으로 식을 세팅하게 되면 다음과 같은 식이 나타나게 된다.



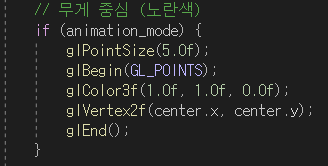
(k) 회전 중에 다각형의 무게 중심에 해당하는 점 그리기

- 구현 여부: 예

- 작동 확인 방법: ‘r’ 키를 누르면 다각형의 무게 중심에 해당하는 점이 그려진다.

- 구현 방법: ‘r’ 키를 누르면 animation\_mode 값이 0에서 1이 된 경우 display() 함수에서 animation\_mode 값이 1일 경우에 무게 중심을 그려주면 된다.

이때 무게 중심은 노란색으로 하였다.



(l) 회전을 멈춤과 동시에 중심 점 사라지게 하기

- 구현 여부: 예

- 작동 확인 방법: ‘r’ 키를 다시 한번 더 누르면 회전이 멈추고 중심 점이 사라진다.

- 구현 방법: ‘r’ 키를 다시 한번 더 눌러서 animation\_mode 값이 1에서 0으로 된 경우 display 함수에서 무게 중심을 그려주지 않고 회전을 중단시키도록 해야 한다.

즉, animation\_mode 값이 0일 경우에 위와 같은 operation 을 중단시키면 된다.

(m) 회전하는 동안 어떠한 operation 들을 모두 정지시키기

- 구현 여부: 예

- 작동 확인 방법: ‘r’ 키를 눌러 회전하는 동안 ‘c’ 키를 누르거나 shift 키를 눌러서 새로운 점을 선택하거나 화살표를 누를 시 이러한 작업이 무시된다.

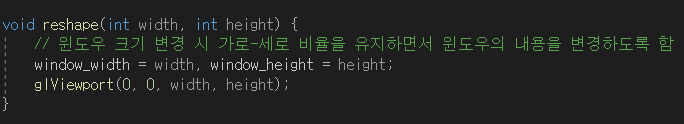
- 구현 방법: 다른 operation 을 할 때 animation\_mode 값이 1일 경우 무시가 되도록 구현하면 된다.

(n) 윈도우 크기 변경시 가로-세로 비율을 유지하면서 윈도우 내용 변경시키기

- 구현 여부: 예

- 작동 확인 방법: 윈도우 사이즈를 변경시키면 된다.

- 구현 방법: reshape 함수에서 window\_width, window\_height 값을 parameter로 받아온 width, height 값으로 변경된 값을 할당해주면서 glViewport 함수를 이용하여 가로-세로 비율을 유지하면서 윈도우 내용을 변경시킬 수 있다.

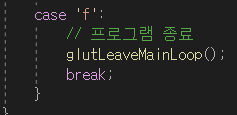


(o) f 키를 누르면 프로그램이 종료

- 구현 여부: 예

- 작동 확인 방법: ‘f’ 키를 누른다.

- 구현 방법: keyboard() 함수에서 ‘f’ 에 대한 케이스에 대해서 glutLeaveMainLoop() 함수를 호출하여 프로그램을 종료시켰다.



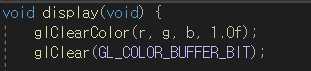
(p) 윈도우 배경, 선택한 점, 점들을 연결하는 선분, 다각형의 무게 중심 점 전부 다른 색깔로 그리기

- 구현 여부: 예

- 작동 확인 방법: 다각형을 만들어서 ‘r’ 키를 누른다.

- 구현 방법: 윈도우 배경색은 전역 변수로 r, g, b 값을 선언하여 display 함수에서 glClearColor 함수를 통해 초록색으로 나타내도록 하였다.





점의 색깔은 파란색, 선분의 색은 빨간색, 그리고 무게 중심은 노란색으로 glColor3f() 함수를 이용하여 그에 맞은 rgb 값을 인자로 넣어줌으로써 구현하였다.