



---

# R E P O R T

실습과제 06

# 내용

과제 명세  
로봇 조립하기  
고찰 및 느낀점

## 과제 명세

### 실습과제 06



- 기간: 2주
- 배점: 10+2+2점
- 강의자료 5장의 로봇 조립 프로그램 구현하기
  - 모델 변환을 이용해 로봇을 조립하자.
  - 동영상 참고: 그래픽스-05장-6 동영상
  - 동영상에서와 같은 기본 기능 구현
  - 자신이 만들거나 수정한 로봇을 이용하는 것을 권장함.
    - 기본 로봇 모델(수업에서 제공) 사용 → 10점(기본)
    - 기본 로봇을 간략히 수정(3DMax이용)하여 사용한 경우 → +2점
    - 완전히 새로운 멋진 모델을 만들어 사용한 경우 → **최대** +2점
  - 로봇을 부품으로 나누는 형태는 기본 모델과 동일하게 처리할 것: 몸통, 머리, 팔/다리의 윗부분과 아랫부분 등

# 로봇 조립하기

## - 파일 인코딩 및 화면 초기화 관련

### 1. glhc.h

<b>glhc.cpp</b>
// 문자열을 화면에 그림 extern void glString(const char* s); // 재질의 색상 설정 : 객체에 따라 결정 extern void glColor(float r, float g, float b, float a); // 좌표계 그리기 : 원점에서 +x(red), +y(blue), +z(gray)로 길이 len인 선분 extern void glDrawCoord(double len);
<b>glWin.cpp</b>
// ase 파일 불러오기 extern char* glFileDlg(const char*);

### 2. glhc.cpp

void glString(const char* s) // 문자열을 화면에 그림
unsigned int i; for (i = 0; i < strlen(s); i++) glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, s[i]);
// 재질의 색상 설정 : 객체에 따라 결정
void glColor(float r, float g, float b, float a) float color[4] = { r, g, b, a }; glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, color); glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, color);
// 좌표계 그리기 : 원점에서 +x(red), +y(blue), +z(gray)로 길이 len인 선분
void glDrawCoord(double len) // 조명 사용 설정 glDisable(GL_LIGHTING); // x, y, z축 그리는 부분(하단 참조) glEnable(GL_LIGHTING);  ----- glLine(0, 0, 0, len, 0, 0); // RED ==> +x axis glLine(0, 0, 0, 0, len, 0); // GREEN ==> +y axis glLine(0, 0, 0, 0, 0, -len); // BLUE ==> +z axis  glLine() 메서드로 축의 길이를 설정하고 glColor3f() 메서드로 색상을 선택하고 glRasterPos3f() 메서드로 현재 래스터의 좌표를 설정하고 glString() 메서드로 축의 이름을 설정합니다.

### 3. glWin.cpp

```
#include <windows.h> // 오류를 막기 위한 헤더
// ase 파일 불러오기
char* glFileDlg(const char* filter)
{
    const int file_name_len = 1024;
    static char fileName[file_name_len] = "";
    OPENFILENAME open_file;
    memset(&open_file, 0, sizeof(OPENFILENAME)); // memset : 메모리 복사 함수. strcpy와 유사함
    open_file.lStructSize = sizeof(OPENFILENAME); // OPENFILENAME 타입으로 크기 지정
    open_file.hwndOwner = NULL;
    open_file.lpstrFilter = L"ASE 3D data (*.ase)W0*.aseW0All(*.*)W0*.W0"; // 인코딩 방식 지정
    open_file.nFilterIndex = 1; // 1칸씩 읽기
    bool ret = GetOpenFileName(&open_file); // 파일 오픈 여부 확인
    return (ret) ? fileName : NULL; // 이상시 NULL 반환
}
```

## - 로봇 조립 관련

### 4. mesh.h

<b>mesh 클래스 구현을 위한 구조체 모음</b>	
<pre> struct Vertex { // Vertex: x, y, z의 위치값     float x, y, z; };  struct Face { // Face: vertex 3개의 인덱스     int vi[3]; };  struct Normal { // Normal: face와 세 정점의 법선벡터     float norFace[3];     float norV1[3], norV2[3], norV3[3]; }; </pre>	
<b>mesh 클래스</b>	
<div> <div>필드</div> <div> <div>face</div> <div>nFace</div> <div>normal</div> <div>nVtx</div> <div>vertex</div> </div> </div>	<div> <div>메서드</div> <div> <div>~Mesh</div> <div>clearAse</div> <div>draw(+ 1개 오...</div> <div>Mesh</div> <div>readAse</div> <div>setColor</div> </div> </div>
<b>필드</b>	
<pre> int nVtx = 0; // Vertex 개수 Vertex* vertex = NULL; // Vertex 리스트 int nFace = 0; // Face 개수 Face* face = NULL; // Face 리스트 Normal* normal = NULL; // Normal 리스트 </pre>	
<b>메서드</b>	
<pre> void clearAse() { // -&gt; vertex, face, normal 리스트의 동적 할당 해제 void readAse(const char* fileName) { // ASE file load void draw(float scale = 1.0f, bool bCoord = false) { // 객체를 그리기 void setColor(float r, float g, float b, float a) { // 객체에 색 지정하기 void draw(float r, float g, float b, float scale = 1.0f, bool bCoord = false) { // 색을 지정해서 객체를 그리기 void clearAse()// -&gt; vertex, face, normal 리스트의 동적 할당 해제 if (nVtx != 0) // Vertex가 남아있을 때     delete[] vertex; if (nFace != 0) { // Face가 남아있을 때     delete[] face;     delete[] normal; } nVtx = nFace = 0; // 초기화 void readAse(const char* fileName) // ASE file load </pre>	

변수

```
FILE* fp; // 객체 선택 파일 포인터
char line[256], str[40]; // 한 라인과 문자열 버퍼
float _x, _y, _z; // 객체 정점 정보
int num = 0; // 읽어올 정점 개수

if ((fp = fopen(fileName, "r")) == NULL) { // 파일 열기 오류 탐지
    cout << "File is Not Found" << endl;
    return
}

.....
ASE 파일을 통해
*MESH, *TIMEVALUE, *MESH_NUMVERTEX, * MESH_NUMFACES, *MESH_VERTEX_LIST, { }, *MESH_NORMALS
등의 개수를 셸
....
```

**void draw(float scale = 1.0f, bool bCoord = false) // 객체를 그리기**

```
glBegin(GL_TRIANGLES); // 객체 그리기 시작
for (int i = 0; i < nFace; i++) {
    // 객체 생성 : face
    Vertex* v1 = &vertex[face[i].vi[0]];
    Vertex* v2 = &vertex[face[i].vi[1]];
    Vertex* v3 = &vertex[face[i].vi[2]];
    // 법선 벡터 정보 얻어와서 그리기 : normal
    glNormal3fv(normal[i].norV1);
    glVertex3f(v1->x / scale, v1->y / scale, v1->z / scale);
    glNormal3fv(normal[i].norV2);
    glVertex3f(v2->x / scale, v2->y / scale, v2->z / scale);
    glNormal3fv(normal[i].norV3);
    glVertex3f(v3->x / scale, v3->y / scale, v3->z / scale);
}
glEnd(); // 객체 그리기 끝
if (bCoord) // bCoord가 활성화되어있을 때
    glDrawCoord(1.0); // 객체를 그림
```

**// 객체에 색 지정하기**

**void setColor(float r, float g, float b, float a)**

```
float color[4] = { r, g, b, a };
glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, color);
glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, color);
```

**// 색을 지정해서 객체를 그리기**

**void draw(float r, float g, float b, float scale = 1.0f, bool bCoord = false)**

```
setColor(r, g, b, 1.0);
draw(scale, bCoord);
```

## 5. Robot.h

로봇 클래스	
<div> <div>필드</div> <div> <div>Body</div> <div>Head</div> <div>LeftArm</div> <div>LeftFoot</div> <div>LeftHand</div> <div>LeftLeg</div> <div>RightArm</div> <div>RightFoot</div> <div>RightHand</div> <div>RightLeg</div> <div>scale</div> </div> </div> <div> <div>메서드</div> <div> <div>draw</div> <div>resize</div> <div>Robot</div> </div> </div>	
<b>필드</b>	
// 로봇의 부품 Mesh Head, Body, RightArm, RightHand, LeftArm, LeftHand; Mesh RightLeg, RightFoot, LeftLeg, LeftFoot;	
float scale; // 모델의 스케일	
<b>메서드</b>	
Robot() // 로봇 readAse 로딩 구현 void resize(bool flag = true) // z 구현 : 크기 다시 설정하기 void draw() // 로봇 조립해서 그리기	
Robot() // 로봇 readAse 로딩 구현 scale = 100; // 좌표값이 -100 ~ 100 범위	
printf("Loading Robot Models...\n"); Head.readAse("s_Head.ASE"); printf("loading Head.....OK.\n"); Body.readAse("S_Body.ASE"); printf("loading Body.....OK.\n"); RightArm.readAse("S_RightArm.ASE"); printf("loading RightArm.....OK.\n"); RightHand.readAse("S_RightHand.ASE"); printf("loading RightHand.....OK.\n"); LeftArm.readAse("S_LeftArm.ASE"); printf("loading LeftArm.....OK.\n"); LeftHand.readAse("S_LeftHand.ASE"); printf("loading LeftHand.....OK.\n"); RightLeg.readAse("S_RightLeg.ASE"); printf("loading RightLeg.....OK.\n"); RightFoot.readAse("S_RightFoot.ASE"); printf("loading RightFoot.....OK.\n"); LeftLeg.readAse("S_LeftLeg.ASE"); printf("loading LeftLeg.....OK.\n"); LeftFoot.readAse("S_LeftFoot.ASE"); printf("loading LeftFoot.....OK.\n");	
void resize(bool flag = true) // z 구현 : 크기 다시 설정하기	
if (flag) scale *= 1.05f;	
else scale *= 0.95;	



```

void draw() // 로봇 조립해서 그리기
Body.draw(0.5, 0.8, 0.8, scale, true);

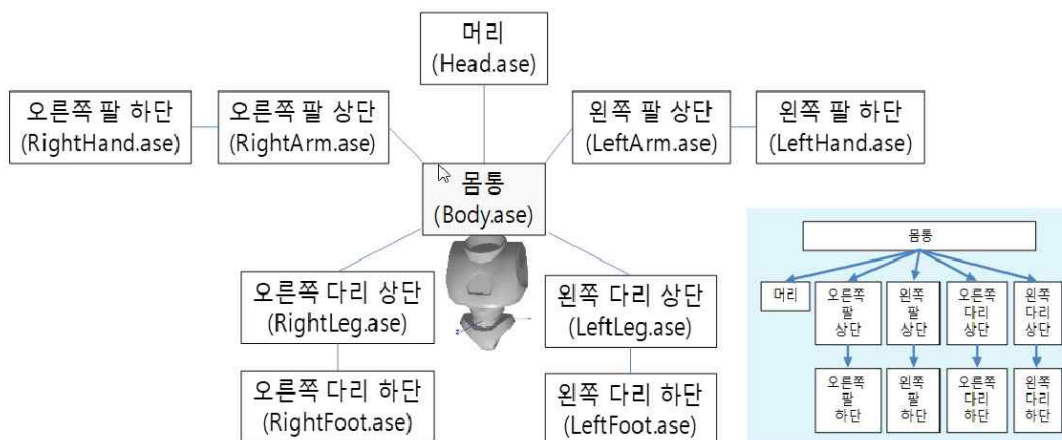
glPushMatrix(); // 몸통 좌표계 저장
glTranslated(0.0, 0.45, -0.07);
glScalef(1.1f, 1.1f, 1.1f);
Head.draw(0.8, 0.7, 0.7, scale);
glPopMatrix(); // 몸통 좌표계로 돌아가기

glPushMatrix(); // 몸통 좌표계 저장
glTranslated(-0.25, 0.32, 0.0);
glScalef(1.0f, 1.0f, 1.0f);
RightArm.draw(0.4, 0.4, 0.8, scale);

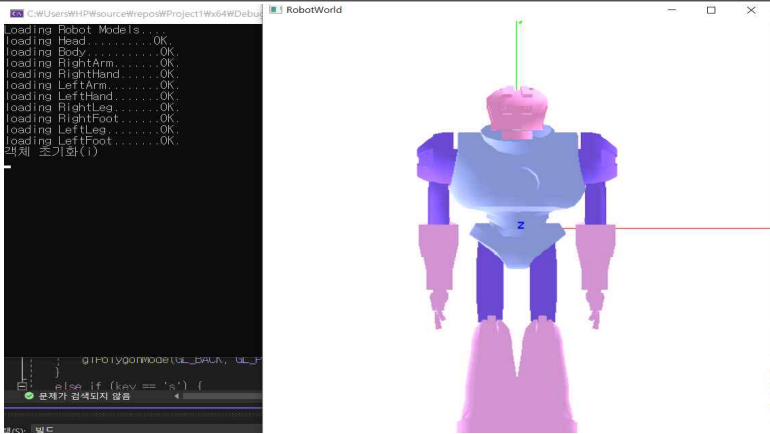
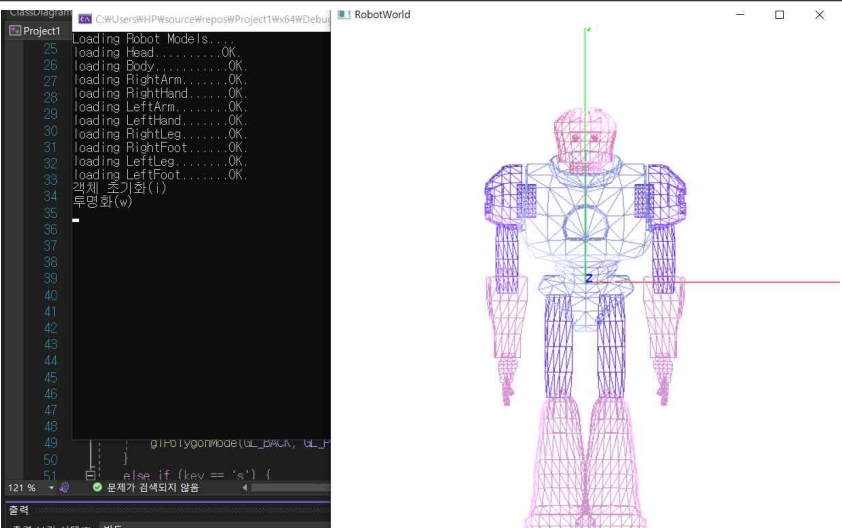
glPushMatrix(); // 오른어깨 좌표계 저장
glTranslated(-0.05, -0.3, 0.0);
glScalef(1.0f, 1.0f, 1.0f);
RightHand.draw(0.8, 0.8, 0.8, scale);
glPopMatrix(); // 오른어깨 좌표계로 돌아가기
glPopMatrix(); // 몸통 좌표계로 돌아가기

.....
이후 아래와 같은 형식으로 진행된다.

```

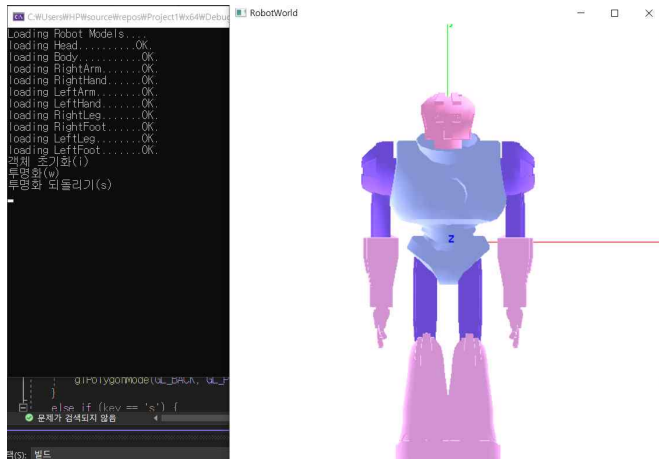


## - 메인(MakeRobot)

<b>헤더</b>	<pre>#include "Robot.h" #include &lt;gl/glut.h&gt;</pre>
<b>변수 목록</b>	<pre>static Mesh obj3D; // Mesh 객체 static Robot robot; // Robot 객체 // static bool bRobotRun = false; // 로봇 실행 여부 static int PrevX, PrevY; // 이전 X, Y 좌표</pre>
<b>기능</b>	<p>- 객체 초기화(i)</p>
<pre>if (key == 'i') {     printf("객체 초기화(i)\n");     glMatrixMode(GL_MODELVIEW);     glLoadIdentity(); }</pre>	<p>// 초기화</p>
 <p>The screenshot shows a C++ IDE with a file explorer on the left displaying a project structure. The main editor shows a list of loaded models: Robot, Models, Head, Body, RightArm, RightHand, LeftArm, LeftHand, RightLeg, RightFoot, LeftLeg, and LeftFoot. The console output shows '객체 초기화(i)' (Object Initialization). The 3D window 'RobotWorld' displays a robot model with a pink head, blue body, and purple limbs. A green Z-axis line is visible in the background.</p>	
<p>- 투명화(w)</p>	<p>// 투명화</p>
<pre>else if (key == 'w') {     printf("투명화(w)\n");     glPolygonMode(GL_FRONT, GL_LINE);     glPolygonMode(GL_BACK, GL_POINT); }</pre>	
 <p>The screenshot shows the same C++ IDE as before, but the console output now includes '투명화(w)' (Transparency). The 3D window 'RobotWorld' displays the robot model with a pink head, blue body, and purple limbs. The robot's body and limbs are rendered in wireframe mode (GL_LINE), and the head is rendered in point mode (GL_POINT). A green Z-axis line is visible in the background.</p>	

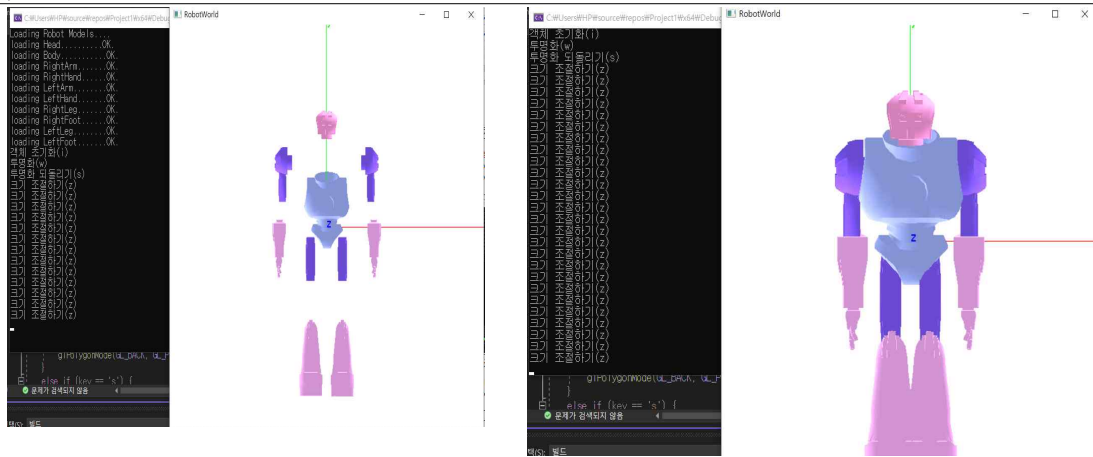
### - 투명화 되돌리기(s)

```
else if (key == 's') { // 투명화 되돌리기
    printf("투명화 되돌리기(s)\n");
    glPolygonMode(GL_FRONT, GL_FILL);
    glPolygonMode(GL_BACK, GL_LINE);
}
```



### - 크기 조절하기(z)

```
else if (key == 'z' || key == 'Z') { // z : 작아지기, Z : 커지기
    printf("크기 조절하기(z)\n");
    robot.resize(key == 'z');
}
```



### - 나가기(q)

```
else if (key == 'q') exit(0); // 나가기
```

### 마우스 이벤트

```
void mouseClicked(int button, int state, int x, int y)
.... 코드 일부 생략
if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN) {
    PrevX = x;
    PrevY = y;
}
void mouseMotion(GLint x, GLint y) ....
glRotated(x - PrevX, 0, 1, 0);
glRotated(y - PrevY, 1, 0, 0); .... 코드 일부 생략
```

PrevX, PrevY를 이용해 로봇의 움직임을 나타낼 때 사용

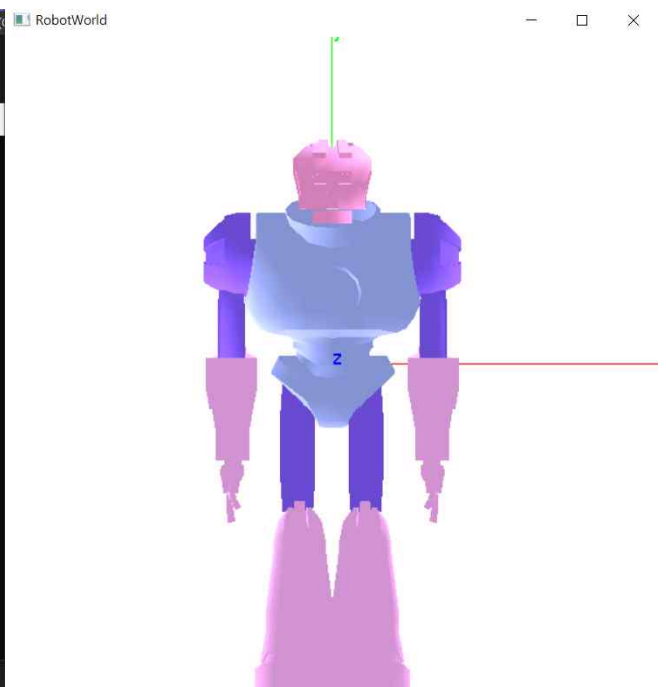
mouseClick

```
(전역 범위) mouseMotion()
<gl/glut.h>

mesh obj3D; // Mesh 객체
Robot robot; // Robot 객체

C:\Users\HP\source\repos\Project1\wx64\Debug\Project1.exe

Loading Robot Models...
loading Head.....OK.
loading Body.....OK.
loading RightArm.....OK.
loading RightHand.....OK.
loading LeftArm.....OK.
loading LeftHand.....OK.
loading RightLeg.....OK.
loading RightFoot.....OK.
loading LeftLeg.....OK.
loading LeftFoot.....OK.
mouseClick : (296, 106)
mouseClick : (296, 106)
mouseClick : (159, 225)
mouseClick : (159, 225)
mouseClick : (302, 316)
mouseClick : (302, 316)
```



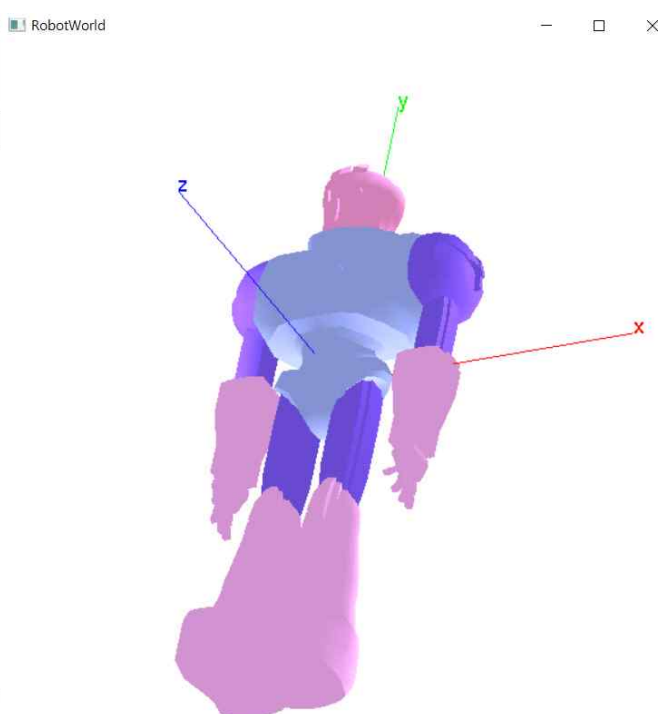
mouseMotion

```
(전역 범위) mouseMotion()
<gl/glut.h>

obj3D; // Mesh 객체
Robot robot; // Robot 객체

C:\Users\HP\source\repos\Project1\wx64\Debug\Project1.exe

mouseMotion : (96, 231)
mouseMotion : (98, 231)
mouseMotion : (99, 232)
mouseMotion : (101, 234)
mouseMotion : (102, 235)
mouseMotion : (103, 236)
mouseMotion : (105, 238)
mouseMotion : (105, 238)
mouseMotion : (106, 239)
mouseMotion : (107, 241)
mouseMotion : (107, 241)
mouseMotion : (108, 243)
mouseMotion : (110, 246)
mouseMotion : (113, 250)
mouseMotion : (114, 253)
mouseMotion : (118, 256)
mouseMotion : (118, 258)
mouseMotion : (120, 260)
mouseMotion : (121, 261)
mouseMotion : (121, 262)
mouseMotion : (122, 263)
mouseMotion : (122, 263)
mouseMotion : (123, 264)
mouseMotion : (124, 265)
mouseMotion : (125, 266)
mouseMotion : (125, 266)
mouseMotion : (126, 267)
mouseMotion : (127, 267)
mouseClick : (127, 267)
```



## 고찰 및 느낀점

이번 과제에서는 로봇을 조립하는 실습을 해보았다. 과제를 진행하며 glWin에서 파일 인코딩 형식을 지정하고 파일 선택자를 만드는 방식을 하는 방법을 배워서 신기했지만, 계속 파일을 여는 데는 실패했기에 그 이유를 찾지 못하고 로봇만 조립하게 되어 한 가지 아쉬움이 있었다. 과제를 이해하는데 시간이 좀 걸려서 새로운 로봇을 조립하기에 시간이 부족해 기존 모델로 실습을 하였지만, 다음 과제에서는 시간을 조금 더 할애해 기존 모델을 수정을 해보고 싶은 생각이 들었다. 로봇이 실제로 출력되어 신기했고 정말 재미있는 실습이었다.