コンピュータグラフィックス修了作品

説明書・ソースプログラム

|  |  |
| --- | --- |
| 学籍番号 | ---------- |
| 氏名 | ---------- |
| 作品名 | 間合イ |
| 入選時の氏名の公表 | 可 |
| 入選時の作品の配布 | 可 |

作品の説明(制作の意図・動機・反省など

普段よく格闘ゲームをプレイするので、そこから着想を得て、格闘ゲームを作ることに決めた。格闘ゲームでは、自分のキャラと相手のキャラとの間合いが重要である。今回は、その間合いに着目してゲームを作ろうと思った。

一般的な格闘ゲームでは、主に体力性が用いられており、相手の体力を先に削り切った方が勝利となる。しかし、体力が尽きない限り、何度相手の攻撃を受けてもよい、というのでは、今回のテーマである間合いが軽視されかねない、と判断し、自分も相手も、一度でも攻撃を当てられれば勝ち、とした。

余裕があれば、背景や凝ったキャラクターデザインなど、見た目を凝ってみようと考えていたが、時間の都合上、キャラクターのデザインは必要最低限を満たすのみとした。

ここからは「間合イ」についての説明を行う。「間合イ」では、自分、相手、共に一撃必殺である。そのため、一撃を当てるための「間合い管理」と「反射神経」が必要である。このゲームには、3種類の攻撃と、「弾き」という要素が存在する。

基本となる「その場攻撃」は、文字通りその場で攻撃ができるため、間合いを保ったまま攻撃ができる。「前進攻撃」では、前進して攻撃できるため、間合いの少し外から攻撃を仕掛けられる。しかし、攻撃判定が発生する前に一歩前進するので、無防備な体を一瞬晒すことになる。「溜め攻撃」は、攻撃判定が発生するのが遅い代わりに、判定が強く、相手の攻撃とかち合った場合には相手を吹き飛ばし、相手が「弾き」の構えをしていた場合にはそのまま断ち切ることができる。「弾き」は、その名の通り、相手の攻撃を弾くことが出来る。発生もはやいため、反射神経が良ければ相手の攻撃をみてからでも弾くことができる。弾かれた相手は大きく後退し、後ろ側の壁が近くなる。後ろに壁を背負うと、引いて間合いを取ることが出来なくなるため、不利である。

使用方法(遊び方)と実行画面の例

操作手順

[難易度設定 (開始から1.5秒以内) ]

0キー・・・難易度　0　に決定　（難易度0では相手は動かない）

1キー・・・難易度　１　に決定

2キー・・・難易度　2　に決定

３キー・・・難易度　3　に決定

[戦闘開始 (開始から1.5秒以降) ]

ｓキー・・・入力取り消し

aキー・・・一歩後退

ｄキー・・・一歩前進

ｗキー・・・「弾き」入力　　　「弾き」発生１フレーム　全体２フレーム（1フレーム0.4秒）

ｊキー・・・「その場攻撃」入力　　　「その場攻撃」発生２フレーム　全体3フレーム

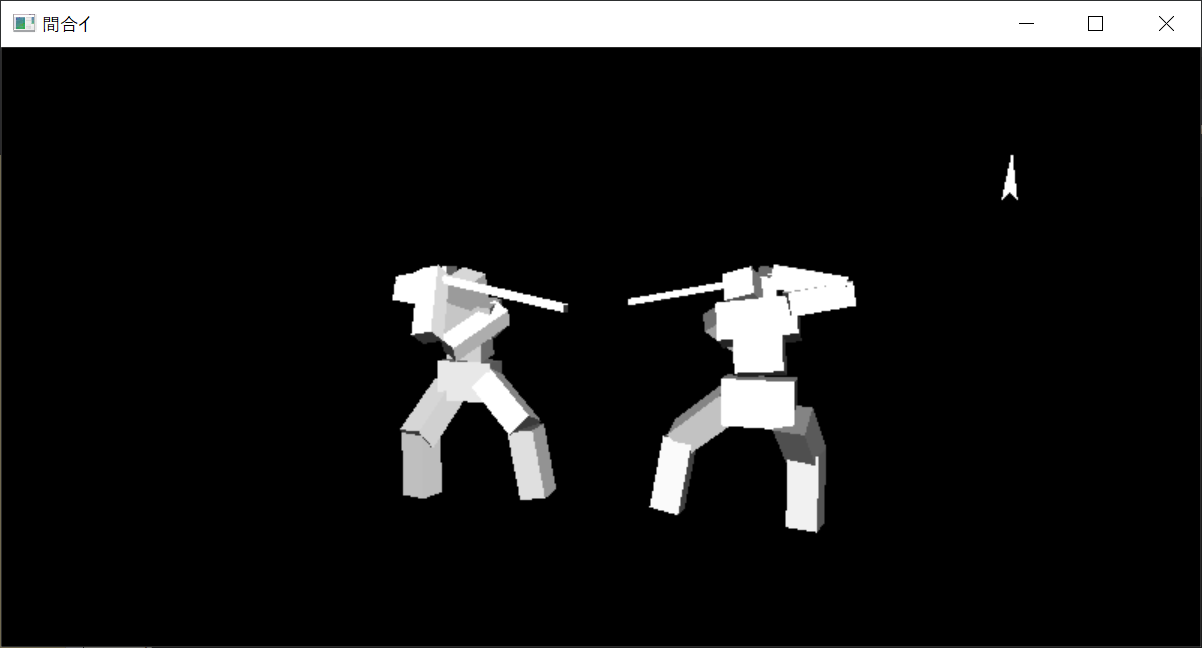
ｋキー・・・「前進攻撃」入力　　　「前進攻撃」発生２フレーム　全体４フレーム

ｌキー・・・「溜め攻撃」入力　　　「溜め攻撃」発生３フレーム　全体４フレーム

spaceキー・・・プログラムの終了

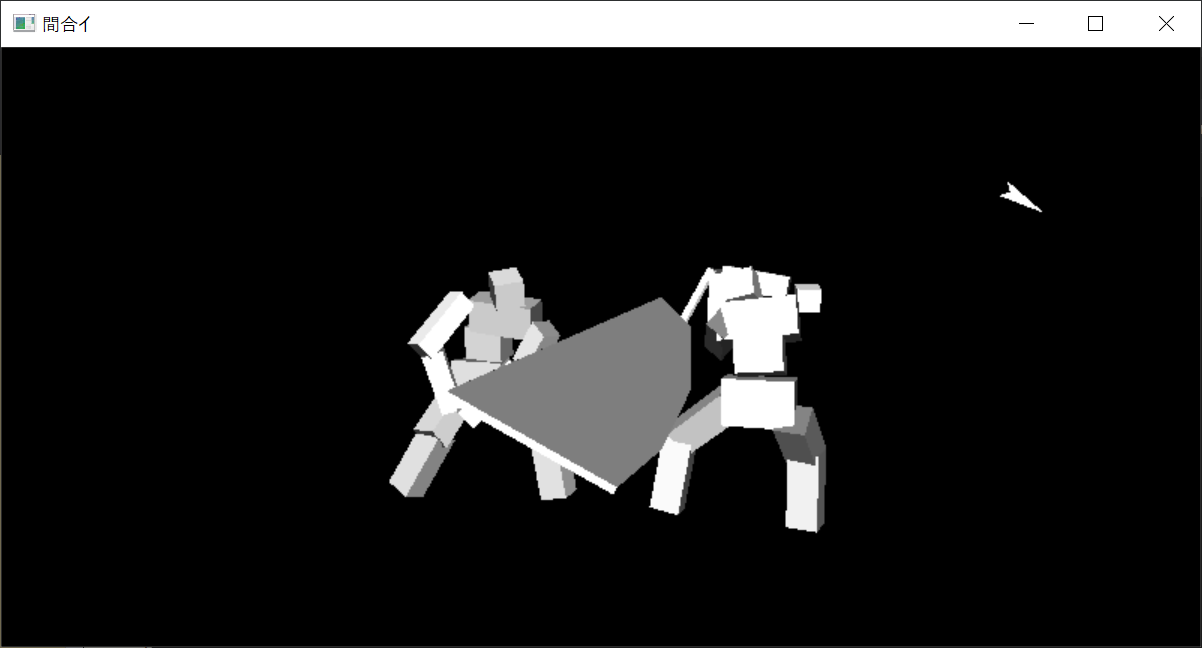
実行例

・難易度設定



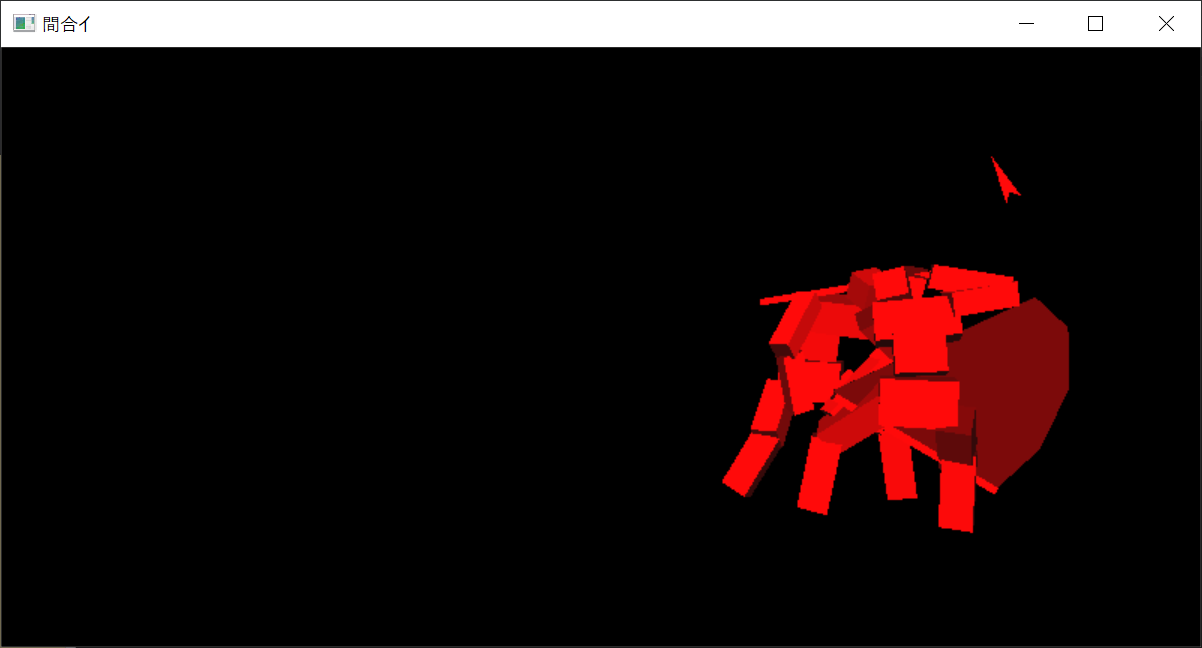
プログラムを実行するとまず1.5秒以内に難易度を決定する。無入力の場合、難易度は１に設定される。

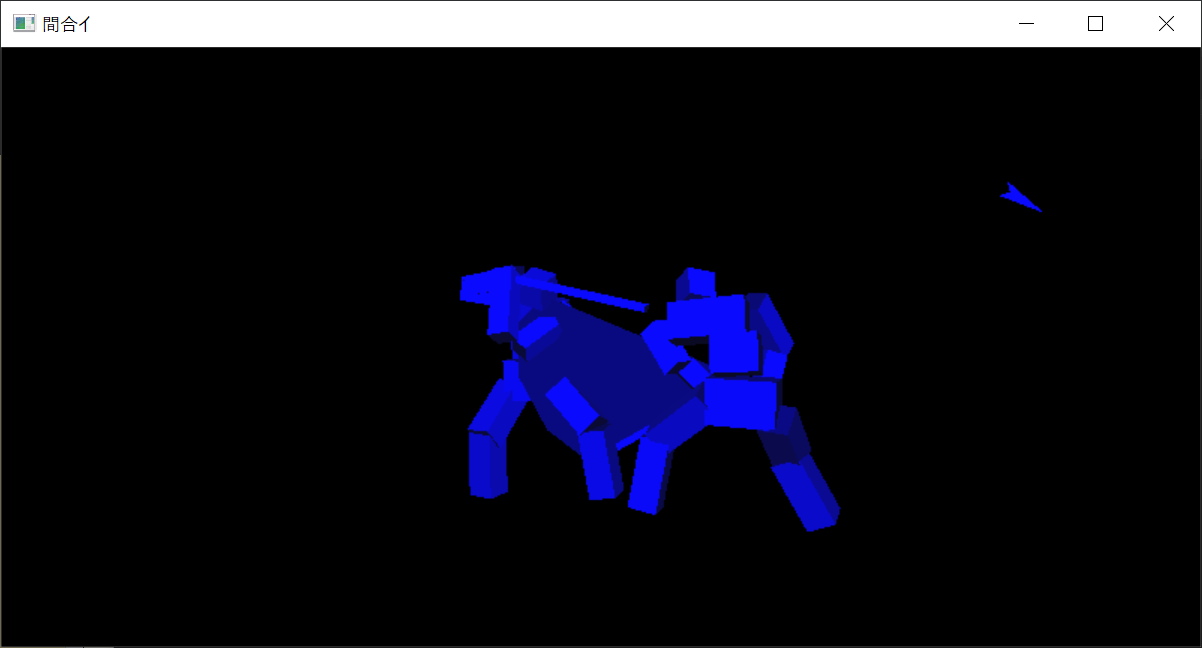
・戦闘開始



右の時計針でフレームのタイミングが把握できる。

・勝敗決定





勝つと赤色に、負けると青色に光が変化する。

spaceキーを押すといつでもゲームを終了できる。

参考書・参考サイト

敵キャラクターの動きをランダムにするため、下記URLのサイトを参考にし、

rand関数とsrand関数とtime関数を使った

https://monozukuri-c.com/langc-funclist-rand/

ソースプログラム

#include "glut.h"

#include <GL/gl.h>

#include <math.h>

#include <stdio.h>

#include <time.h>

#define pi 3.141592

#define T 300 //タイマーの間隔[ミリ秒]

#define set 1500 //最初の難易度設定時のタイマー[ミリ秒]

int judge = 0;//勝敗判定　１で勝ち　２で負け

char input = '0';

bool setting = true;

int level = 1;

int my\_state = 0;

int my\_state\_next = 0;

int enemy\_state = 0;

int enemy\_state\_next = 0;

int my\_location = -2;

int enemy\_location = 2;

int my\_attack = 0;

int enemy\_attack = 0;

double RotAngle = 0.0;//タイマーに使う

float amber[] = { 0.5,0.5,0.5,1.0 };//光の色

void KeyboardHandler(unsigned char key, int x, int y) {//キーボードの入力管理

// 押されたキーがスペース(空白)だったらプログラムを終了する

if (key == ' ')

exit(0);

if (setting) {

if (key == '1')

level = 1;

else if (key == '2')

level = 2;

else if (key == '3')

level = 3;

else if (key == '0')

level = 0;

}

else if (key == 'd') {//キーボードでコマンドをインプットする。

input = 'd';

}

else if (key == 'a') {

input = 'a';

}

else if (key == 'w') {

input = 'w';

}

else if (key == 's') {

input = 's';

}

else if (key == 'j') {

input = 'j';

}

else if (key == 'k') {

input = 'k';

}

else if (key == 'l') {

input = 'l';

}

}

void attackbox(int howlong) {//間合いを入力する

glBegin(GL\_POLYGON);

glNormal3f(0, 0, -1.0);

glVertex3f(0.5, 2, 0.6);

glVertex3f(0.5, 1, 0.6);

glVertex3f(0.5 + howlong, 1, 0.6);

glVertex3f(0.5 + howlong, 2, 0.6);

glEnd();

}

void box() {

glBegin(GL\_POLYGON);//上面

glNormal3f(0, 1, 0);

glVertex3f(-0.5, 0.5, -0.5);//1

glVertex3f(0.5, 0.5, -0.5);//2

glVertex3f(0.5, 0.5, 0.5);//3

glVertex3f(-0.5, 0.5, 0.5);//4

glEnd();

glBegin(GL\_POLYGON);//下面

glNormal3f(0, -1, 0);

glVertex3f(-0.5, -0.5, -0.5);//5

glVertex3f(0.5, -0.5, -0.5);//6

glVertex3f(0.5, -0.5, 0.5);//7

glVertex3f(-0.5, -0.5, 0.5);//8

glEnd();

glBegin(GL\_QUAD\_STRIP);//側面

glNormal3f(0, 0, -1);

glVertex3f(-0.5, 0.5, -0.5);//1

glVertex3f(-0.5, -0.5, -0.5);//5

glVertex3f(0.5, 0.5, -0.5);//2

glVertex3f(0.5, -0.5, -0.5);//6

glNormal3f(1, 0, 0);

glVertex3f(0.5, 0.5, 0.5);//3

glVertex3f(0.5, -0.5, 0.5);//7

glNormal3f(0, 0, 1);

glVertex3f(-0.5, 0.5, 0.5);//4

glVertex3f(-0.5, -0.5, 0.5);//8

glNormal3f(-1, 0, 0);

glVertex3f(-0.5, 0.5, -0.5);//1

glVertex3f(-0.5, -0.5, -0.5);//5

glEnd();

}

void human(int state) {//キャラのモデルをstateで管理

if (state == 0 || state == 1 || state == 3) {//ニュートラル状態

//あたま

glPushMatrix();

glTranslatef(0.1, 2.8, -0.3);

glRotatef(30, 1, 0, 0);

glRotatef(-30, 0, 1, 0);

glScalef(0.4, 0.4, 0.4);

box();

glPopMatrix();

//むね

glPushMatrix();

glTranslatef(0.0, 2.4, -0.8);

glRotatef(20, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(1.0, 0.5, 0.4);

box();

glPopMatrix();

//ひだりうで

glPushMatrix();

glTranslatef(0.2, 2.2, -0.4);

glRotatef(45.0, 1, 0, 0);

glRotatef(45.0, 0, 1, 0);

glRotatef(-90.0, 0, 0, 1);

glScalef(0.3, 1.0, 0.3);

box();

glPopMatrix();

//ひだりて

glPushMatrix();

glTranslatef(-0.3, 2.6, 0.2);

glRotatef(-30.0, 1, 0, 0);

glRotatef(70, 0, 1, 0);

glRotatef(0.0, 0, 0, 1);

glScalef(0.3, 1.0, 0.3);

box();

glPopMatrix();

//みぎうで

glPushMatrix();

glTranslatef(-0.7, 2.7, -1.0);

glRotatef(0.0, 1, 0, 0);

glRotatef(0.0, 0, 1, 0);

glRotatef(80.0, 0, 0, 1);

glScalef(0.3, 0.8, 0.3);

box();

glPopMatrix();

//みぎて

glPushMatrix();

glTranslatef(-0.6, 2.9, -0.9);

glRotatef(0.0, 1, 0, 0);

glRotatef(-20.0, 0, 1, 0);

glRotatef(-80.0, 0, 0, 1);

glScalef(0.3, 0.9, 0.3);

box();

glPopMatrix();

//はら

glPushMatrix();

glTranslatef(0.0, 2.0, -0.95);

glRotatef(15, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.6, 0.5, 0.4);

box();

glPopMatrix();

//こし

glPushMatrix();

glTranslatef(0.0, 1.4, -0.95);

glRotatef(360, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.9, 0.6, 0.4);

box();

glPopMatrix();

//みぎもも

glPushMatrix();

glTranslatef(-0.55, 0.9, -0.9);

glRotatef(-30, 0, 0, 1);

glRotatef(-40, 1, 0, 0);

glRotatef(45, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//みぎあし

glPushMatrix();

glTranslatef(-0.7, 0.2, -0.7);

glRotatef(360, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-30, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりもも

glPushMatrix();

glTranslatef(0.6, 1.1, -0.55);

glRotatef(45, 0, 0, 1);

glRotatef(-30, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりあし

glPushMatrix();

glTranslatef(1.0, 0.3, -0.3);

glRotatef(10, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//剣

glPushMatrix();

glTranslatef(0.5, 2.8, 0.4);

glRotatef(0.0, 1, 0, 0);

glRotatef(-30.0, 0, 1, 0);

glRotatef(80.0, 0, 0, 1);

glScalef(0.1, 2.5, 0.1);

box();

glPopMatrix();

}else if(state == 20 ||state == 30 || state == 41){//振り下ろす直前

//あたま

glPushMatrix();

glTranslatef(0.1, 2.8, -0.3);

glRotatef(30, 1, 0, 0);

glRotatef(-30, 0, 1, 0);

glScalef(0.4, 0.4, 0.4);

box();

glPopMatrix();

//むね

glPushMatrix();

glTranslatef(0.0, 2.4, -0.8);

glRotatef(20, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(1.0, 0.5, 0.4);

box();

glPopMatrix();

//ひだりうで

glPushMatrix();

glTranslatef(0.7, 2.7, 0.0);

glRotatef(0, 1, 0, 0);

glRotatef(-45, 0, 1, 0);

glRotatef(-60.0, 0, 0, 1);

glScalef(0.3, 0.7, 0.3);

box();

glPopMatrix();

//ひだりて

glPushMatrix();

glTranslatef(0.5, 3.2, 0.2);

glRotatef(0, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glRotatef(45, 0, 0, 1);

glScalef(0.3, 1.0, 0.3);

box();

glPopMatrix();

//みぎうで

glPushMatrix();

glTranslatef(-0.7, 2.7, -0.3);

glRotatef(0, 1, 0, 0);

glRotatef(45, 0, 1, 0);

glRotatef(60, 0, 0, 1);

glScalef(0.3, 0.5, 0.3);

box();

glPopMatrix();

//みぎて

glPushMatrix();

glTranslatef(-0.45, 3.05, -0.2);

glRotatef(10, 1, 0, 0);

glRotatef(10, 0, 1, 0);

glRotatef(-45, 0, 0, 1);

glScalef(0.3, 0.9, 0.3);

box();

glPopMatrix();

//はら

glPushMatrix();

glTranslatef(0.0, 2.0, -0.95);

glRotatef(15, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.6, 0.5, 0.4);

box();

glPopMatrix();

//こし

glPushMatrix();

glTranslatef(0.0, 1.4, -0.95);

glRotatef(360, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.9, 0.6, 0.4);

box();

glPopMatrix();

//みぎもも

glPushMatrix();

glTranslatef(-0.55, 0.9, -0.9);

glRotatef(-30, 0, 0, 1);

glRotatef(-40, 1, 0, 0);

glRotatef(45, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//みぎあし

glPushMatrix();

glTranslatef(-0.7, 0.2, -0.7);

glRotatef(360, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-30, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりもも

glPushMatrix();

glTranslatef(0.6, 1.1, -0.55);

glRotatef(45, 0, 0, 1);

glRotatef(-30, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりあし

glPushMatrix();

glTranslatef(1.0, 0.3, -0.3);

glRotatef(10, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//剣

glPushMatrix();

glTranslatef(-0.8, 3.4, 0.4);

glRotatef(0.0, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glRotatef(-80.0, 0, 0, 1);

glScalef(0.1, 2.5, 0.1);

box();

glPopMatrix();

}else if(state == 21 || state == 31 || state == 42){//振り下ろす

//あたま

glPushMatrix();

glTranslatef(0.7, 2.8, 0.0);

glRotatef(30, 1, 0, 0);

glRotatef(-70, 0, 1, 0);

glScalef(0.4, 0.4, 0.4);

box();

glPopMatrix();

//むね

glPushMatrix();

glTranslatef(0.6, 2.4, -0.6);

glRotatef(20, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(1.0, 0.5, 0.4);

box();

glPopMatrix();

//ひだりうで

glPushMatrix();

glTranslatef(1.1, 2.0, 0.0);

glRotatef(0, 1, 0, 0);

glRotatef(-45, 0, 1, 0);

glRotatef(-40.0, 0, 0, 1);

glScalef(0.3, 0.7, 0.3);

box();

glPopMatrix();

//ひだりて

glPushMatrix();

glTranslatef(0.5, 1.4, 0.2);

glRotatef(0, 1, 0, 0);

glRotatef(10, 0, 1, 0);

glRotatef(-45, 0, 0, 1);

glScalef(0.3, 1.0, 0.3);

box();

glPopMatrix();

//みぎうで

glPushMatrix();

glTranslatef(-0.25, 2.3, -0.2);

glRotatef(0, 1, 0, 0);

glRotatef(45, 0, 1, 0);

glRotatef(-45, 0, 0, 1);

glScalef(0.3, 1.0, 0.3);

box();

glPopMatrix();

//みぎて

glPushMatrix();

glTranslatef(-0.25, 1.5, -0.2);

glRotatef(10, 1, 0, 0);

glRotatef(10, 0, 1, 0);

glRotatef(15, 0, 0, 1);

glScalef(0.3, 0.9, 0.3);

box();

glPopMatrix();

//はら

glPushMatrix();

glTranslatef(0.3, 2.0, -0.95);

glRotatef(15, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.6, 0.5, 0.4);

box();

glPopMatrix();

//こし

glPushMatrix();

glTranslatef(0.2, 1.4, -0.95);

glRotatef(360, 1, 0, 0);

glRotatef(-25, 0, 1, 0);

glScalef(0.9, 0.6, 0.4);

box();

glPopMatrix();

//みぎもも

glPushMatrix();

glTranslatef(-0.35, 0.9, -0.9);

glRotatef(-30, 0, 0, 1);

glRotatef(-40, 1, 0, 0);

glRotatef(45, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//みぎあし

glPushMatrix();

glTranslatef(-0.7, 0.2, -0.7);

glRotatef(-30, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-30, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりもも

glPushMatrix();

glTranslatef(0.9, 1.1, -0.55);

glRotatef(45, 0, 0, 1);

glRotatef(-30, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりあし

glPushMatrix();

glTranslatef(1.3, 0.3, -0.3);

glRotatef(10, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//剣

glPushMatrix();

glTranslatef(1.1, 0.8, 0.7);

glRotatef(0.0, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glRotatef(60.0, 0, 0, 1);

glScalef(0.1, 2.5, 0.1);

box();

glPopMatrix();

//残像

//表

glPushMatrix();

glBegin(GL\_POLYGON);

glNormal3f(0, 0, 1.0);

glVertex3f(0, 1.5, 0.71);

glVertex3f(2.2, 0.2, 0.71);

glVertex3f(2.8, 0.7, 0.61);

glVertex3f(3.2, 1.5, 0.51);

glVertex3f(3.2, 2.3, 0.41);

glVertex3f(2.8, 2.7, 0.31);

glVertex3f(2.2, 1.5, 0.21);

glEnd();

glPopMatrix();

//裏

glPushMatrix();

glBegin(GL\_POLYGON);

glNormal3f(0, 0, -1.0);

glVertex3f(0, 1.5, 0.7);

glVertex3f(2.2, 0.2, 0.7);

glVertex3f(2.8, 0.7, 0.6);

glVertex3f(3.2, 1.5, 0.5);

glVertex3f(3.2, 2.3, 0.4);

glVertex3f(2.8, 2.7, 0.3);

glVertex3f(2.2, 1.5, 0.2);

glEnd();

glPopMatrix();

}else if(state == 22 || state ==32 || state == 33 || state == 43){//後隙

//あたま

glPushMatrix();

glTranslatef(0.7, 2.8, 0.0);

glRotatef(30, 1, 0, 0);

glRotatef(-70, 0, 1, 0);

glScalef(0.4, 0.4, 0.4);

box();

glPopMatrix();

//むね

glPushMatrix();

glTranslatef(0.6, 2.4, -0.6);

glRotatef(20, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(1.0, 0.5, 0.4);

box();

glPopMatrix();

//ひだりうで

glPushMatrix();

glTranslatef(1.1, 2.0, 0.0);

glRotatef(0, 1, 0, 0);

glRotatef(-45, 0, 1, 0);

glRotatef(-40.0, 0, 0, 1);

glScalef(0.3, 0.7, 0.3);

box();

glPopMatrix();

//ひだりて

glPushMatrix();

glTranslatef(0.5, 1.4, 0.2);

glRotatef(0, 1, 0, 0);

glRotatef(10, 0, 1, 0);

glRotatef(-45, 0, 0, 1);

glScalef(0.3, 1.0, 0.3);

box();

glPopMatrix();

//みぎうで

glPushMatrix();

glTranslatef(-0.25, 2.3, -0.2);

glRotatef(0, 1, 0, 0);

glRotatef(45, 0, 1, 0);

glRotatef(-45, 0, 0, 1);

glScalef(0.3, 1.0, 0.3);

box();

glPopMatrix();

//みぎて

glPushMatrix();

glTranslatef(-0.25, 1.5, -0.2);

glRotatef(10, 1, 0, 0);

glRotatef(10, 0, 1, 0);

glRotatef(15, 0, 0, 1);

glScalef(0.3, 0.9, 0.3);

box();

glPopMatrix();

//はら

glPushMatrix();

glTranslatef(0.3, 2.0, -0.95);

glRotatef(15, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.6, 0.5, 0.4);

box();

glPopMatrix();

//こし

glPushMatrix();

glTranslatef(0.2, 1.4, -0.95);

glRotatef(360, 1, 0, 0);

glRotatef(-25, 0, 1, 0);

glScalef(0.9, 0.6, 0.4);

box();

glPopMatrix();

//みぎもも

glPushMatrix();

glTranslatef(-0.35, 0.9, -0.9);

glRotatef(-30, 0, 0, 1);

glRotatef(-40, 1, 0, 0);

glRotatef(45, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//みぎあし

glPushMatrix();

glTranslatef(-0.7, 0.2, -0.7);

glRotatef(-30, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-30, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりもも

glPushMatrix();

glTranslatef(0.9, 1.1, -0.55);

glRotatef(45, 0, 0, 1);

glRotatef(-30, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりあし

glPushMatrix();

glTranslatef(1.3, 0.3, -0.3);

glRotatef(10, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//剣

glPushMatrix();

glTranslatef(1.1, 0.8, 0.7);

glRotatef(0.0, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glRotatef(60.0, 0, 0, 1);

glScalef(0.1, 2.5, 0.1);

box();

glPopMatrix();

}

else if(state == 40 || state == 60 || state == 61 || state == 62){//溜めモーション　と　弾かれモーション

//あたま

glPushMatrix();

glTranslatef(-0.2, 2.8, -0.3);

glRotatef(30, 1, 0, 0);

glRotatef(-30, 0, 1, 0);

glRotatef(30, 0, 0, 1);

glScalef(0.4, 0.4, 0.4);

box();

glPopMatrix();

//むね

glPushMatrix();

glTranslatef(-0.2, 2.4, -0.8);

glRotatef(20, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(1.0, 0.5, 0.4);

box();

glPopMatrix();

//ひだりうで

glPushMatrix();

glTranslatef(0.5, 2.7, 0.0);

glRotatef(0, 1, 0, 0);

glRotatef(-45, 0, 1, 0);

glRotatef(-60.0, 0, 0, 1);

glScalef(0.3, 0.7, 0.3);

box();

glPopMatrix();

//ひだりて

glPushMatrix();

glTranslatef(0.3, 3.2, 0.2);

glRotatef(0, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glRotatef(45, 0, 0, 1);

glScalef(0.3, 1.0, 0.3);

box();

glPopMatrix();

//みぎうで

glPushMatrix();

glTranslatef(-0.9, 2.7, -0.3);

glRotatef(0, 1, 0, 0);

glRotatef(45, 0, 1, 0);

glRotatef(60, 0, 0, 1);

glScalef(0.3, 0.5, 0.3);

box();

glPopMatrix();

//みぎて

glPushMatrix();

glTranslatef(-0.65, 3.05, -0.2);

glRotatef(10, 1, 0, 0);

glRotatef(10, 0, 1, 0);

glRotatef(-45, 0, 0, 1);

glScalef(0.3, 0.9, 0.3);

box();

glPopMatrix();

//はら

glPushMatrix();

glTranslatef(-0.1, 2.0, -0.95);

glRotatef(15, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.6, 0.5, 0.4);

box();

glPopMatrix();

//こし

glPushMatrix();

glTranslatef(0.0, 1.4, -0.95);

glRotatef(360, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.9, 0.6, 0.4);

box();

glPopMatrix();

//みぎもも

glPushMatrix();

glTranslatef(-0.55, 0.9, -0.9);

glRotatef(-30, 0, 0, 1);

glRotatef(-40, 1, 0, 0);

glRotatef(45, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//みぎあし

glPushMatrix();

glTranslatef(-0.7, 0.2, -0.7);

glRotatef(360, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-30, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりもも

glPushMatrix();

glTranslatef(0.6, 1.1, -0.55);

glRotatef(45, 0, 0, 1);

glRotatef(-30, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりあし

glPushMatrix();

glTranslatef(1.0, 0.3, -0.3);

glRotatef(10, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//剣

glPushMatrix();

glTranslatef(-1.0, 2.8, -0.2);

glRotatef(0.0, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glRotatef(-50.0, 0, 0, 1);

glScalef(0.1, 2.5, 0.1);

box();

glPopMatrix();

}

else if(state == 50){//弾き

//あたま

glPushMatrix();

glTranslatef(0.1, 2.8, -0.3);

glRotatef(30, 1, 0, 0);

glRotatef(-30, 0, 1, 0);

glScalef(0.4, 0.4, 0.4);

box();

glPopMatrix();

//むね

glPushMatrix();

glTranslatef(0.0, 2.4, -0.8);

glRotatef(20, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(1.0, 0.5, 0.4);

box();

glPopMatrix();

//ひだりうで

glPushMatrix();

glTranslatef(0.4, 2.2, -0.4);

glRotatef(-60.0, 1, 0, 0);

glRotatef(45.0, 0, 1, 0);

glRotatef(-10.0, 0, 0, 1);

glScalef(0.3, 1.0, 0.3);

box();

glPopMatrix();

//ひだりて

glPushMatrix();

glTranslatef(0.3, 2.5, 0.2);

glRotatef(-10.0, 1, 0, 0);

glRotatef(60, 0, 1, 0);

glRotatef(0.0, 0, 0, 1);

glScalef(0.3, 1.1, 0.3);

box();

glPopMatrix();

//みぎうで

glPushMatrix();

glTranslatef(-0.7, 2.7, -0.5);

glRotatef(0.0, 1, 0, 0);

glRotatef(80.0, 0, 1, 0);

glRotatef(80.0, 0, 0, 1);

glScalef(0.3, 0.4, 0.3);

box();

glPopMatrix();

//みぎて

glPushMatrix();

glTranslatef(-0.1, 2.9, -0.1);

glRotatef(0.0, 1, 0, 0);

glRotatef(-20.0, 0, 1, 0);

glRotatef(-80.0, 0, 0, 1);

glScalef(0.3, 0.9, 0.3);

box();

glPopMatrix();

//はら

glPushMatrix();

glTranslatef(0.0, 2.0, -0.95);

glRotatef(15, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.6, 0.5, 0.4);

box();

glPopMatrix();

//こし

glPushMatrix();

glTranslatef(0.0, 1.4, -0.95);

glRotatef(360, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.9, 0.6, 0.4);

box();

glPopMatrix();

//みぎもも

glPushMatrix();

glTranslatef(-0.55, 0.9, -0.9);

glRotatef(-30, 0, 0, 1);

glRotatef(-40, 1, 0, 0);

glRotatef(45, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//みぎあし

glPushMatrix();

glTranslatef(-0.7, 0.2, -0.7);

glRotatef(360, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-30, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりもも

glPushMatrix();

glTranslatef(0.6, 1.1, -0.55);

glRotatef(45, 0, 0, 1);

glRotatef(-30, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりあし

glPushMatrix();

glTranslatef(1.0, 0.3, -0.3);

glRotatef(10, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//剣

glPushMatrix();

glTranslatef(1.0, 2.0, 0.4);

glRotatef(0.0, 1, 0, 0);

glRotatef(-30.0, 0, 1, 0);

glRotatef(30.0, 0, 0, 1);

glScalef(0.1, 2.5, 0.1);

box();

glPopMatrix();

}

else if (state == 51) {//弾き上げ

//あたま

glPushMatrix();

glTranslatef(0.1, 2.8, -0.3);

glRotatef(30, 1, 0, 0);

glRotatef(-30, 0, 1, 0);

glScalef(0.4, 0.4, 0.4);

box();

glPopMatrix();

//むね

glPushMatrix();

glTranslatef(0.0, 2.4, -0.8);

glRotatef(20, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(1.0, 0.5, 0.4);

box();

glPopMatrix();

//ひだりうで

glPushMatrix();

glTranslatef(0.5, 2.4, -0.4);

glRotatef(70, 1, 0, 0);

glRotatef(30.0, 0, 1, 0);

glRotatef(-10.0, 0, 0, 1);

glScalef(0.35, 1.0, 0.35);

box();

glPopMatrix();

//ひだりて

glPushMatrix();

glTranslatef(0.6, 3.2, 0.2);

glRotatef(-10.0, 1, 0, 0);

glRotatef(60, 0, 1, 0);

glRotatef(0.0, 0, 0, 1);

glScalef(0.3, 1.1, 0.3);

box();

glPopMatrix();

//みぎうで

glPushMatrix();

glTranslatef(-0.4, 3.0, -0.5);

glRotatef(-45.0, 1, 0, 0);

glRotatef(80.0, 0, 1, 0);

glRotatef(80.0, 0, 0, 1);

glScalef(0.3, 1.3, 0.3);

box();

glPopMatrix();

//みぎて

glPushMatrix();

glTranslatef(-0.1, 3.5, -0.1);

glRotatef(0.0, 1, 0, 0);

glRotatef(-20.0, 0, 1, 0);

glRotatef(-80.0, 0, 0, 1);

glScalef(0.3, 1.1, 0.3);

box();

glPopMatrix();

//はら

glPushMatrix();

glTranslatef(0.0, 2.0, -0.95);

glRotatef(15, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.6, 0.5, 0.4);

box();

glPopMatrix();

//こし

glPushMatrix();

glTranslatef(0.0, 1.4, -0.95);

glRotatef(360, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.9, 0.6, 0.4);

box();

glPopMatrix();

//みぎもも

glPushMatrix();

glTranslatef(-0.55, 0.9, -0.9);

glRotatef(-30, 0, 0, 1);

glRotatef(-40, 1, 0, 0);

glRotatef(45, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//みぎあし

glPushMatrix();

glTranslatef(-0.7, 0.2, -0.7);

glRotatef(360, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-30, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりもも

glPushMatrix();

glTranslatef(0.6, 1.1, -0.55);

glRotatef(45, 0, 0, 1);

glRotatef(-30, 1, 0, 0);

glRotatef(0, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//ひだりあし

glPushMatrix();

glTranslatef(1.0, 0.3, -0.3);

glRotatef(10, 0, 0, 1);

glRotatef(360, 1, 0, 0);

glRotatef(-20, 0, 1, 0);

glScalef(0.4, 1.0, 0.4);

box();

glPopMatrix();

//剣

glPushMatrix();

glTranslatef(1.5, 3.5, 0.4);

glRotatef(0.0, 1, 0, 0);

glRotatef(-30.0, 0, 1, 0);

glRotatef(80.0, 0, 0, 1);

glScalef(0.1, 2.5, 0.1);

box();

glPopMatrix();

}

else if (state != 10000) {

glBegin(GL\_POLYGON);

glNormal3f(0, 1, 0);

glVertex3f(0.5, 4, 0.5);

glVertex3f(0.5, 4, -0.5);

glVertex3f(-0.5, 2, -0.5);

glVertex3f(-0.5, 2, 0.5);

glEnd();

glBegin(GL\_POLYGON);

glNormal3f(0, -1, 0);

glVertex3f(0.5, 0, 0.5);

glVertex3f(0.5, 0, -0.5);

glVertex3f(-0.5, 0, -0.5);

glVertex3f(-0.5, 0, 0.5);

glEnd();

glBegin(GL\_QUAD\_STRIP);

glNormal3f(0, 0, 1);

glVertex3f(-0.5, 3, 0.5);

glVertex3f(-0.5, 0, 0.5);

glVertex3f(0.5, 3, 0.5);

glVertex3f(0.5, 0, 0.5);

glNormal3f(1, 0, 0);

glVertex3f(0.5, 3, -0.5);

glVertex3f(0.5, 0, -0.5);

glNormal3f(0, 0, -1);

glVertex3f(-0.5, 3, -0.5);

glVertex3f(-0.5, 0, -0.5);

glNormal3f(-1, 0, 0);

glVertex3f(-0.5, 3, 0.5);

glVertex3f(-0.5, 0, 0.5);

glEnd();

}

}

void clock\_needle(void)

{

glBegin(GL\_POLYGON);

glNormal3f(0, 0, 11);

glVertex3f(0, 0.5, 1);

glVertex3f(0.1, -0.1, 1);

glVertex3f(0, 0, 1);

glVertex3f(-0.1, -0.1, 1);

glEnd();

}

void display(void)

{

glClearColor(0.0, 0.0, 0.0, 1.0);

glMatrixMode(GL\_PROJECTION);//視点

glLoadIdentity();

gluPerspective(45, 2.0, 5, -40);

gluLookAt(0, 3, 10, 0, 2, 0, 0, 1, 0);

//glOrtho(-6, 6, -1, 5, -40, 5.0);

glShadeModel(GL\_FLAT);//光源関係

glEnable(GL\_LIGHT0);

glEnable(GL\_LIGHTING);

float position[] = { 0.0, 6.0, 10.0, 1.0 };

glLightfv(GL\_LIGHT0, GL\_POSITION, position);

glLightfv(GL\_LIGHT0, GL\_DIFFUSE, amber);

glLightfv(GL\_LIGHT0, GL\_AMBIENT, amber);

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glEnable(GL\_DEPTH\_TEST);

glColor3f(1.0, 1.0, 1.0);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glPushMatrix();

glTranslatef(5, 4, 0.0);

glRotatef(RotAngle, 0, 0, -1);

glScalef(1, 1, 1);

clock\_needle();

glPopMatrix();

//自分のキャラ

glPushMatrix();

glTranslatef(my\_location, 0, 0);

human(my\_state);

glPopMatrix();

//敵キャラ

glPushMatrix();

glTranslatef(enemy\_location, 0, 0);

glRotatef(180, 0, 1, 0);

human(enemy\_state);

glPopMatrix();

glFlush();

}

void game\_contents();

void IncAngle(int TimerNumber) {

if (judge != 0)

TimerNumber = 2;

if (TimerNumber == 1) {

setting = false;//難易度設定を終える

//時計を回転させる

RotAngle = RotAngle + 30.0;

if (RotAngle >= 360.0)

RotAngle = RotAngle - 360.0;

//動きを管理

my\_state = my\_state\_next;

enemy\_state = enemy\_state\_next;

if (my\_state == 0) {//入力受け付け

if (input == 'd') {

my\_state = 1;

}

else if (input == 'a') {

my\_state = 3;

}

else if (input == 'j') {

my\_state = 20;

}

else if (input == 'k') {

my\_state = 30;

}

else if (input == 'l') {

my\_state = 40;

}

else if (input == 'w') {

my\_state = 50;

}

}

input = 's';

if (enemy\_state == 0) {//敵のコマンド入力

srand((unsigned int)time(NULL));

int random = rand() \*7 % 100;

if (level == 0) {

enemy\_state = 0;

}

else if (level == 1) {

if (my\_state == 21 || my\_state == 31) {

if (random < 30)

enemy\_state = 50;

}

else if (random < 40) {

}

else if (random < 55) {

enemy\_state = 20;

}

else if (random < 60) {

enemy\_state = 30;

}

else if (random < 75) {

enemy\_state = 1;

}

else {

enemy\_state = 3;

}

}

else if (level == 2) {

if (my\_state == 21 || my\_state == 31) {

if (random < 60)

enemy\_state = 50;

}

else if (random < 30) {

}

else if (random < 60) {

enemy\_state = 20;

}

else if (random < 80) {

enemy\_state = 30;

}

else if (random < 90) {

enemy\_state = 1;

}

else {

enemy\_state = 3;

}

}

else if (level == 3) {

if (my\_state == 21 || my\_state == 31) {

if (random < 95)

enemy\_state = 50;

}

else if (my\_state == 41) {

if (random < 70)

enemy\_state = 20;

}

else if (random < 55) {

}

else if (random < 65) {

enemy\_state = 20;

}

else if (random < 75) {

enemy\_state = 30;

}

else if (random < 85) {

enemy\_state = 40;

}

else if (random < 95) {

enemy\_state = 1;

}

else {

enemy\_state = 3;

}

}

}

game\_contents();//ゲームの動作を計算

input = 's';//input をニュートラルに戻す

glutTimerFunc(T, IncAngle, 1);//自分自身を再設定してタイマー起動

glutPostRedisplay();//再描画

}

else if (TimerNumber == 2) {//勝敗決定

if (judge == 1) {

amber[2] = 0.0;

}

else if (judge == 2) {

amber[0] = 0.0;

}

amber[1] = 0.0;

glutPostRedisplay();//再描画

}

else if (TimerNumber == 3) {//最初の難易度設定

input = 's';//input をニュートラルに戻す

glutTimerFunc(T, IncAngle, 1);//タイマー１を設定してタイマー起動

glutPostRedisplay();//再描画

}

}

void game\_contents() {

//現在のstateから行動を決定

if (my\_state == 1) {//前移動１

my\_state\_next = 0;

if (my\_location < enemy\_location - 1)

my\_location++;

}

else if (my\_state == 3) {//後移動３

my\_state\_next = 0;

if (my\_location > -5)

my\_location--;

}

else if (my\_state == 20) {//攻撃その１ 20～

my\_state\_next = 21;

}

else if (my\_state == 21) {

my\_state\_next = 22;

my\_attack = 1;

}

else if (my\_state == 22) {

my\_state\_next = 0;

my\_attack = 0;

}

else if (my\_state == 30) {//攻撃その２　30～

my\_state\_next = 31;

if (my\_location < enemy\_location - 1)

my\_location++;

}

else if (my\_state == 31) {

my\_state\_next = 32;

my\_attack = 1;

}

else if (my\_state == 32) {

my\_state\_next = 33;

my\_attack = 0;

}

else if (my\_state == 33) {

my\_state\_next = 0;

}

else if (my\_state == 40) {//攻撃その３　40～

my\_state\_next = 41;

}

else if (my\_state == 41) {

my\_state\_next = 42;

}

else if (my\_state == 42) {

my\_state\_next = 43;

my\_attack = 100;

}

else if (my\_state == 43) {

my\_state\_next = 0;

my\_attack = 0;

}

else if (my\_state == 50) {//弾き　50～

my\_state\_next = 51;

my\_attack = 2;

}

else if (my\_state == 51) {

my\_state\_next = 0;

my\_attack = 0;

}

else if (my\_state == 60) {//弾かれ　60～

my\_state\_next = 61;

my\_attack = 0;

}

else if (my\_state == 61) {

my\_state\_next = 62;

if (my\_location > -5)

my\_location--;

}

else if (my\_state == 62) {

my\_state\_next = 0;

if (my\_location > -5)

my\_location--;

}

//現在のstateから相手の行動を決定

if (enemy\_state == 1) {//前移動１

enemy\_state\_next = 0;

if (my\_location < enemy\_location - 1)

enemy\_location--;

}

else if (enemy\_state == 3) {//後移動３

enemy\_state\_next = 0;

if (enemy\_location < 5)

enemy\_location++;

}

else if (enemy\_state == 20) {//攻撃その１ 20～

enemy\_state\_next = 21;

}

else if (enemy\_state == 21) {

enemy\_state\_next = 22;

enemy\_attack = 1;

}

else if (enemy\_state == 22) {

enemy\_state\_next = 0;

enemy\_attack = 0;

}

else if (enemy\_state == 30) {//攻撃その２　30～

enemy\_state\_next = 31;

if (my\_location < enemy\_location - 1)

enemy\_location--;

}

else if (enemy\_state == 31) {

enemy\_state\_next = 32;

enemy\_attack = 1;

}

else if (enemy\_state == 32) {

enemy\_state\_next = 33;

enemy\_attack = 0;

}

else if (enemy\_state == 33) {

enemy\_state\_next = 0;

}

else if (enemy\_state == 40) {//攻撃その３　40～

enemy\_state\_next = 41;

}

else if (enemy\_state == 41) {

enemy\_state\_next = 42;

}

else if (enemy\_state == 42) {

enemy\_state\_next = 43;

enemy\_attack = 100;

}

else if (enemy\_state == 43) {

enemy\_state\_next = 0;

enemy\_attack = 0;

}

else if (enemy\_state == 50) {//弾き　50～

enemy\_state\_next = 51;

enemy\_attack = 2;

}

else if (enemy\_state == 51) {

enemy\_state\_next = 0;

enemy\_attack = 0;

}

else if (enemy\_state == 60) {//弾かれ　60～

enemy\_state\_next = 61;

enemy\_attack = 0;

}

else if (enemy\_state == 61) {

enemy\_state\_next = 62;

if (enemy\_location < 5)

enemy\_location++;

}

else if (enemy\_state == 62) {

enemy\_state\_next = 0;

if (enemy\_location < 5)

enemy\_location++;

}

//判定計算

if (my\_attack == 100) {

if (enemy\_attack == 1) {//自分攻撃３　相手攻撃

if (my\_location + 3 >= enemy\_location - 3) {

my\_state\_next = 0;//ニュートラルモーションに移行

my\_attack = 0;

enemy\_state\_next = 60;//弾かれモーションに移行

}

}

else if (enemy\_attack == 100) {

if (my\_location + 3 >= enemy\_location - 3) {

my\_state\_next = 60;//弾かれモーションに移行

enemy\_state\_next = 60;//弾かれモーションに移行

}

}

else {

if (my\_location + 3 >= enemy\_location) {

judge = 1;

}

}

}

else if (my\_attack == 1) {

if (enemy\_attack == 1) {//自分攻撃　相手攻撃

if (my\_location + 3 >= enemy\_location - 3) {

my\_state\_next = 60;//弾かれモーションに移行

enemy\_state\_next = 60;//弾かれモーションに移行

}

}

else if (enemy\_attack == 2) {//自分攻撃　相手弾き

if (my\_location + 3 >= enemy\_location) {

my\_state\_next = 60;//弾かれモーションに移行

}

}

else if (enemy\_attack == 0) {//自分攻撃　相手無防備

if (my\_location + 3 >= enemy\_location) {

judge = 1;//動けないモードに移行

}

}

else if (enemy\_attack == 100) {//自分攻撃　相手攻撃３

if (my\_location + 3 >= enemy\_location - 3) {

my\_state\_next = 60;

enemy\_state\_next = 0;

enemy\_attack = 0;

}

}

}

else if (my\_attack == 2) {

if (enemy\_attack == 1) {//自分弾き　相手攻撃

if (my\_location >= enemy\_location - 3) {

enemy\_state\_next = 60;//弾かれモーションに移行

}

}

else if (enemy\_attack == 100) {

if (my\_location >= enemy\_location - 3) {

judge = 2;

}

}

}

else if (enemy\_attack == 1) {//自分無防備　相手攻撃

if (my\_location >= enemy\_location - 3) {

judge = 2;//動けないモードに移行

}

}

}

int main(int argc, char\*\* argv)

{

glutInit(&argc, argv);

glutInitWindowPosition(0, 0);

glutInitWindowSize(800, 400);

glutInitDisplayMode(GLUT\_RGBA | GLUT\_DEPTH);

glutCreateWindow("間合イ");

glutDisplayFunc(display);

glutKeyboardFunc(KeyboardHandler);

glutTimerFunc(set, IncAngle, 3);//タイマーハンドラの初回の設定と起動

glutMainLoop();

}