デジタルコンテンツ１

最終レポート

メンバー

後藤岬：対象の動作プログラム

井上文陽：メインプログラム

青野和真：計算処理プログラム

作品の目的

昨今の学生の趣味の上位にはテレビゲームが入っていることが多いと思います。そこで、テレビゲームに学習要素を組み込めば受け入れられると判断し、計算シューティングを作ることにしました。シューティングに至ったのは、100 枡計算で養うような、計算力を育てるために、早く解かないと敵が接近してくる、といった緊張感を持って取り組めるようなゲームにしたいと考えました。

難易度の設定

easy

一桁の足し算引き算に加えて、九九も行う難易度。

ここまではどの敵も一撃でで倒せる。

normal

二桁までの足し算引き算、掛け算の計算を含めたもの。ここからはボス限定で、一度の攻撃だけでは倒せない。

mania

三桁の足し算引き算に加えて三桁のかけ算を行う。ボスは当然の如く、通常キャラも一撃では倒せない。解けるものなら解いてみろ！といった難易度。

オリジナルの要素

問題を解く、敵をクリックする。この２つの要素のうち、片方をこなしただけではこのゲームを進めることができないことです。問題を解くと同時に、敵をクリックして攻撃する。この２つの要素を同時進行することで、ゲームをこなすのに必要な計算力が養われていきます。

プログラム説明

Class1:game

他のクラスを引っ張ってきて実行するためのクラスで、加えて、メニュー画面、キーボード操作にを行う、プログラム全体をまとめるクラスです。

```
int ans;
int p, stack;
String num = "";
int rate = 60;
num_show number;
enemy e1;
//num_res result;
void setup() {
  size(300, 300);
  number = new num_show();//下画面に数字を表示
  ans = number.next();
  e1 = new enemy(random(50, 200), random(50, 200), 0.3);//キャラクターの表示,攻撃判定(hit 関数)

  //result = new num_res();
  //time();
  p = 0;//得点：point[これによりキャラクターの出す量を決める]
  stack = 0;//弾数
  frameRate(rate);
}
void draw() {
  fill(0);
  background(255);
  textAlign(CENTER);
  textSize(20);
  e1.display();
  e1.move();
  number.display();
  text(num, width*2/3, height*3/4);
  text("hit="+p, width*2/3, height*6/7);
  if (stack>0) {
```

```
    text("stack="+stack, width*2/3, height*3/4+50);
  }
  else {
    text("enpty", width*2/3, height*3/4+50);
  }
}

void keyPressed() {
  number.check(key);
}
void mousePressed() {
  e1.hit();
}
```

Class2:enemy

このクラスでは、敵オブジェクトのサイズの変化、出現位置、ゲームの終了条件、敵を撃退するための行動を行うクラスになっています。

```
class enemy {
  float x, y;
  float speed;
  float w=0.1;
  PImage img;
  enemy(float x0, float y0, float sp) {
    img = loadImage("mzl_oofbzymj.jpg");//敵オブジェクトの画像を読み込む
    x = x0;
    y = y0;
    speed = sp;
  }
  void display() {
    fill(0,255,255);
    image(img,x,y,x,y);
    fill(0);
    text(100-(int)w,x,y);//残り時間のカウントダウン表示
    if((int)w>100){
      damage();
    }
```

```
}
void move() {
   w=w+speed;
}
void Restart(){//敵オブジェクト破壊時に次の敵が出てくる
   w=0.1;
   x=random(50,200);
   y=random(50,200);
}
float sizex1(){//ここから敵オブジェクトのサイズ
   return x-w;
}
float sizex2(){
   return x+w;
}
float sizey1(){
   return y-w;
}
float sizey2(){
   return y+w;
}
void damage(){
   Restart();
   text("OUT!!",width/2,height/2);//敵と接触してゲームオーバーになる
   noLoop();
}
 void hit(){
   if (stack != 0) {//プレイヤーの残弾が複数発あるとき
      if (mouseX>=e1.sizex1()&&
         mouseX<=e1.sizex2()&&
         mouseY>=e1.sizey1()&&
         mouseY<=e1.sizey2())
      {
         stack--;
         p++;
         e1.Restart();
```

```
            text("hit!!", width/2, height/2);//敵に攻撃した時のエフェクト
          }
        }
    }
}
```

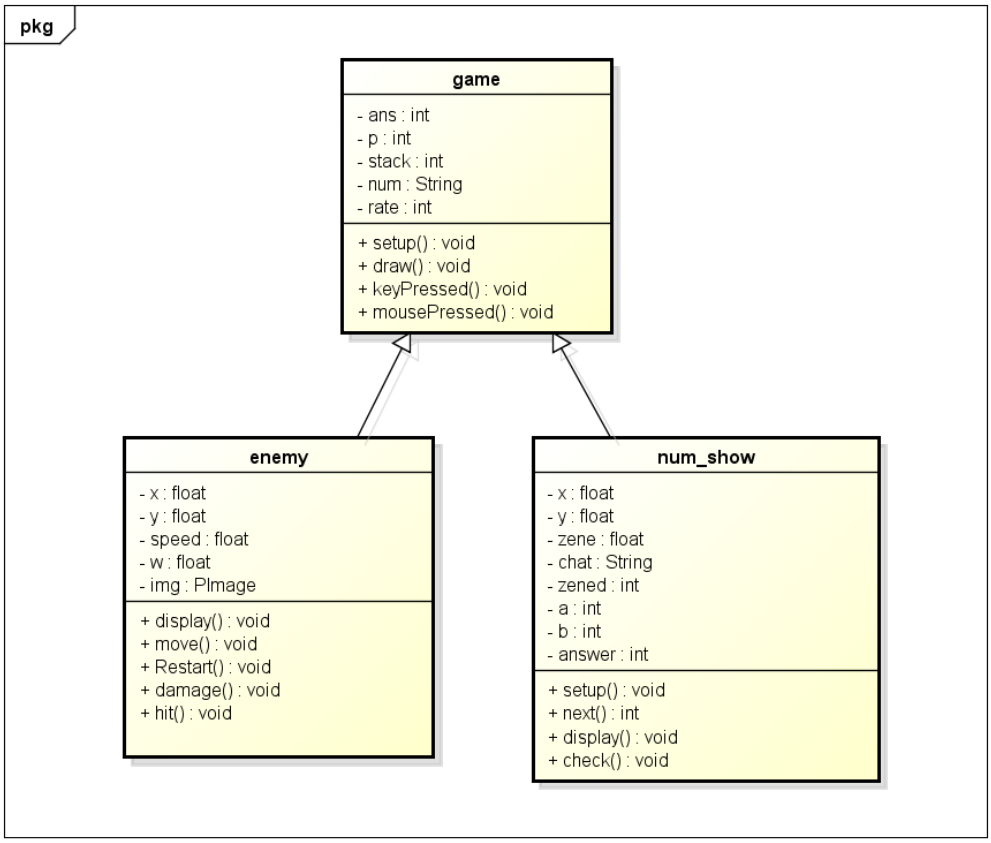
Class3:num_show

このクラスでは計算問題を難易度ごとにランダムで作成し、画面に表示するプログラムのクラスとなっています。

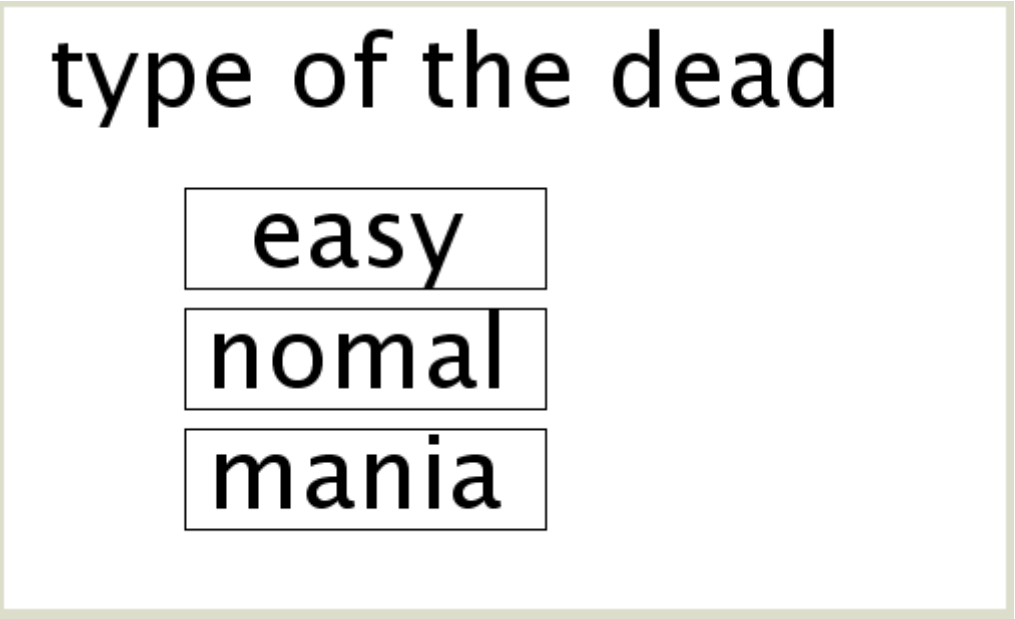
```
class num_show {
    float x;
    float y;
    float zene;
    String[] chat = {
        "+", "-"
    };
    int zened;
    int a;
    int b;
    int answer;

    void setup() {
        next();
        display();
    }

    int next() {
        x = random(0, 10);
        y = random(0, 10);
        zene = random(0, 2);
        zened = int(zene);
        a = int(x);
        b = int(y);
        if (zened == 0) {
            answer = a+b;
        }
        else if (zened == 1) {
```

```
            answer = a-b;
        }
        return answer;
    }
    void display() {
        fill(0);
        textAlign(CENTER);
        text(a + chat[zened] + b + "=", width/2, height*3/4);
    }
         void check(char key){
        if (key == ENTER) {
            if (int(num) == ans) {
                ans = number.next();
                stack++;
                //e1.Restart();
            }
            num = "";
        }
        else if (key == BACKSPACE) {
            num = "";
        }
        else {
            num = num + str(key);
        }
    }
}
```

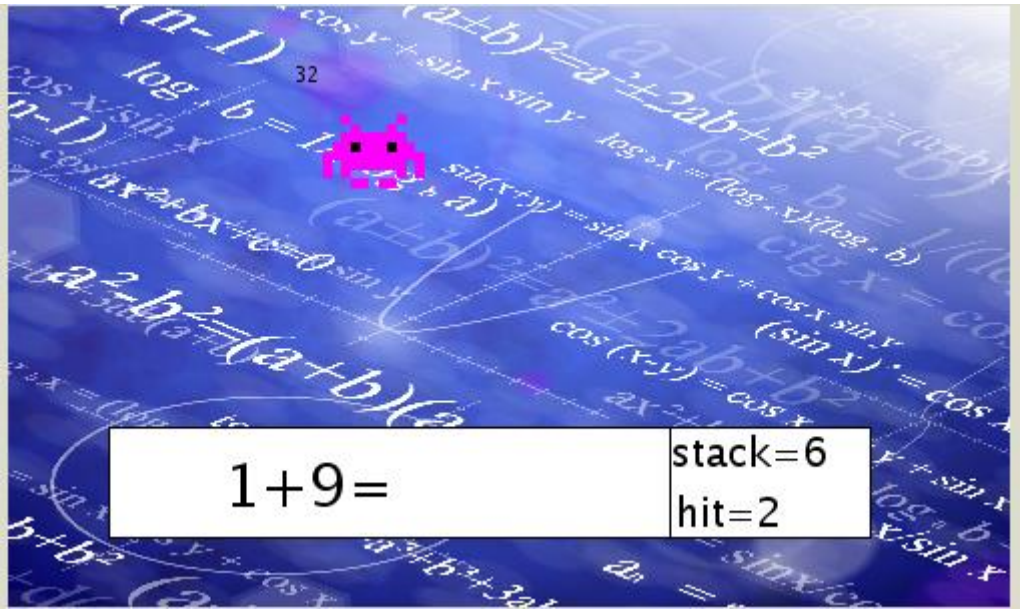
クラス図


pkg
game
- ans : int
- p : int
- stack : int
- num : String
- rate : int
+ setup() : void
+ draw() : void
+ keyPressed() : void
+ mousePressed() : void
enemy
- x : float
- y : float
- speed : float
- w : float
- img : PImage
+ display() : void
+ move() : void
+ Restart() : void
+ damage() : void
+ hit() : void
num_show
- x : float
- y : float
- zene : float
- chat : String
- zened : int
- a : int
- b : int
- answer : int
+ setup() : void
+ next() : int
+ display() : void
+ check() : void
powered by Astah

実行の様子



メニュー画面です。今回は easy を選択しました。



ゲーム画面です。計算画面、敵オブジェクトの表示は正常、敵の上に表示されるカウントダウンも予定どうりに表示されました。プレイヤーはキーボードで計算を行い、マウスクリックで敵を撃退します。

敵に接触してゲームオーバーになった時の画面です。

ボスを含めてすべての敵を倒してステージクリアした時の画面です。

使用結果と今後の課題

当初の予定どうり計算とシューティングを同時にこなすといったゲームを作ることができました。ゲームオーバーになるときも、より進めそうだと思わせることで、何回もチャレンジしたくなるようなゲームになったと思います。時間があれば、BGM を加えたり、敵オブジェクトの動きをもう少し多彩にできれば良かったと思います。