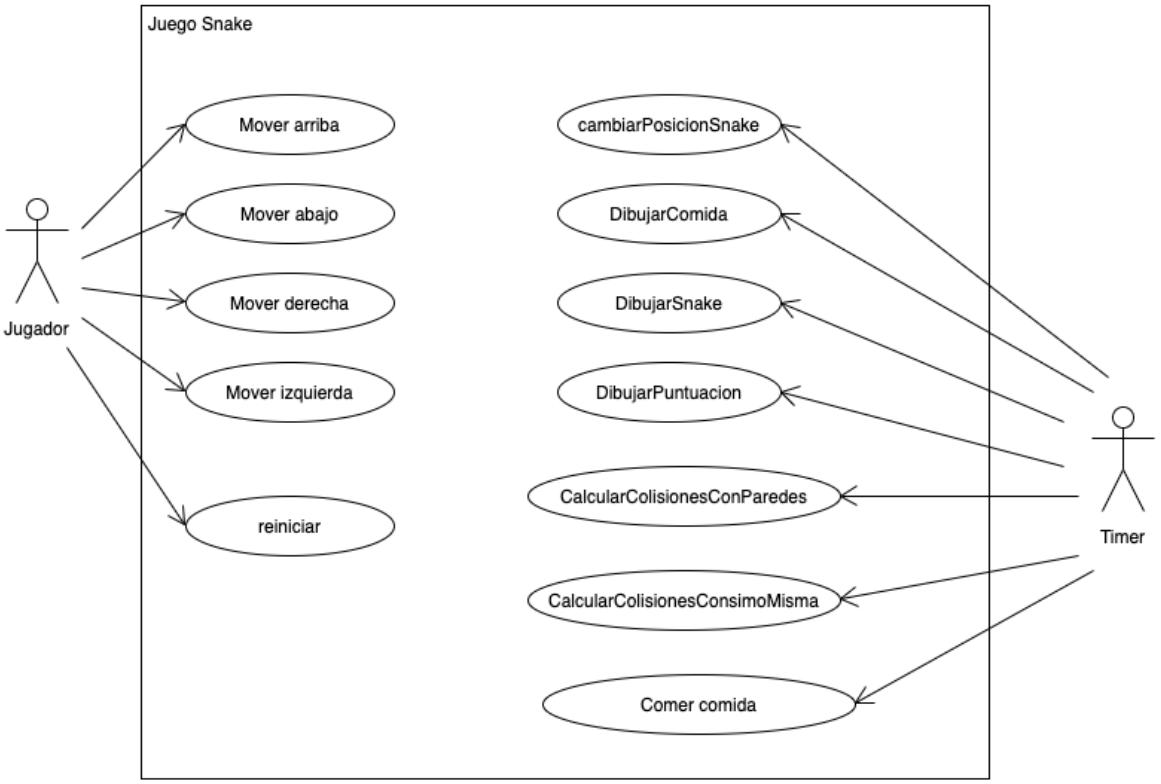
Ejercicio 2: Juego Snake

Puedes ver los ficheros en la carpeta compartida:

Entornos de desarrollo-->Juegos clasicos-->Snake

Parte 1: Casos de uso

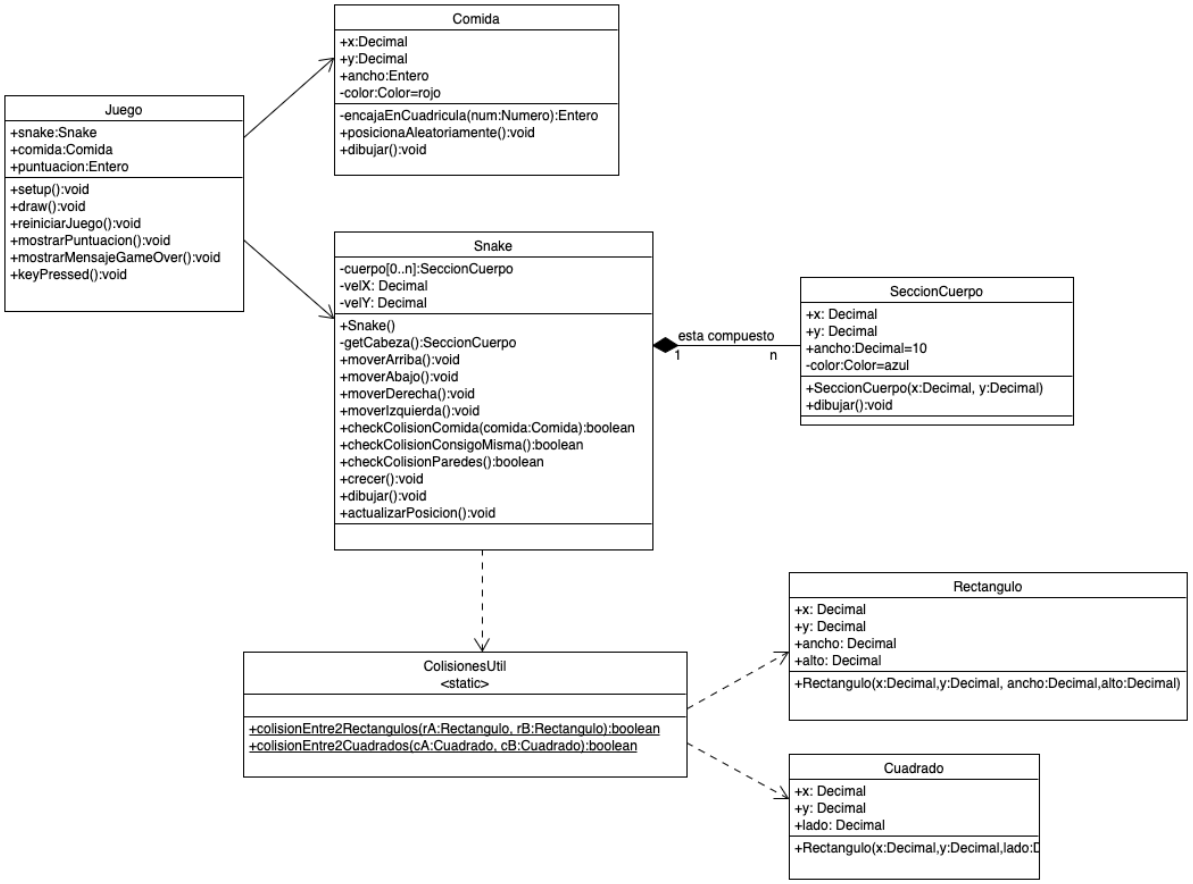
Realiza el diagrama de casos de uso del juego Snake.
Solución:



Parte 2: Diagrama clases

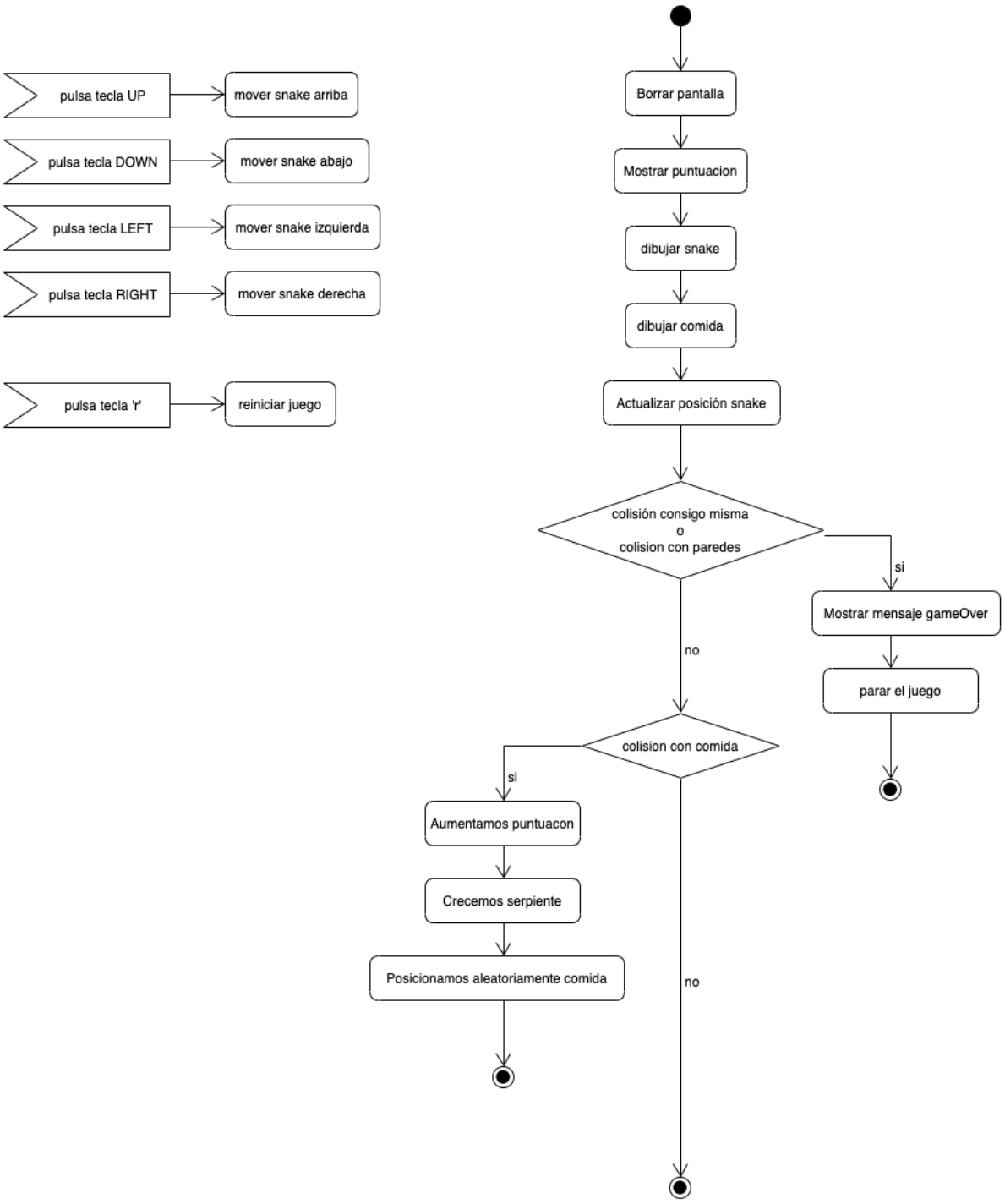
Construye el diagrama de clases UML

Solución:



Parte 3: Diagrama de actividad

Construye el diagrama uml de actividad (solamente del programa principal).
Esto se correspondería con la actividad que se realiza en 1 fotograma


pulsa tecla UP
mover snake arriba
pulsa tecla DOWN
mover snake abajo
pulsa tecla LEFT
mover snake izquierda
pulsa tecla RIGHT
mover snake derecha
pulsa tecla 'r'
reiniciar juego
Borrar pantalla
Mostrar puntuacion
dibujar snake
dibujar comida
Actualizar posición snake
colisión consigo misma
o
colision con paredes
si
Mostrar mensaje gameOver
parar el juego
no
colision con comida
si
Aumentamos puntuacon
Crecemos serpiente
Posicionamos aleatoriamente comida
no

Parte 4: Código fuente

Codifica el código fuente usando el lenguaje Java y la librería gráfica Processing/P5

```
static class Rectangulo {
  float x;
  float y;
  float ancho;
  float alto;
  public Rectangulo(float x, float y, float ancho, float alto) {
    this.x = x;
    this.y = y;
    this.ancho = ancho;
    this.alto = alto;
  }
}
static class Cuadrado {
  float x;
  float y;
  float lado;
  public Cuadrado(float x, float y, float lado) {
    this.x = x;
    this.y = y;
    this.lado = lado;
  }
}
static class ColisionesUtil {
  /*
Lado derecho de r1 es mayor que lado izquierdo de r2
   Lado izquierdo de r1 es menor que lado derecho de r2
   Lado superior de r1 es mayor que lado inferior de r2
   Lado inferior de r1 es menor que lado superior de r2
   Si se cumplen estas cuatro condiciones es que ambos rectángulos están
colisionando.
   */
  public static  boolean colisionEntre2Rectangulos(Rectangulo rA,
Rectangulo rB) {
    float left = rA.x;
    float right = rA.x + rA.ancho;
    float top = rA.y;
    float bottom = rA.y + rA.alto;
    float r_left = rB.x;
    float r_right = rB.x + rB.ancho;
    float r_top = rB.y;
```

```
    float r_bottom = rB.y+ rB.alto;

    if ( right <= r_left ||  left >= r_right) {
      return false;
    } else if ( bottom <= r_top || top >= r_bottom) {
      return false;
    }

    return true;
  }

  static public  boolean colisionEntre2Cuadrados(Cuadrado cA, Cuadrado
cB) {
    Rectangulo rA = new Rectangulo(cA.x, cA.y, cA.lado, cA.lado);
    Rectangulo rB = new Rectangulo(cB.x, cB.y, cB.lado, cB.lado);
    return colisionEntre2Rectangulos(rA, rB);
  }
}

class Comida {
  private float x;
  private float y;
  private int ancho=10;
  private color col = color(255, 0, 0);
  Comida() {
  }

  private int encajaEnCuadricula(float num) {
    return (int)(Math.round(num/ancho) * ancho);
  }

  public void posicionaAleatoriamente() {
    x=(int)random(0, width-ancho*2);
    x=encajaEnCuadricula(x);
    y=(int)random(0, height-ancho*2);
    y=encajaEnCuadricula(y);
  }

  public void dibujar() {
    fill(col);
    rect(x, y, ancho, ancho);
  }
}

class SeccionCuerpo {
```

```
  public float x;
  public float y;
  private int ancho=10;
  private color col = color(0, 0, 255);

  public SeccionCuerpo(float x, float y) {
    this.x=x;
    this.y=y;
  }

  public void dibujar() {
    fill(col);
    rect(x, y, ancho, ancho);
  }
}

class Snake {
  private ArrayList<SeccionCuerpo> cuerpo = new
ArrayList<SeccionCuerpo>();
  private float velX;
  private float velY;

  public Snake() {
    cuerpo.add(new SeccionCuerpo(width/2, height/2));
    velX=velY=0;
  }

  private SeccionCuerpo getCabeza() {
    return cuerpo.get(0);
  }

  public void moverArriba() {
    velX=0;
    velY=-10f;
  }
  public void moverAbajo() {
    velX=0;
    velY=10f;
  }
  public void moverDerecha() {
    velX=10;
    velY=0;
  }
  public void moverIzquierda() {
    velX=-10f;
```

```
    velY=0;
  }

  public boolean checkColisionConComida(Comida comida) {
    Cuadrado cuadradoSnake = new Cuadrado(getCabeza().x, getCabeza().y,
getCabeza().ancho);
    Cuadrado cuadradoComida = new Cuadrado(comida.x, comida.y,
comida.ancho);
    return ColisionesUtil.colisionEntre2Cuadrados(cuadradoSnake,
cuadradoComida);
  }

  public boolean checkColisionConsigoMisma() {
    SeccionCuerpo cabeza = getCabeza();
    Cuadrado cuadradoCabeza = new Cuadrado(cabeza.x, cabeza.y,
cabeza.ancho);

    for (SeccionCuerpo seccion : cuerpo) {
      if (seccion == cabeza) continue;
      Cuadrado cuadradoSeccion = new Cuadrado(seccion.x, seccion.y,
seccion.ancho);

      if ( ColisionesUtil.colisionEntre2Cuadrados(cuadradoCabeza,
cuadradoSeccion) ) {
        return true;
      }
    }
    return false;
  }

  public boolean checkColisionParedes() {
    return getCabeza().x>width || getCabeza().x<0 ||
getCabeza().y>height || getCabeza().y<0;
  }

  public void crecer() {
    cuerpo.add(new SeccionCuerpo(getCabeza().x, getCabeza().y) );
  }

  public void dibujar() {
    for (SeccionCuerpo s : cuerpo) {
      s.dibujar();
    }
  }
```

```
  public void actualizarPosicion() {
    for (int i=cuerpo.size()-1; i>0; i--) {
      cuerpo.get(i).x = cuerpo.get(i-1).x;
      cuerpo.get(i).y = cuerpo.get(i-1).y;
    }

    getCabeza().x += velX;
    getCabeza().y += velY;
  }
}



Snake snake;
Comida comida;
int puntuacion=0;

void setup() {
  size(200, 200);
  frameRate(14);
  snake = new Snake();
  comida = new Comida();
  comida.posicionaAleatoriamente();
}

void draw() {
  background(0);

  mostrarPuntuacion();

  snake.dibujar();
  comida.dibujar();

  snake.actualizarPosicion();


  if (snake.checkColisionConsigoMisma() || snake.checkColisionParedes())
{
    mostrarMensajeGameOver();
    noLoop();
  }

  if (snake.checkColisionConComida(comida)) {
    puntuacion++;
    snake.crecer();
```

```
    comida.posicionaAleatoriamente();
  }
}

void reiniciarJuego() {
  puntuacion = 0;
  snake = new Snake();
  comida = new Comida();
  comida.posicionaAleatoriamente();
  loop();
}

void mostrarPuntuacion() {
  fill(0, 255, 0);
  textSize(32);
  text(puntuacion, width/2, 35);
}

void mostrarMensajeGameOver() {
  fill(255, 0, 0);
  textSize(25);
  String frase = "GAME OVER";
  text(frase, (width-textWidth(frase))/2, height/2);
}


void keyPressed() {
  if (keyCode == UP) {
    snake.moverArriba();
  } else if (keyCode == DOWN) {
    snake.moverAbajo();
  } else if (keyCode == RIGHT) {
    snake.moverDerecha();
  } else if (keyCode == LEFT) {
    snake.moverIzquierda();
  } else if (key == 'r' || key == 'R') {
    reiniciarJuego();
  } else if (key == ' ') {
    puntuacion++;
    snake.crecer();
  }
}
```