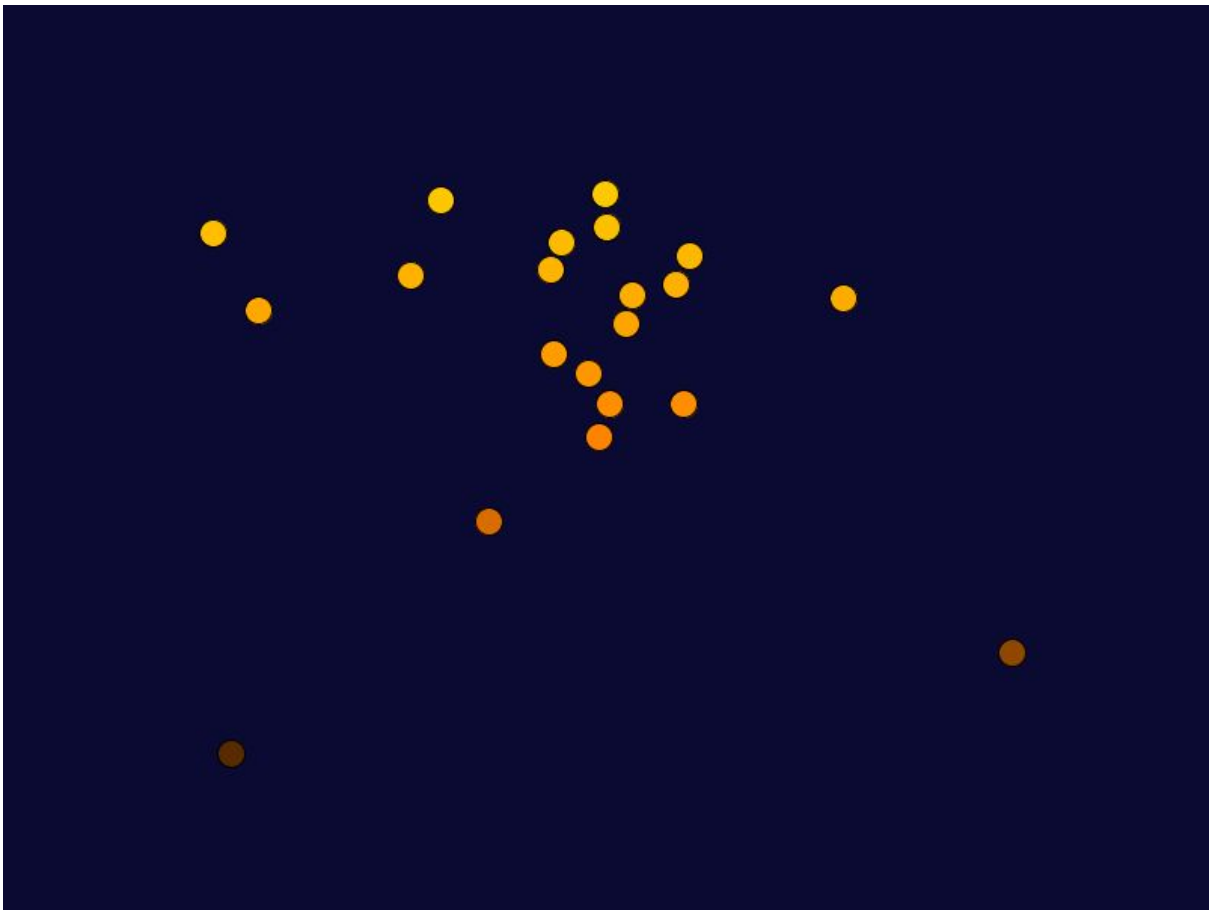
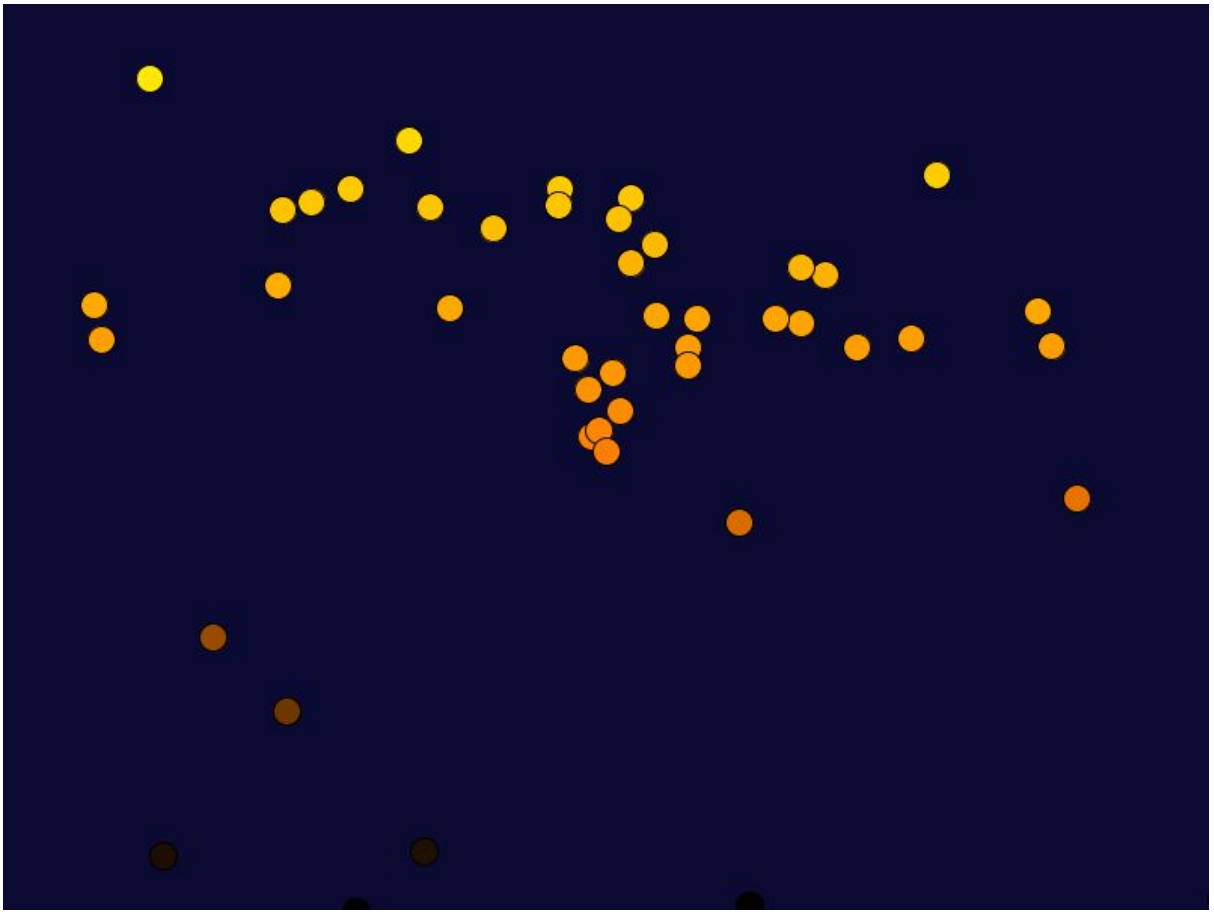


```
float ms;           // millis()
float t;            // time value that restarts each run
float r = 9;        // radius
float spawnTimer;         // used together with objPerSeconds
float objPerSeconds = 12;  // how many objects will be spawn per seconds
float velWeight = 3.5;     // the weight of velocity vector
float acceleration = 0.05; // acceleration y-axis
float wind = -0.015;          // wind
ArrayList<PhysOb> object;  // array of objects

void setup() {
  size(800, 600);
  object = new ArrayList<PhysOb>();
  object.add(new PhysOb(new PVector(width/2, height/2)));
  ms = millis();
}

void draw() {
  background(10, 10, 50);
  t = millis()-ms; // resets the timer
```

```
  t = t/1000;      // to seonds
  ms = millis();   // stores the ms to next draw
  spawnTimer += t*objPerSeconds;

  // launch each objPerSeconds
  if(spawnTimer >= 1){
    object.add(new PhysOb(new PVector(width/2, height/2))); // creates new object
    spawnTimer = 0;
  }

  // loops through all objects on screen
  for(int i = 0; i<object.size(); i++){
    PhysOb obj = object.get(i);
    obj.update();  // movement
    obj.display(); // draw the object

    // deletes the object if its position is under the ground
    if(obj.checkHeight()){
      object.remove(i);
    }
  }
}

class PhysOb  {
  PVector pos;
  PVector vel;
  PVector acc;

  // constructor
  PhysOb (PVector position) {
    acc = new PVector(wind, acceleration);
    float degree = random(-135, -45); // random degree
    float radians = degree * PI/180;  // degree to radians
    vel = new PVector(cos(radians), sin(radians));
    vel.mult(velWeight);
    pos = position;
  }

  void update() {
    vel.add(acc.mult(1+t));
    pos.add(vel.mult(1+t));
  }

  void display() {
    // color goes from orange to black based on pos.y
    fill( ((height-pos.y)/height/2)*4*255, ((height-pos.y)/height/2)*2*255, 0);
```

```
    ellipse(pos.x, pos.y, r*2, r*2);
  }

  boolean checkHeight() {
    if (pos.y > height+r) {
      return true;
    } else {
      return false;
    }
  }
}

void keyPressed() {
  save("myimage.png");
}
```



```
float ms;                   // millis()
float t;                  // time value that restarts each run
float r = 9;                // radius
float spawnTimer;             // used together with objPerSeconds
```

```
float objPerSeconds = 20;      // how many objects will be spawn per seconds
float velWeight = 3.5;         // the weight of velocity vector
float acceleration = 0.05;     // acceleration y-axis
float wind = -0.015;           // wind
int explodeChance = 6;         // chance of explode in percent
int ballsFromExplosion = 10;   // number of balls spawn from explosion
float maxExplodeTime = 0.8;    // max explodetime in seconds
ArrayList<PhysOb> object;      // array of objects

void setup() {
  size(800, 600);
  object = new ArrayList<PhysOb>();
  object.add(new PhysOb(new PVector(width/2, height/2), false));
  ms = millis();
}

void draw() {
  background(10, 10, 50);
  t = millis()-ms; // resets the timer
  t = t/1000;      // to seonds
  ms = millis();   // stores the ms to next draw
  spawnTimer += t*objPerSeconds;

  if(spawnTimer >= 1){  // launch each objPerSeconds
    object.add(new PhysOb(new PVector(width/2, height/2), false)); // creates new object
    spawnTimer = 0;
  }

  // loops through all objects on screen
  for(int i = 0; i<object.size(); i++){
    PhysOb obj = object.get(i);
    obj.update();  // movement
    obj.display(); // draw the object

    // if the ball should explode and it is time for it to explode
    if(obj.willExplode ==  true && obj.timeToExplode()){
      obj.willExplode = false;  // so it doesn't explode again
      for(int a=0; a<ballsFromExplosion; a++){
        object.add(new PhysOb(new PVector(obj.pos.x, obj.pos.y), true));
      }
    }

    // removes the object from the arraylist if its position is under the ground
    if(obj.checkHeight()){
      object.remove(i);
    }
```

```
  }

}

class PhysOb  {
  PVector pos;
  PVector vel;
  PVector acc;
  float timeBorn;
  float explosionTime;
  boolean willExplode;

  // constructor
  PhysOb (PVector position, boolean createdFromExplosion) {
    float radians, degree;
    acc = new PVector(wind, acceleration);
    if(!createdFromExplosion){         // if its not created from explosion
      degree = random(-135, -45);      // random degree
      willExplode = randomExplode();
      explosionTime = random(0, maxExplodeTime);
      print("\n" + explosionTime);
    } else {                           // if its created from explosion
      degree = random(-180, 180);      // random degree
      willExplode = false;
    }
    radians = degree * PI/180;         // degree to radians
    vel = new PVector(cos(radians), sin(radians));
    vel.mult(velWeight);
    pos = position;
    timeBorn = millis();
  }

  void update() {
    vel.add(acc.mult(1+t));
    pos.add(vel.mult(1+t));
  }

  void display() {
    // color goes from orange to black based on pos.y
    fill( ((height-pos.y)/height/2)*4*255, ((height-pos.y)/height/2)*2*255, 0);
    ellipse(pos.x, pos.y, r*2, r*2);
  }

  boolean checkHeight() {
    if (pos.y > height+r) {
      return true;
```

```
    } else {
      return false;
    }
  }

  boolean randomExplode(){
    if(random(0, 100) <= explodeChance){
      return true;
    } else {
      return false;
    }
  }

  boolean timeToExplode(){
    if(millis()/1000-timeBorn/1000 > explosionTime){
        return true;
    } else {
      return false;
    }
  }
}


void keyPressed() {
  save("myimage.png");
}
```