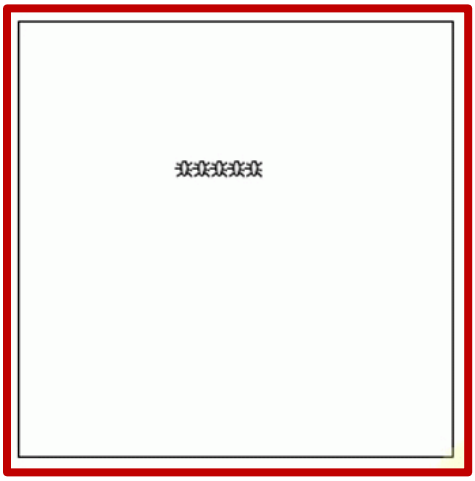
SPĒLE «ČŪSKA» II
32.nodarbība

IEPRIEKŠĒJĀ NODARBĪBĀ

- Iepriekšējā nodarbībā – čūska, kura pārvietojas dotajā virzienā un zaudē dzīvību, ieskrienot sienā.
- Mums jau ir:
  - čūskas ķermenis, kas seko galvai.

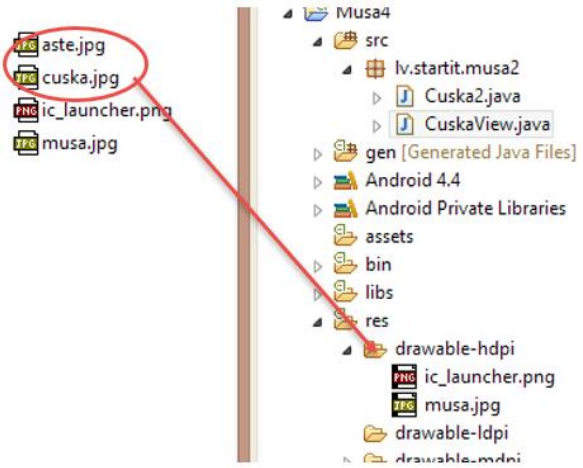


ŠAJĀ NODARBĪBĀ

- Pabeigsim čūsku.
- Darbu saraksts:
  - pievienot galvas, astes un ābola attēlus,
  - pabarot čūsku,
  - pārbaudīt, lai čūska neēd pati sevi.

PIEVIENO GALVU, ASTI UN ĀBOLU

- Uzzīmē galvu, asti, ābolu.
- Attēlu izmēri tādi paši, kā mušai (kas tagad būs čūskas ķermenis) – 10x10.
- Pievieno attēlus drawable-hdpi mapē.



SAGATAVO GALVU, ASTI UN ĀBOLU

```
public class CuskaView extends View {
  Paint krasa;
  Bitmap musaJpg;
  Bitmap galva;
  Bitmap aste;
  Bitmap abols;
  boolean prepared = false;
. . .
public void prepareForDrawing(){
  . . .
  musaJpg =  BitmapFactory.decodeResource(getResources(), R.drawable.musa);
  galva = BitmapFactory.decodeResource(getResources(), R.drawable.cuska);
  aste = BitmapFactory.decodeResource(getResources(), R.drawable.aste);
  abols = BitmapFactory.decodeResource(getResources(), R.drawable.abols);
. . .
}
. . .
}
```

ZĪMĒ GALVU UN ASTI

```
public void onDraw(Canvas canvas){
  if (prepared){
    int len = snakesXy.length;
    // zimet robezu
    canvas.drawRect(0, 0, MAX_X*10 - 1 , MAX_Y*10 - 1, krasa);
    //zimet musu
    for( int i=0; i<snakesXy.length; i++ ) {
    // zīmē ķermeni
```

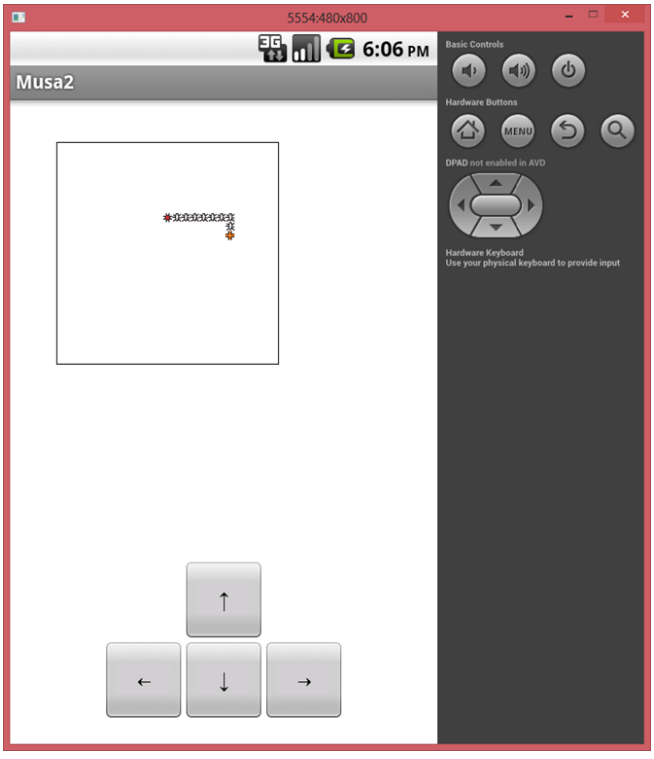
```
    for( int i=1; i<len-1; i++ ) {
      canvas.drawBitmap(musaJpg, snakesXy[i][0]*10, snakesXy[i][1]*10, null);
    }
    // zīmē galvu
    canvas.drawBitmap(galva, snakesXy[0][0]*10, snakesXy[0][1]*10, null);
    // zīmē asti
    canvas.drawBitmap(aste, snakesXy[len-1][0]*10, snakesXy[len-1][1]*10,
null);
  }
}
```

snakeXy

	0	1
0	5	7
1	5	6
2	5	5
3	5	4
4	5	3

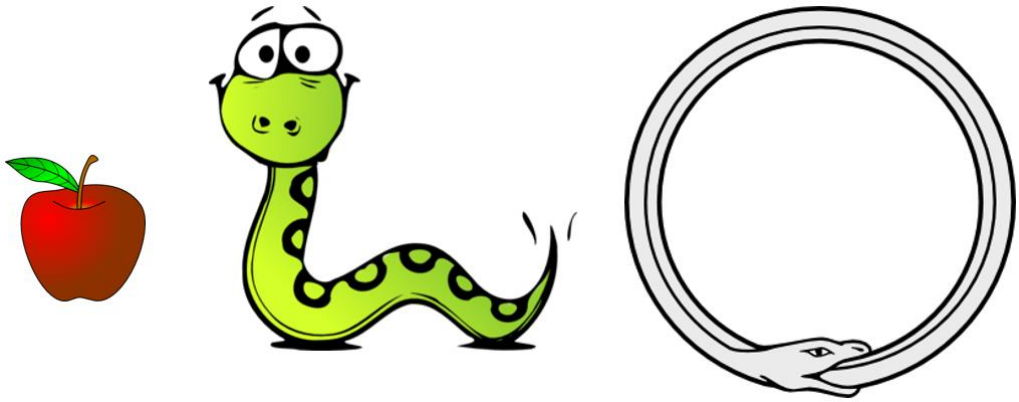
```
len = 5
snakeXy[len-1][1] = 3
```

TESTĒJAM



KO TĀLĀK?

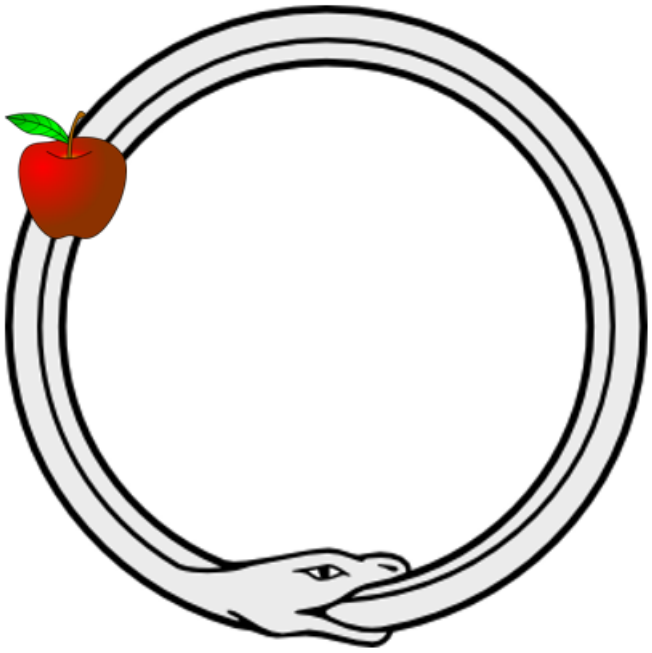
- Čūska ir jāpabaro.
- Ir jāpārbauda, vai čūska neēd pati sevi.



KAS BŪS NEPIECIEŠAMS?

Abos gadījumos nepieciešama metode, kas pārbauda, vai punkts ir uz čūskas:

- lai čūska neapēd sevi,
- lai ābols netiek uzģenerēts uz čūskas ķermeņa.



METODE isPointOnSnake()

```
// atgriež true, ja dotais punkts atrodas uz čūskas ķermeņa
// ja galva==1, tad pārbaudē neiekļauj galvu (sevis ēšanai)
// ja galva==0, tad pārbauda arī galvu (ābola novietošanai)
boolean isPointOnSnake( int px, int py, int galva ) {
  for( int i=galva; i<snakesXy.length; i++ ) {
    // ja punkts uz čūskas, tad atgriež true
    if( snakesXy[i][0] == px && snakesXy[i][1] == py ) {
      return true;
    }
  }
  // ja punkts nebija uz čūskas, tad atgriež false
  return false;
}
```

snakeXy

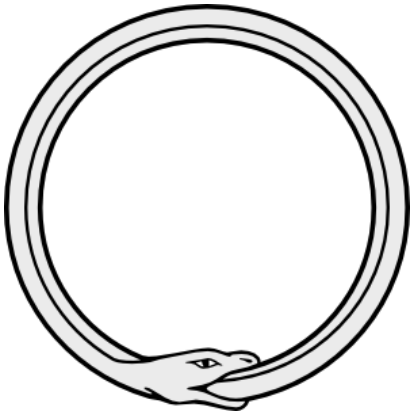
	0	1
0	5	7
1	5	6
2	5	5
3	5	4
4	5	3

abolsX = 5
abolsY = 4

UZLABOTĀ METODE ValidPosition()

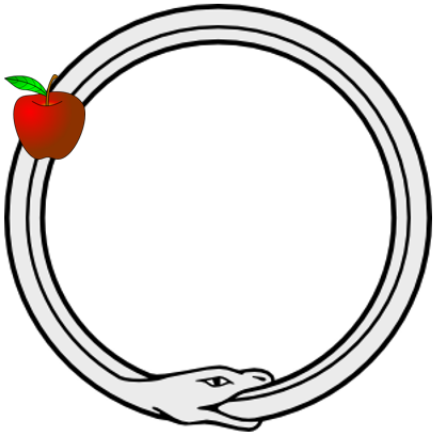
```
public boolean validPosition() {
  if (0 <= x && x < MAX_X && 0 <= y && y < MAX_Y) {
    // ja mēs nepieskārāmies malām, tad
    // pārbaudam, vai neēdam paši sevi
    if( isPointOnSnake( x, y, 1 ) ) {
      // ja ēd sevi, tad mirst
      return false;
    } else {
```

```
      // ja neēd sevi un nav ārpus laukuma,
      // tad dzīvojam!
      return true;
    }
  } else {
    return false;
  }
}
```



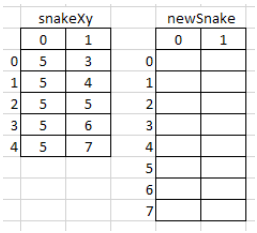
ĀBOLA ĢENERĒŠANA

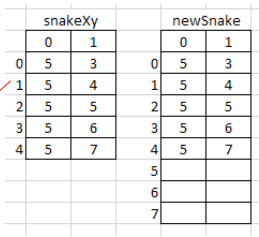
```
public class CuskaView extends View {
. . .
  int  appleX=1, appleY=1;
. . .
  void newApple() {
    do {
      appleX = r.nextInt(MAX_X);
      appleY = r.nextInt(MAX_Y);

      // turpināt ģenerēt ābolu, kamēr tas ir uz čūskas ķermeņa
    } while( isPointOnSnake(appleX, appleY, 0) == true );
  }
}
public void prepareForDrawing(){
. . .
  // pēc cikla, kas aizpilda čūskas ķermeni!
  newApple();
  prepared = true;
}
```

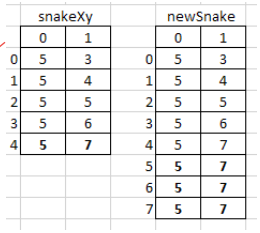


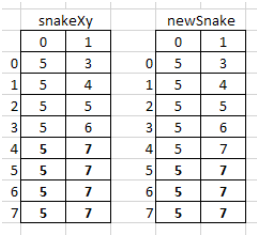
UZLABOTĀ METODE move()

```
public void move() {
  x = x + dX;
  y = y + dY;
  for( int i=snakesXy.length-1; i>0; i-- ) {
    snakesXy[i][0] = snakesXy[i-1][0];
    snakesXy[i][1] = snakesXy[i-1][1];
  }
  snakesXy[0][0] = x;
  snakesXy[0][1] = y;
  if( x == appleX && y == appleY ) {
    newApple(); // jauns ābols
    int    len = snakesXy.length;
    int    i;
    int[][] newSnake = new int[len+3][2]; // jauns masīvs
    for( i=0; i<len; i++ ) { // nokopē čūsku uz jauno masīvu
      newSnake[i][0] = snakesXy[i][0];
      newSnake[i][1] = snakesXy[i][1];
    }
    for( i=len; i<len+3; i++ ) { // masīvu papildina ar čūskas
                astes koordināti
      newSnake[i][0] = snakesXy[len-1][0];
      newSnake[i][1] = snakesXy[len-1][1];
    }
    snakesXy = newSnake; // saglabā jauno masīvu kā mūsu čūsku
  }
}
```

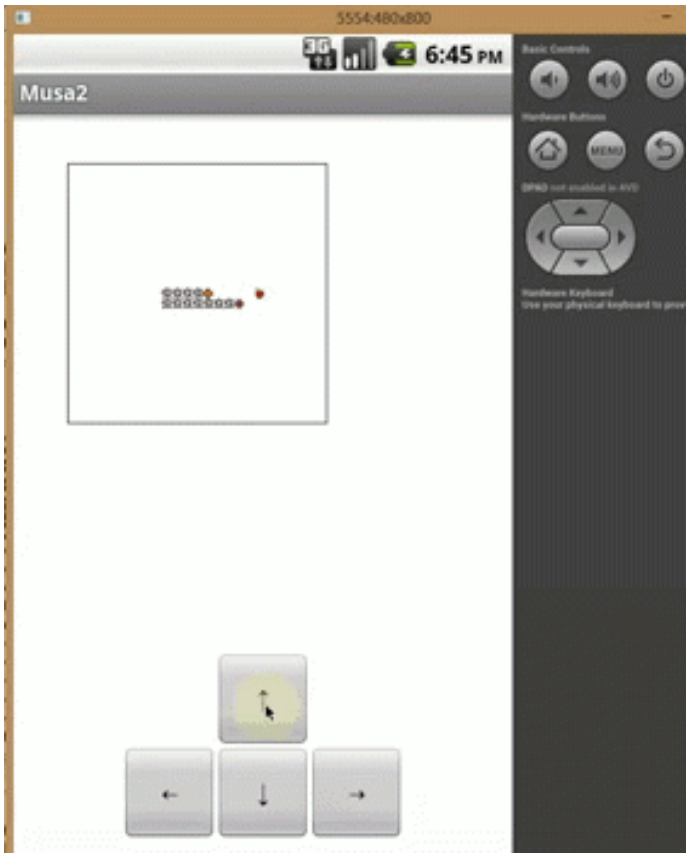








TESTĒJAM



ATTĒLI NO

- http://www.clker.com/clipart-29197.html
- http://www.clker.com/clipart-snake-no-white-drule.html
- http://www.clker.com/clipart-3982.html