SPĒLE «ČŪSKA» II
32.nodarbība

1. UZDEVUMS

1. Kura čūska apēda pati sevi?

a)

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

b)

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

c)

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

d)

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

2. UZDEVUMS

1. Kurš cikls pareizi pārkopēs esošo čūsku no masīva snakesXy uz jaunu masīvu newSnake?

A.
```
for( i=0; i<snakesXy.length; i++ ) {
  snakesXy[i][0] = newSnake[i][0];
  snakesXy[i][1] = newSnake[i][1];
}
```
B.
```
for( i=0; i<snakesXy.length; i++ ) {
  newSnake[i][0] = snakesXy[i][0];
  newSnake[i][1] = snakesXy[i][1];
}
```
C.
```
for( i=0; i<newSnake.length; i++ ) {
  snakesXy[i][0] = newSnake[i][0];
  snakesXy[i][1] = newSnake[i][1];
}
```
D.
```
for( i=0; i<newSnake.length; i++ ) {
  newSnake[i][0] = snakesXy[i][0];
  newSnake[i][1] = snakesXy[i][1];
}
```

UZDEVUMI PRAKTISKAJAM DARBAM

- Pievienot pogu Starts, kura jāspiež:
  - lai sāktu spēli pirmo reizi,
  - lai atsāktu spēli, kad čūska ir zaudējusi dzīvību.
- Izveidot vairākus ābolus.
- Ar laiku paātrināt čūskas pārvietošanās ātrumu.

ATRISINĀJUMI PRAKTISKAJAM DARBAM – POGA STARTS

```
// === Aktivitātes klasē ===
protected void onCreate(Bundle savedInstanceState) {
  ...
  //stradnieks.postDelayed(uzdevums, 1000); // nokomentē, lai spēle nesākas tikko parādās uz ekrāna
  ...
}
// pogas start metode
public void start(View v) {
  // sagatavo jaunu čūsku
  musa.startNew();
  // Svarīgi! Izdzēš no rindas jau strādājošos strādniekus,
  //   lai spaidot pogu Starts ātri pēc kārtas, rindā nerastos
  //   lieki, vēl neapstrādāti strādnieki.
  stradnieks.removeCallbacks(uzdevums);
  // palaiž jaunu strādnieku
  stradnieks.postDelayed(uzdevums, 500);
}
// === vizuālās komponentes klasē ===
public void prepareForDrawing(){
  ...
  //prepared = true; // nokomentē, lai neko nezīmē pirms nav izpildīts startNew()
  ...
}
public void startNew() { // jauna metode, kas inicializē čūsku un ģenerē ābolu
```

```
  x = 12; // čūskas sākuma koordināte
  y = 12;
  dX = 0; // čūskas sākuma virziens
  dY = -1;
  snakesXy = new int[10][2]; // čūskas sākuma garums
  for( int i=0; i<snakesXy.length; i++ ) { // aizpilda čūskas ķermeni
    snakesXy[i][0] = x;
    snakesXy[i][1] = y+i;
  }
  newApple(); // pēc cikla, kas aizpilda čūskas ķermeni!
  prepared = true;
}
```

ATRISINĀJUMI PRAKTISKAJAM DARBAM – VAIRĀKI ĀBOLI

// viens ābols

```
// ābola koordināte mainīgajos
// nav ciklu
int appleX=1, appleY=1;
canvas.drawBitmap(abols, appleX*10,
appleY*10,
                    null);
newApple();
if( x == appleX && y == appleY ) {
  newApple();
  . . .
}
void newApple() {
  do {
    appleX = r.nextInt(MAX_X);
    appleY = r.nextInt(MAX_Y);
  } while( isPointOnSnake(appleX,
           appleY, 0) == true );
}
```

// daudzi āboli

```
// koordinātes ābolu masīvā
// visur lieto ciklus
int[]appleX = new int[3]; // cik ābolus vēlies?
int[]appleY = new int[3];
for( int a=0; a<appleX.length; a++ ) {
  canvas.drawBitmap(abols, appleX[a]*10,
                    appleY[a]*10, null);
}
for( int a=0; a<appleX.length; a++ ) {
  newApple( a );
}
for( int a=0; a<appleX.length; a++ ) {
  if( x == appleX[a] && y == appleY[a] ) {
    newApple( a );
    . . .
  }
}
void newApple( int n ) {
  do {
    appleX[n] = r.nextInt(MAX_X);
    appleY[n] = r.nextInt(MAX_Y);
  } while( isPointOnSnake(appleX[n],
           appleY[n], 0) == true );
```

ATRISINĀJUMI PRAKTISKAJAM DARBAM – PAĀTRINĀŠANA

```
public class Cuska2 extends Activity {
. . .
int atrums; // ātrums
. . .
public void start(View v){
atrums = 333; // kad piespiež «Starts», tad izvēlas sākuma ātrumu
. . .
}
. . .
public void update(){
. . .
stradnieks.removeCallbacks(uzdevums); // izmet no rindas iepriekšējos
callback (ja nu uzkrājušies)
stradnieks.postDelayed(uzdevums, atrums); // nākamā kustība, izmantojot
mainīgo ātrums
atrums = atrums - 1; // katru reizi no ātruma atņemam 1 milisekundi
if( atrums < 50 ) { // pārbauda, lai ātrums nekļūst pārāk liels
atrums = 50;
}
. . .
}
. . .
}
```