



Lecția 11 – Python Games

Trainer: Hrișcă Miruna

Aplicații folosite + alternative ▾

01

PyCharm

[Link - windows](#)

[Link - Apple](#)

02

One Compiler

[Link](#)

03

VS Code

[Link](#)

04

Online GDB

[Link](#)

Biblioteca Turtle

1. `onscreenclick(funcție)`

- Această funcție spune calculatorului:

„Când cineva face click pe ecran, apelează această funcție!”

- Este ca și cum ai spune: „Dacă apeși pe ecran, broasca țestoasă va reacționa.”

2. `hideturtle()`

- Face broasca țestoasă **invizibilă**, astfel încât **vedem doar desenul**, nu și broasca.
- Imaginează-ți că broasca țestoasă este un **pix invizibil** care desenează fără să o vedem.

3. `tracer(False)`

- Această funcție **oprește animația** desenului, ca să nu mai vedem broasca țestoasă mișcându-se pas cu pas.
- Astfel, desenul apare **instantaneu**, mult mai repede.

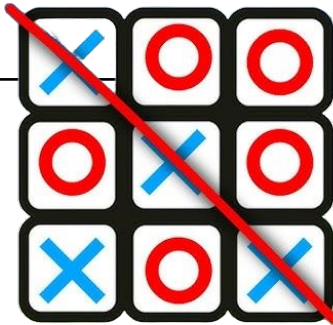


Biblioteca FreeGames



1. `line(start, end)` – desenează o linie între două puncte.
 - `start` și `end` sunt puncte (de obicei vectori cu coordonate x și y).





Despre jocul X și O

Ce este jocul X și O

- Este un joc clasic pentru **doi jucători**.
- Tabla este formată dintr-o **grilă 3x3** (9 pătrate).
- Fiecare jucător are un simbol: **X** sau **O**.
- Scopul jocului: să conectezi **trei simboluri identice pe rând**, fie pe orizontală, verticală sau diagonală.

Cum jucăm

1. **Apăsăm pe ecran** pentru a plasa simbolul nostru în pătratul dorit.
2. Simbolul se plasează doar dacă pătratul e liber.
3. După fiecare mutare, jocul verifică dacă cineva a câștigat sau dacă tabla e plină (remiză).

Cum câștigi

- Câștigi dacă reușești să conectezi **trei simboluri identice pe rând**:
 - **Orizontal**: pe aceeași linie.
 - **Vertical**: pe aceeași coloană.
 - **Diagonal**: de la un colț la celălalt.



Programarea jocului

```
from turtle import *  
from freegames import line
```



- ❖ `turtle` – folosit pentru a desena grafic pe ecran.
- ❖ `line` din `freegames` – funcție care desenează o linie între două puncte $(x1, y1)$ și $(x2, y2)$.



Programarea jocului

```
def grid():  
    """Draw tic-tac-toe grid."""  
    line(-67, 200, -67, -200)  
    line(67, 200, 67, -200)  
    line(-200, -67, 200, -67)  
    line(-200, 67, 200, 67)
```

- ❖ Desenează cele **4 linii care formează grila 3x3** pentru X și O:
 - Două verticale: la $x = -67$ și $x = 67$.
 - Două orizontale: la $y = -67$ și $y = 67$.
- ❖ Această grilă împarte ecranul în 9 pătrățele egale.

Programarea jocului

```
def drawx(x, y):  
    """Draw X player."""  
    line(x, y, x + 133, y + 133)  
    line(x, y + 133, x + 133, y)
```

- ❖ Desenează simbolul **X** în pătratul ales.
- ❖ Folosește două linii care se intersectează în diagonală:
 1. De la colțul stânga-jos la colțul dreapta-sus.
 2. De la colțul stânga-sus la colțul dreapta-jos.

Programarea jocului



```
def drawo(x, y):  
    """Draw O player."""  
    up()  
    goto(x + 67, y + 5)  
    down()  
    circle(62)
```

- ❖ Desenează simbolul **O** în pătratul ales.
- ❖ `up()` și `goto()` mută cursorul fără să deseneze.
- ❖ `down()` începe desenarea.
- ❖ `circle(62)` desenează un cerc cu raza 62.



Programarea jocului

```
def floor(value):  
    """Round value down to grid with square size 133."""  
    return ((value + 200) // 133) * 133 - 200
```



- ❖ **value + 200**
 - Ecranul are coordonate de la **-200 până la 200** (pe ambele axe).
 - Adăugând 200, mutăm intervalul la **0 până la 400**, ca să fie mai ușor de lucrat cu împărțirea.
- ❖ **// 133**
 - **//** este **împărțirea întreagă** (rezultatul e rotunjit în jos la cel mai apropiat număr întreg).
 - 133 = dimensiunea unui pătrat.
 - Această parte calculează **în ce pătrat se află coordonata**.
- ❖ *** 133**
 - Înmulțim rezultatul împărțirii cu 133 pentru a obține **punctul de început al pătratului**.
- ❖ **- 200**
 - Scădem 200 pentru a **reveni la sistemul de coordonate original**, care merge de la -200 la 200.

Programarea jocului




```
state = {'player': 0}
players = [drawx, drawo]
```



- ❖ `state['player']` – păstrează jucătorul curent:
 - `0` → X
 - `1` → O
- ❖ `players` – listă cu funcțiile pentru desenarea X și O.



Programarea jocului



```
def tap(x, y):  
    """Draw X or O in tapped  
    square."""  
    x = floor(x)  
    y = floor(y)  
    player = state['player']  
    draw = players[player]  
    draw(x, y)  
    update()  
    state['player'] = not  
    player
```

- ❖ `floor(x)` și `floor(y)` – ajustează coordonatele la pătratul corect.
- ❖ `player = state['player']` – află cine e rândul.
- ❖ `draw = players[player]` – alege funcția `drawx` sau `drawo`.
- ❖ `draw(x, y)` – desenează simbolul în pătrat.
- ❖ `update()` – actualizează ecranul.
- ❖ `state['player'] = not player` – schimbă rândul la următorul jucător.

Programarea jocului



```
setup(420, 420, 370, 0)
hideturtle()
tracer(False)
grid()
update()
onscreenclick(tap)
done()
```



- ❖ `setup(420, 420, 370, 0)` – dimensiunea ferestrei.
- ❖ `hideturtle()` – ascunde cursorul.
- ❖ `tracer(False)` – controlează animația manual.
- ❖ `grid()` – desenează tabla de joc.
- ❖ `update()` – afișează totul.
- ❖ `onscreenclick(tap)` – apelează funcția `tap` la apăsarea mouse-ului.
- ❖ `done()` – păstrează fereastra deschisă până când o închidem.