# Centipede Game

In [game theory](https://en.wikipedia.org/wiki/Game_theory), the **centipede game**, first introduced by [Robert Rosenthal](https://en.wikipedia.org/wiki/Robert_W._Rosenthal) in 1981, is an [extensive form game](https://en.wikipedia.org/wiki/Extensive_form_game) in which two players take turns choosing either to take a slightly larger share of an increasing pot, or to pass the pot to the other player. The payoffs are arranged so that if one passes the pot to one's opponent and the opponent takes the pot on the next round, one receives slightly less than if one had taken the pot on this round. Although the traditional centipede game had a limit of 100 rounds (hence the name), any game with this structure but a different number of rounds is called a centipede game.

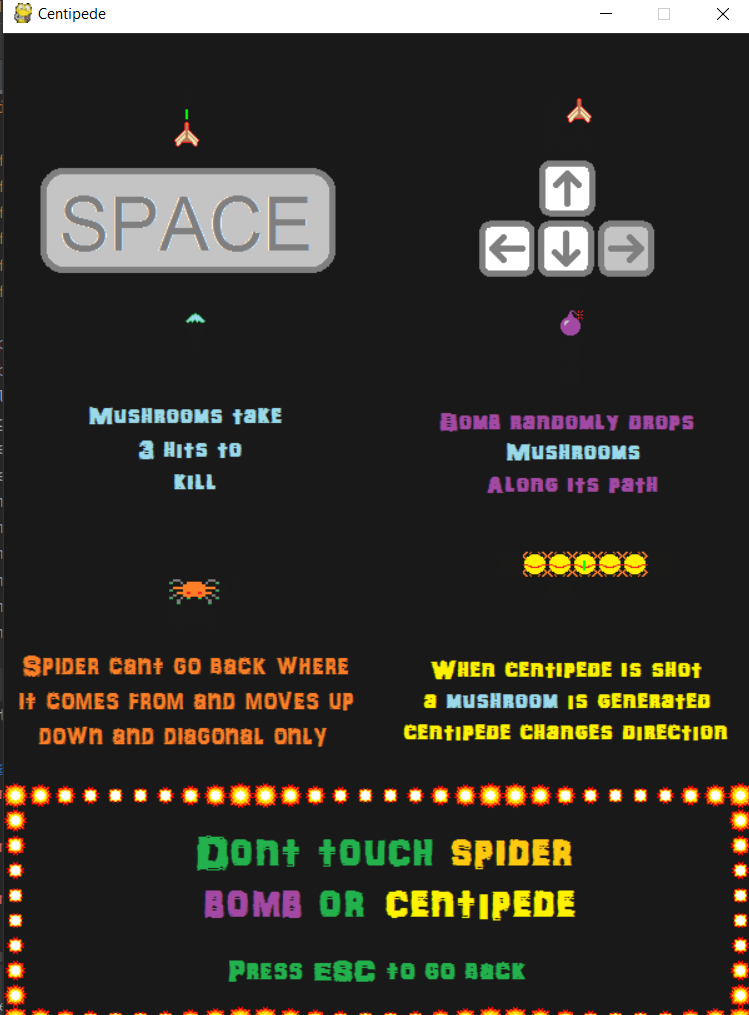
The unique [subgame perfect equilibrium](https://en.wikipedia.org/wiki/Subgame_perfect_equilibrium) (and every [Nash equilibrium](https://en.wikipedia.org/wiki/Nash_equilibrium)) of these games indicates that the first player take the pot on the very first round of the game; however, in [empirical](https://en.wikipedia.org/wiki/Empirical) tests, relatively few players do so, and as a result, achieve a higher payoff than the payoff predicted by the equilibria analysis. These results are taken to show that subgame perfect equilibria and Nash equilibria fail to predict human play in some circumstances. The Centipede game is commonly used in introductory game theory courses and texts to highlight the concept of [backward induction](https://en.wikipedia.org/wiki/Backward_induction) and the [iterated elimination of dominated strategies](https://en.wikipedia.org/wiki/Dominance_(game_theory)), which show a standard way of providing a solution to the game.

Centipede also balances its monsters to become harder and harder as players scores increase. And since the score can never decrease, the tension escalates over the course of the game. Most obvious is the spider, whose speed approximately doubles once the score reaches 5000 (1000 if the game's operator has set the game to "hard"). The spider also maneuvers in a smaller and smaller area of the bottom of the screen as the score gets really high, eventually moving only one row out of the player's six-row area. With the spider thus constrained, it is both more likely to hit players and less ikely for players to be able to shoot it. Recall that the flea drops from the top of the screen based on the quantity of mushrooms in the bottom half of the screen. When players start the game, if there are less than five mushrooms in that area, the flea will come down, dropping more as it does so. As the score increases, however, so does the quantity of mushrooms needed to prevent the flea's appearance. Now players must leave more and more mushrooms in that space to prevent the flea from coming out and cluttering the top of the screen with mushrooms.

At the start of each wave, the game always generates a total of twelve centipede segments and heads at the top of the screen. This means that ifa twelve-segment centipede appears at the top of the screen, it will be the only centipede. If a seven-segment centipede appears, then five other centipede heads will appear as well, thus totaling the magic number of twelve. The more centipedes that appear, the more hallenging it is for players to shoot them all, and the more likely one will sneak to the bottom of thesCreen. The game starts by releasing a single twelve-segment centipede. In the next wave, a slow eleven segment centipede appears along with one head. In the following wave, a fast eleven-segment and one head combination arrive. Then a slow ten-segment and two heads appear. With each wave there are a greater number of individual centipedes for players to keep track of and a greater escalation ot tension. The game cycles around once twelve individual heads are spawned, and then becomes harder by only spawning fast centipedes. The player's death also provides a brief respite from the tension. When the player's ship is destroyed, the waye starts over and hence the centipede returns to the top of the screen. Before this, however, all of the mushrooms on the SCreen are reset. This means that all the partially destroyed mushrooms are returned to their undamaged state and all of the mushrOoms poisoned by the scorpion are returned to their unpoisoned state. Many waves into the game, the increased mushroom density makes shooting poisoned mushrooms all but impossible, and with those p0isoned mushrooms in place, players are bombarded by centipedes hurtling toward them in every single wave. Thus, players are almost relieved when their shooter is destroyed and all those poisoned mushrooms are removed from the top of the screen. This causes the game to be much more relaxed, at least for a time.

When the game runs, its contents are as follows;





**My code;**

import pygame,random,time,pygame.event  
  
from Player import \*  
from Fire import \*  
from Spider import \*  
from Bomb import \*  
from LilCenti import \*  
from Expo import \*  
  
pygame.init()  
bg =(25,25,25)  
level=1  
game\_map=[]  
empty=pygame.Surface([20,20])  
empty.fill(bg)  
mushroom\_image = pygame.image.load('shroom1.png')  
mushroom\_image2 = pygame.image.load('shroom2.png')  
mushroom\_image3 = pygame.image.load('shroom3.png')  
mushroom\_image.set\_colorkey((0,0,0))  
mushroom\_image2.set\_colorkey((0,0,0))  
mushroom\_image3.set\_colorkey((0,0,0))  
  
def setup\_game\_map():  
 global game\_map  
 game\_map = []  
 for x in range(40):  
 arrayOfZeros = [0]\*30  
 game\_map.append(arrayOfZeros)  
 for x in range (30):  
 randomX=random.randint(0,29)  
 randomY=random.randint(0,27)  
 game\_map[randomX][randomY] = 1  
  
def draw\_game\_map():  
 for column in range(30):  
 for row in range(40):  
 spot = game\_map[row][column]  
 if spot == 1:  
 screen.blit(empty,[column\*20, row\*20])  
 screen.blit(mushroom\_image, [column\*20, row\*20])  
 if spot == 2:  
 screen.blit(empty,[column\*20, row\*20])  
 screen.blit(mushroom\_image2, [column\*20, row\*20])  
 if spot == 3:  
 screen.blit(empty,[column\*20, row\*20])  
 screen.blit(mushroom\_image3, [column\*20, row\*20])  
 if spot == 4:  
 screen.blit(empty,[column\*20, row\*20])  
 game\_map[row][column] = 0  
  
def get\_key():  
 while 1:  
 event = pygame.event.poll()  
 if event.type == pygame.KEYDOWN:  
 return event.key  
 else:  
 pass  
size=[600,800]  
screen=pygame.display.set\_mode(size)  
pygame.display.set\_caption("Centipede")  
  
player=Player(300,700)  
fire=Fire()  
bomb=Bomb()  
fireGroup=pygame.sprite.Group()  
fireGroup.add(fire)  
clock=pygame.time.Clock()  
going=True  
  
background=pygame.Surface(size)  
background.fill(bg)  
screen.blit(background,(0,0))  
  
allsprites=pygame.sprite.Group()  
allsprites.add(player)  
allsprites.add(fireGroup)  
allsprites.add(bomb)  
  
expos=pygame.sprite.Group()  
  
centis=pygame.sprite.Group()  
for m in range(12):  
 centi=LilCenti(20\*m,-20)  
 centis.add(centi)  
  
spider=Spider()  
allsprites.add(spider)  
allsprites.add(centis)  
allsprites.add(expos)  
setup\_game\_map()  
clock\_tick=20  
game\_mode='menu'  
tickCounter=0  
  
gameOverFont = pygame.font.Font('ARDARLING.ttf' ,70)  
clickToStart = pygame.font.Font('ARDARLING.ttf',40)  
highScore = pygame.font.Font('ARDARLING.ttf',50)  
  
menu\_header=[]  
menu\_header.append(pygame.image.load("menu\_centi1.png"))  
menu\_header.append(pygame.image.load("menu\_centi2.png"))  
menu\_high=[]  
menu\_high.append(pygame.image.load("menu\_high1.png"))  
menu\_high.append(pygame.image.load("menu\_high2.png"))  
menu\_ins=[]  
menu\_ins.append(pygame.image.load("menu\_ins1.png"))  
menu\_ins.append(pygame.image.load("menu\_ins2.png"))  
menu\_play=[]  
menu\_play.append(pygame.image.load("menu\_play1.png"))  
menu\_play.append(pygame.image.load("menu\_play2.png"))  
menu\_quit=[]  
menu\_quit.append(pygame.image.load("menu\_quit1.png"))  
menu\_quit.append(pygame.image.load("menu\_quit2.png"))  
menu\_footer=pygame.image.load("menu\_footer.png")  
  
inst\_space=[]  
for i in range(1,5):  
 inst\_space.append(pygame.image.load("instructions%d.png" % i))  
inst\_up=[]  
for i in range(1,5):  
 inst\_up.append(pygame.image.load("instructions\_up%d.png" % i))  
inst\_shroom=[]  
for i in range(1,7):  
 inst\_shroom.append(pygame.image.load("instructions\_shroom%d.png" % i))  
inst\_bomb=[]  
for i in range(1,5):  
 inst\_bomb.append(pygame.image.load("instructions\_bomb%d.png" % i))  
inst\_spider=[]  
for i in range(1,5):  
 inst\_spider.append(pygame.image.load("instructions\_sp%d.png" % i))  
inst\_centi=[]  
for i in range(1,5):  
 inst\_centi.append(pygame.image.load("instructions\_centi%d.png" % i))  
inst\_footer=pygame.image.load("instructions\_footerAAA.png")  
  
  
high\_footer=pygame.image.load("high\_footer.png")  
  
playerNames=['AAA','AAA','AAA','AAA','AAA','AAA','AAA','AAA','AAA']  
playerScores=[999,888,777,666,555,444,333,22,1]  
  
currentUser=['A','A','A']  
currentCharacter=0  
currentScore=0  
lastScore=0  
  
menu\_selection=1  
slowDownAnimation=0  
while going:   
 clock.tick(clock\_tick)  
 tickCounter+=1  
 for event in pygame.event.get():  
 if event.type==pygame.QUIT:  
 going=False  
 if game\_mode=='savescore':  
 lastScore=currentScore  
 currentCharacter=0  
 userText=''  
 for i in range(len(currentUser)):  
 userText+=currentUser[i]  
 text = gameOverFont.render(userText, True, (255,255,255))  
 text\_rect = text.get\_rect()  
 text\_x = screen.get\_width() / 2 - text\_rect.width / 2  
 screen.blit(text, [text\_x, text\_y+300])  
 pygame.display.flip()  
 while currentCharacter<3:  
 inkey=get\_key()  
 if inkey == pygame.K\_RETURN:  
 game\_mode='menu'  
 break  
 userText=''  
 if inkey == pygame.K\_BACKSPACE:  
 currentUser=currentUser[0:-1]  
 elif inkey <= 127:  
 currentUser[currentCharacter]=chr(inkey-32)  
 currentCharacter+=1  
   
 for i in range(len(currentUser)):  
 userText+=currentUser[i]  
   
 text = gameOverFont.render(userText, True, (255,255,255))  
 text\_rect = text.get\_rect()  
 text\_x = screen.get\_width() / 2 - text\_rect.width / 2  
 refresh=pygame.Surface([text\_rect.width,text\_rect.height])  
 refresh.fill(bg)  
 screen.blit(refresh,[text\_x, text\_y+300])  
 screen.blit(text, [text\_x, text\_y+300])  
 pygame.display.flip()  
  
 print('name done')  
 print(lastScore)  
 print(userText)  
 for i in range(9):  
 if lastScore>=playerScores[i]:  
 playerScores.insert(i,lastScore)  
 playerNames.insert(i,userText)  
 print(userText)  
 break  
 game\_mode='menu'   
 if game\_mode=='high':  
 pygame.display.set\_caption("Centipede")  
 title=gameOverFont.render('High Scores',True,(255,255,255))  
 screen.blit(high\_footer,(0,719))  
  
 title\_rect = title.get\_rect()  
 title\_x = screen.get\_width() / 2 - title\_rect.width / 2  
 title\_y = 40  
 screen.blit(title, [title\_x, title\_y])  
 for i in range(9):  
 name = highScore.render(str(i+1)+'. '+playerNames[i],True,(255,255,255))  
 text\_rect = name.get\_rect()  
 name\_x = screen.get\_width() / 4 - text\_rect.width / 2  
 name\_y = 150 + 60\*(i)  
 screen.blit(name,[name\_x,name\_y])  
  
 for i in range(9):  
 name = highScore.render(str(playerScores[i]),True,(255,255,255))  
 text\_rect = name.get\_rect()  
 name\_x = 3\*(screen.get\_width() / 4) - text\_rect.width / 2  
 name\_y = 150 + 60\*(i)  
 screen.blit(name,[name\_x,name\_y])  
 keys=pygame.key.get\_pressed()  
 if(keys[pygame.K\_ESCAPE]):  
 game\_mode='menu'  
 refresh=pygame.Surface([600,800])  
 refresh.fill(bg)  
 screen.blit(refresh,[0,0])  
 menu\_selection=2  
   
   
 if game\_mode=='inst':  
 pygame.display.set\_caption("Centipede")  
 if(tickCounter%10==0):  
 slowDownAnimation+=1  
 screen.blit(inst\_footer,(0,600))  
 screen.blit(inst\_space[slowDownAnimation%4],(0,0))  
 screen.blit(inst\_up[slowDownAnimation%4],(300,0))  
 screen.blit(inst\_shroom[slowDownAnimation%6],(0,200))  
 screen.blit(inst\_bomb[slowDownAnimation%4],(300,200))  
 screen.blit(inst\_spider[slowDownAnimation%4],(0,400))  
 screen.blit(inst\_centi[slowDownAnimation%4],(300,400))  
 keys=pygame.key.get\_pressed()  
 if(keys[pygame.K\_ESCAPE]):  
 game\_mode='menu'  
 refresh=pygame.Surface([600,800])  
 refresh.fill(bg)  
 screen.blit(refresh,[0,0])  
 menu\_selection=3  
 slowDownAnimation=0  
  
 if game\_mode=='menu':  
 pygame.display.set\_caption("Centipede")  
 keys=pygame.key.get\_pressed()  
 if(keys[pygame.K\_DOWN] and menu\_selection<4):  
 menu\_selection+=1  
  
 if(keys[pygame.K\_UP] and menu\_selection>1):  
 menu\_selection-=1  
   
 screen.blit(menu\_header[tickCounter%2],(0,0))  
 screen.blit(menu\_footer,(0,625))  
 if menu\_selection==1:  
 screen.blit(menu\_play[1],(0,200))  
 else:  
 screen.blit(menu\_play[0],(0,200))  
  
 if menu\_selection==2:  
 screen.blit(menu\_high[1],(0,325))  
 else:  
 screen.blit(menu\_high[0],(0,325))  
  
 if menu\_selection==3:  
 screen.blit(menu\_ins[1],(0,425))  
 else:  
 screen.blit(menu\_ins[0],(0,425))  
  
 if menu\_selection==4:  
 screen.blit(menu\_quit[1],(0,525))  
 else:  
 screen.blit(menu\_quit[0],(0,525))  
  
 if(keys[pygame.K\_RETURN]):  
 refresh=pygame.Surface([600,800])  
 refresh.fill(bg)  
 screen.blit(refresh,[0,0])  
 if menu\_selection==1:  
 game\_mode='play'  
 elif menu\_selection==2:  
 game\_mode='high'  
 elif menu\_selection==3:  
 game\_mode='inst'  
 elif menu\_selection==4:  
 going=False  
 elif game\_mode=='gameover':  
  
 lastScore = currentScore  
   
 text = gameOverFont.render("GAME OVER!", True, (255,255,255))  
 text\_rect = text.get\_rect()  
 text\_x = screen.get\_width() / 2 - text\_rect.width / 2  
 text\_y = screen.get\_height() / 2 - text\_rect.height / 2  
 screen.blit(text, [text\_x, text\_y-200])  
  
 text = clickToStart.render("Hit [S] to Save", True, (255,255,255))  
 text\_rect = text.get\_rect()  
 text\_x = screen.get\_width() / 2 - text\_rect.width / 2  
 screen.blit(text, [text\_x, text\_y+210])  
   
 text = clickToStart.render("Hit Enter to Start Again", True, (255,255,255))  
 text\_rect = text.get\_rect()  
 text\_x = screen.get\_width() / 2 - text\_rect.width / 2  
 screen.blit(text, [text\_x, text\_y+90])  
  
 text = clickToStart.render("Hit Escape for Menu", True, (255,255,255))  
 text\_rect = text.get\_rect()  
 text\_x = screen.get\_width() / 2 - text\_rect.width / 2  
 screen.blit(text, [text\_x, text\_y+150])  
   
 keys=pygame.key.get\_pressed()  
 if(keys[pygame.K\_RETURN]):  
 game\_mode='play'  
 refresh=pygame.Surface([600,800])  
 refresh.fill(bg)  
 screen.blit(refresh,[0,0])  
 centis=pygame.sprite.Group()  
 for m in range(12):  
 centi=LilCenti(20\*m,-20)  
 centis.add(centi)  
 setup\_game\_map()  
 allsprites.add(centis)  
 allsprites=pygame.sprite.Group()  
 allsprites.add(player)  
 allsprites.add(fireGroup)  
 allsprites.add(bomb)  
 allsprites.add(spider)  
 allsprites.add(centis)  
 allsprites.add(expos)  
 spider.deactivate()  
 bomb.deactivate()  
 fire.deactivate()  
 if(keys[pygame.K\_ESCAPE]):  
 game\_mode='menu'  
 refresh=pygame.Surface([600,800])  
 refresh.fill(bg)  
 screen.blit(refresh,[0,0])  
  
 centis=pygame.sprite.Group()  
 for m in range(12):  
 centi=LilCenti(20\*m,-20)  
 centis.add(centi)  
 setup\_game\_map()  
 allsprites.add(centis)  
 allsprites=pygame.sprite.Group()  
 allsprites.add(player)  
 allsprites.add(fireGroup)  
 allsprites.add(bomb)  
 allsprites.add(spider)  
 allsprites.add(centis)  
 allsprites.add(expos)  
 spider.deactivate()  
 bomb.deactivate()  
 fire.deactivate()  
 menu\_selection=1  
  
 if(keys[pygame.K\_s]):  
 game\_mode='savescore'  
  
 centis=pygame.sprite.Group()  
 for m in range(12):  
 centi=LilCenti(20\*m,-20)  
 centis.add(centi)  
 setup\_game\_map()  
 allsprites.add(centis)  
 allsprites=pygame.sprite.Group()  
 allsprites.add(player)  
 allsprites.add(fireGroup)  
 allsprites.add(bomb)  
 allsprites.add(spider)  
 allsprites.add(centis)  
 allsprites.add(expos)  
 spider.deactivate()  
 bomb.deactivate()  
 fire.deactivate()  
 if game\_mode=='play':  
 shootTileX=int(fire.x/20)  
 shootTileY=int(fire.y/20)  
  
 keys=pygame.key.get\_pressed()  
 if(keys[pygame.K\_SPACE] and fire.canFire):  
 fire.activate(player.rect.x+8,player.rect.y+6)  
 for c in centis:  
 if c.left\_right==1 and c.rect.x<580:  
 if game\_map[int(c.rect.y/20)][int(c.rect.x/20)+1]:  
 c.collide()  
 else:  
 if game\_map[int(c.rect.y/20)][int(c.rect.x/20)-1]:  
 c.collide()  
  
 if c.rect.x==fire.rect.x-8 and c.rect.y==fire.rect.y-6:  
 c.kill()  
 game\_map[shootTileY-1][shootTileX]=1  
 currentScore+=10  
 fire.deactivate()  
  
 if game\_map[shootTileY-1][shootTileX]>0:  
 game\_map[shootTileY-1][shootTileX]=game\_map[shootTileY-1][shootTileX]+1  
 currentScore+=3  
 fire.deactivate()  
   
  
 if spider.isActive==0:  
 rnd=random.randint(0,500/level)  
 if rnd==0:  
 spider.activate()  
  
 if pygame.sprite.spritecollide(spider,fireGroup,False):  
 expo=Explode(spider.rect.x,spider.rect.y)  
 allsprites.add(expo)  
 expos.add(expo)  
 spider.deactivate()  
 fire.deactivate()  
 currentScore+=50  
  
 if bomb.isActive==0:  
 rnd=random.randint(0,10/level)  
 if rnd==0:  
 bomb.activate()  
 else:   
 if(bomb.drop):  
 rnd=random.randint(1,5)  
  
 if(rnd==1 and bomb.ay>0 and bomb.isActive):  
 game\_map[int(bomb.ay/20)+1][int(bomb.ax/20)]=1  
 bomb.drop=0  
  
 if pygame.sprite.spritecollide(bomb,fireGroup,False):  
 expo=Explode(bomb.rect.x,bomb.rect.y)  
 allsprites.add(expo)  
 expos.add(expo)  
 bomb.deactivate()  
 fire.deactivate()  
 currentScore+=30  
  
 if pygame.sprite.spritecollide(player,centis,False):  
 expo=Explode(player.rect.x,player.rect.y)  
 allsprites.add(expo)  
 expos.add(expo)  
 game\_mode='gameover'  
 if pygame.sprite.collide\_rect(player,spider):  
 expo=Explode(player.rect.x,player.rect.y)  
 allsprites.add(expo)  
 expos.add(expo)  
 game\_mode='gameover'  
 if pygame.sprite.collide\_rect(player,bomb):  
 expo=Explode(player.rect.x,player.rect.y)  
 allsprites.add(expo)  
 expos.add(expo)  
 game\_mode='gameover'  
  
  
  
 if(tickCounter%3==0):  
 player.update(keys)  
  
 pygame.display.set\_caption("Score : "+str(currentScore))  
   
 allsprites.clear(screen,background)  
 fire.update()  
 spider.update()  
 bomb.update()  
 centis.update()  
 expos.update()  
 draw\_game\_map()  
 allsprites.draw(screen)  
 pygame.display.flip()  
pygame.quit()