ARAKNOID

A MINI-PROJECT REPORT

Submitted by

C SAMINATHAN 180701210

M SARAVANAN 180701219

In partial fulfillment of the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE , CHENNAI

MAY -2019

ANNA UNIVERSITY

CHENNAI

BONAFIDE CERTIFICATE

Certified that this project **“ARAKNOID”**is the bonafide work of **“M SARAVANAN and C SAMINATHAN”** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.P.KUMAR M.E Ph.D.,	**Mr.M.CHITHAMBARATHANU**
HEAD OF THE DEPARTMENT	**ASSISTANT PROFESSOR(SS)**
Dept. of Computer Science and Engg,	Dept. of Computer Science and Engg,
Rajalakshmi Engineering College,Chennai	Rajalakshmi Engineering College,Chennai

Submitted for the **ANNA UNIVERSITY** practical examination Mini-Project work viva voce held on __________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

Arkanoid is one of the oldest block buster games which made the previous generation go crazy. Here we have recreated the modern version of it. In this, the ball initially is in touch with the bar. After the ball is released, it hits the series of rectangular blocks at the top of it and the ball returns. If the ball hits a block, that particular block gets vanished and a part of the block falls down,if the residue of the block hits the bar,the life gets reduced with the exception to the power blocks. Each time when the ball returns to the bar, the player has to move the block (only left and right) and project it to the blocks. If the player fails to move the bar where ball is coming, and leaves the ball without touching, he loses a life. For each player, initially 3 lives are given. After all the blocks are disappeared, the player completes that level. We have 5 levels and at each level, the difficulty of the game gets increased and also the player gets extra one life.There are also powers like Big ball,Long Bar,Slow ball remain hidden in random blocks . They will fall only after hitting those particular blocks.If the player catch those block while falling he will get the power for the upcoming ten seconds.The player has to complete all the Five levels to win the game.

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved and honorable chairman **MR. S. MEGANATHAN** and the chairperson **DR. M.THANGAM MEGANATHAN** for their timely support and encouragement.

We are greatly indebted to our respected and honorable principal **Dr. S.N. MURUGESAN** for his able support and guidance

No Words Of Gratitude Will Suffice For The Unquestioning Support Extended To Us By Our Head Of The Department **Dr.P.KUMAR M.E Ph.D.,** For Being Ever Supporting Force During Our Project Work

We also extend our sincere and hearty thanks to our internal guide **Mr.M.CHITHAMBARATHANU B.E.,M.Tech.,**for his valuable guidance and motivation during the completion of this project.

Our sincere thanks to our family members, friends and other staff members of information technology.

SARAVANAN M(180701219)

SAMINATHAN C(180701210)

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Araknoid is a arcade game which was developed in 19th century. It was a great entertainer to a large mob of all ages. For nearly 40 years it has been a unforgettable game in every ones lifetime.The main theme of the game is to hit all the blocks in the screen while balancing it with a bar without leaving it below. Will the player break all the blocks and win the game?

1.2 SCOPE OF THE WORK

Araknoid is purely an entertainment game created only for the enjoyment of the people. We have recreated a new version of the classic game which has been one of the most loved game of our childhood.It is a mind relaxing game made just for fun. Without giving a second to take a look out of the screen, the game keeps the player fully into the game. The main change from the previous versions is the breaking blocks function. Each time when we hit a block, a piece of the block falls down and the player should be cautious not to touch them. Few even has powers in them. By Catching those blocks the player get some advantages like slow ball,long bar,big ball which will make his gameplay easy

1.3 PROBLEM STATEMENT

We have to control the ball and direct it towards the appropriate place considering boundaries and collision conditions. We have to manage the attributes of the blocks, bar and the ball, be userfriendly and allow him to operate the game as easy as possible.

1.4 AIM AND OBJECTIVES OF THE PROJECT

The aim of the project is to recreate the araknoid game with added functions and modified gameplay. We have created this game with more different levels and new functions and with a little increased difficulty level.The distinct thing in this game is its hard gameplay. One would even say that it is impossible to complete even one level. Difficulty keeps accreting at each second.

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS

Processor	:	Pentium IV Or Higher
Memory Size	:	256 GB (Minimum)
HDD	:	40 GB (Minimum)

2.2 SOFTWARE SPECIFICATIONS

Operating System	:	WINDOWS XP or Higher
Front – End	:	python
Back - End	:	Microsoft access
Language	:	python

CHAPTER 3

MODULE DESCRIPTION

I. Pygame:

Pygame is a cross-platform set of Python modules designed for writing video games. It includes computer graphics and sound libraries designed to be used with the Python programming language.Pygame is primariry used to create games using python platform. Using pygame only the whole game is created . Pygame is not a built in module. It is necessary to install it before running this game.Pygame adds functionality on top of the excellent SDL library. This allows you to create fully featured games and multimedia programs in the python language. Pygame is highly portable and runs on nearly every platform and operating system. Pygame itself has been downloaded millions of times.

II. Stack:

Stack is a user defined module created to use attributes of ball and bar such as speed length and radius. The stack module consists of a stack which contains the above listed attributes. We retrieve and use these attributes in the specified levels.At each level we pop up the attributes and use them to create the images. The stack module consist of attributes of all the objects used in the game for all the levels

III. Time:

Time is a inbuilt module in python. This module provides a number of functions to deal with dates and the time. The sleep function from the time module is used in the game for a small pause at appropriate places. Python time module provides the ability to read, represent, and reset the time information in many ways.

Some of the useful time functions in Python:

- **time.time()**
- **time.clock()**
- **time.ctime()**
- **time.sleep()**
- **time.struct_time class**
- **time.strftime()**

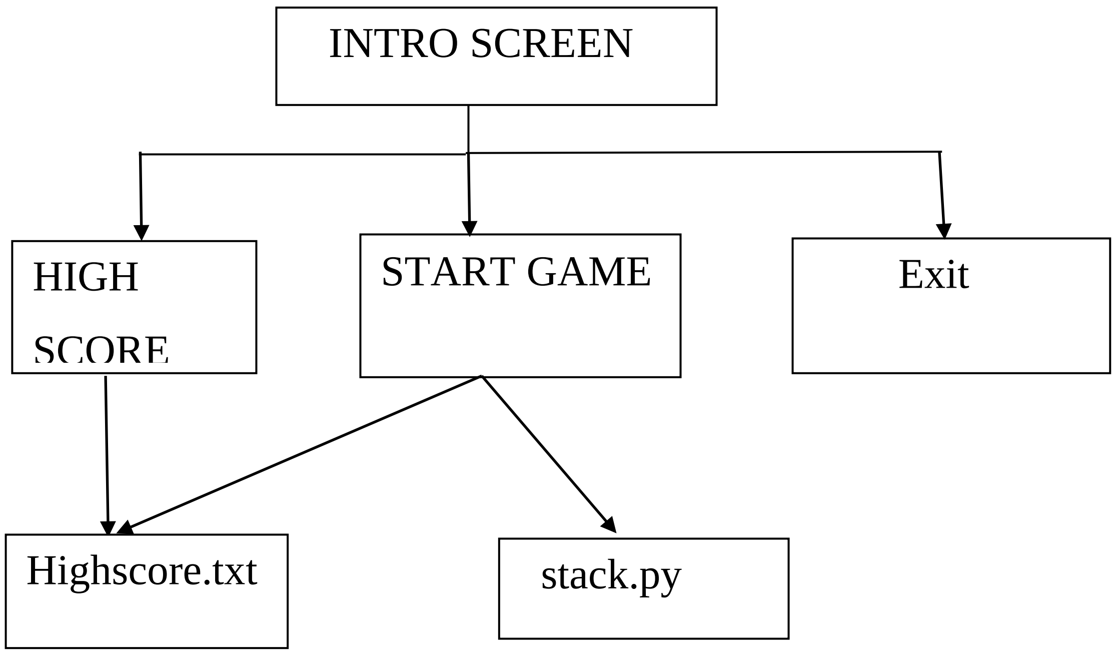
IV. Random:

Random is a inbuilt module in python. It is used to generate random numbers.Random module implements a pseudo-random number generator, and contains methods that let us directly solve many different programming issues where randomness comes into play.
Here it used for randomly allocating colors for the blocks and hiding random powers in random blocks .

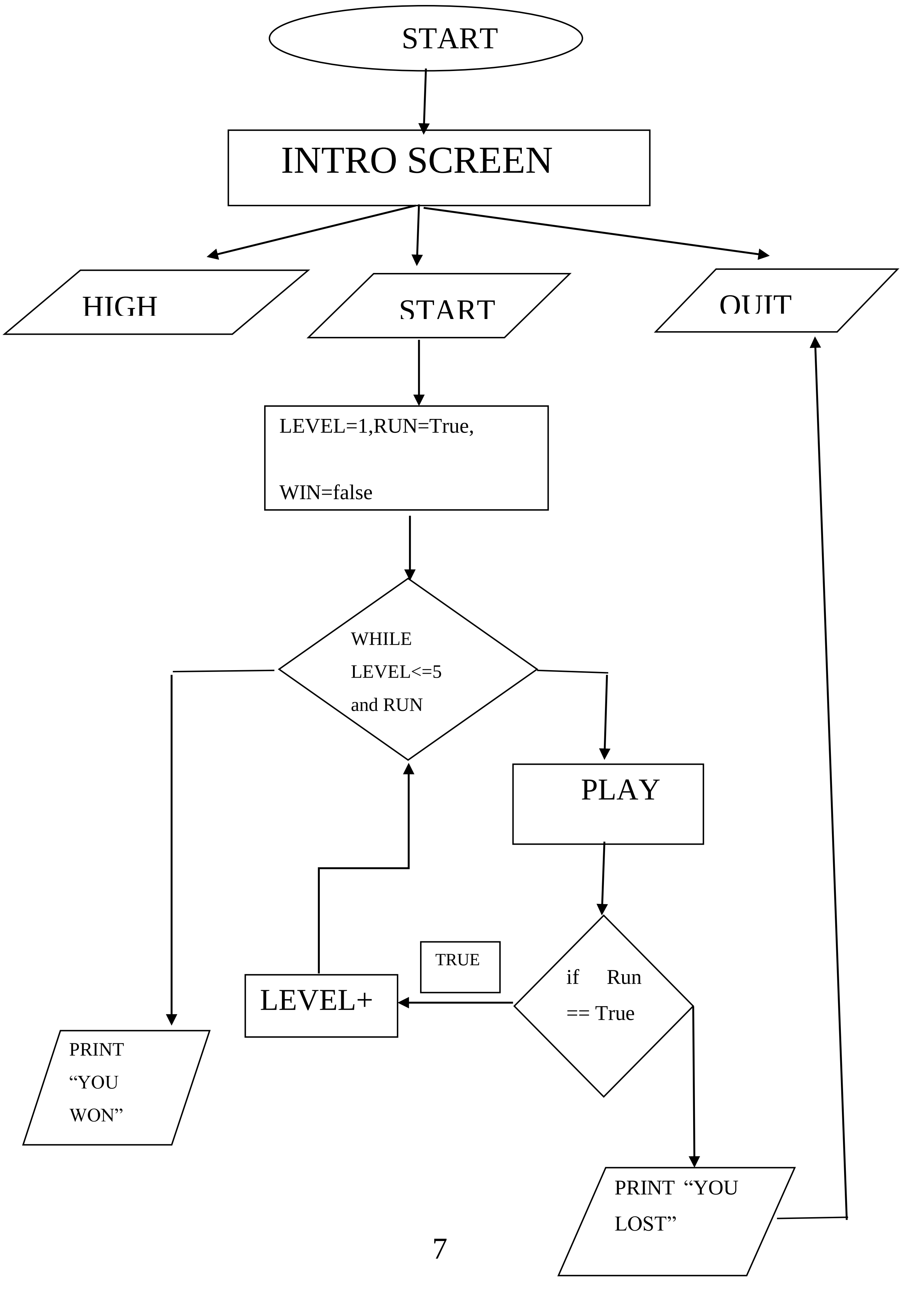
CHAPTER 4

SYSTEM DESIGN

4.1 ARCHITECTURE DIAGRAM



4.2 FLOW CHART



CHAPTER 5

SAMPLE CODING

```
import pygame
from pygame.locals import *
from stack import   *
import time
import random
#COLORS
BLACK=[0,0,0]
WHITE=[255,255,255]
GREY=[128,128,128]
RED=[255,0,0]
SILVER=[192,192,192]
PINK=(255,20,147)
YELLOW=(255,255,0)
BLUE=(0,255,255)
GREEN=(0,255,0)
VIOLET=( 108, 52, 131 )
ORANGE=(255, 117, 26)
GREEN2= [ 0,255,127]
PINK2=[144, 12, 63]
def reset():
#stack
```

```
    global
s,emptystr,highscore,show_score,launched,Start,win,speedvar,change_s
peed,changespeedinterval,bar,lostlife
    global
total_score,Level_Score,block_list,all_sprites_list,player_sprite,sprited_
block_rect,iron_block_list,hs,level,run
    global
counterval0,counterval1,counterval2,counter0,counter1,counter2
    counter0=False
    counter1=False
    counter2=False

    counterval0=counterval1=counterval2=0
    s=Stack()

    emptystr=str()
    highscore=int()
    show_score=bool()
    launched=bool()
    Start=bool()
    win=bool()
    speedvar=int()          #speedchange for each changespeedinterval
    change_speed=bool()         # whether the speed change at the instant
    changespeedinterval=int()      # for no.of.Level_Score increase, the
speed will increase
    bar=0
    lostlife=0
    total_score=0      # total Level_Score
    Level_Score=0              # Level_Score of each level
      #Group
    block_list=pygame.sprite.Group()
    all_sprites_list=pygame.sprite.Group()
    player_sprite=pygame.sprite.Group()

     #List
    sprited_block_rect=list()
    iron_block_list=list()
    power_block_list=list()
    hs=0
    with open("HighScore.txt",'r') as f:
                        hs=f.read()
    clock=pygame.time.Clock()
    level=1
    run=True
# self.life=3
class Block(pygame.sprite.Sprite):
    def  __init__(self,color,w,h):
        super().__init__()
        self.color=color
        self.image=pygame.Surface([w,h])
        self.image.fill(self.color)
        self.rect=self.image.get_rect()
        self.health= 200 if color == [128,128,128] else 100

    def update(self):
        self.rect.y+=10
```

```
class Bar(pygame.sprite.Sprite):
    def __init__(self,color,w,h):
          super().__init__()
          self.length=w
          self.height=h
          self.color=color
          self.image=pygame.Surface([self.length,self.height])
          self.image.fill(self.color)
          self.rect=self.image.get_rect()
          self.rect.x= screen_width//2
          self.rect.y= screen_height - (self.height-1)     # (h is the bar's
length)
class Ball(pygame.sprite.Sprite):
    def __init__(self,color,radius,speed,speedvar):
        global lostlife
        super().__init__()
        self.color=color
        self.radius=radius
        self.x_speed=self.y_speed=speed
        self.speedvar=speedvar
        self.life=3-lostlife

self.image=pygame.Surface([self.radius,self.radius],pygame.SRCALPH
A)  #SCRALPHA for TRANSOARENCY
        self.image.fill(BLUE)
        self.image.set_colorkey(WHITE)
        self.rect=self.image.get_rect()
        pygame.draw.ellipse(self.image,self.color,self.rect)
```

```
def update(self):
   global Start
   global changespeedinterval
   global changespeed
   global win
   global launched
   global bar
   global lostlife
   global fall_list
   global counterval0,counterval1,counterval2
   global counter0,counter1,counter2
   global ball
   global specialcondition
   index=0
   if self.rect.x+self.radius > screen_width or self.rect.x < 0 :
       self.x_speed*=-1
   #index=self.rect.collidelistall(all_block_rect)
   hit=True if len(sprited_block_rect) >0 else False
   # hit become True if the ball collides with any blocks
   if hit :
         self.rect.y+=10
         self.y_speed*=-1
   if self.rect.y< 0 or self.rect.colliderect(bar.rect) :
         self.y_speed*=-1
         if bar.rect.x<= self.rect.x <= bar.rect.x + bar.length//3 :
            if self.x_speed > 0 :
             self.x_speed*=-1
            elif   bar.rect.x+bar.length//2   <=   self.rect.x   <=   bar.rect.x   +
bar.length :
                if self.x_speed < 0 :
                  self.x_speed*=-1
      if self.rect.y> screen_height and self.life >0.5 and launched :
           self.life-=1
           lostlife+=1
           time.sleep(0.5)
           self.rect.x=bar.rect.x+30
           self.rect.y=bar.rect.y-30
           launched=False
      if self.rect.y+self.radius > screen_height and self.life <=1:
          Start=False
          win=False

      for block in fall_list:
              if block.rect.y<screen_height:
                 block.update()
              if block.rect.y>screen_height:
                   fall_list.remove(block)
              if block.rect.colliderect(bar.rect):
                 for block in fall_list:
                     block.rect.y+=screen_height
                     fall_list.clear()
                 if block.color==PINK2:
                     counter1=True
                     colorb=bar.color
```

```
    heightb=bar.height
    lenghtb=bar.length
    xpot=bar.rect.x
    ypot=bar.rect.y
    all_sprites_list.remove(bar)
    bar=Bar(PINK2,lenghtb+18,20)
    all_sprites_list.add(bar)
    bar.rect.x=xpot
    bar.rect.y=ypot
    specialcondition=True
    break
elif block.color==BLACK: #slow ball
    counter2=True
    self.x_speed*=0.5
    self.y_speed*=0.5
    break
elif block.color==WHITE:
    counter0=True

    rad=self.radius
    speed=self.x_speed
    speedv=self.speedvar
    xpo=self.rect.x
    ypo=self.rect.y
    all_sprites_list.remove(ball)
    player_sprite.remove(ball)
    ball=Ball(GREEN,rad+30,speed,speedv)
    ball.rect.x=xpo
```

```
                ball.rect.y=ypo
                all_sprites_list.add(ball)
                player_sprite.add(ball)
                self.update()
                self.radius+=30
                break
            if self.life > 0.5:
             self.life-=0.25
             lostlife+=0.25
             self.rect.x=bar.rect.x+30
             self.rect.y=bar.rect.y-30
             launched=False
            else:
              Start=False
              win=False
            break
if counter0 or counter1 or counter2:
 if counterval0>=1000:#reverse condition
     counterval0=0
     counter0=False
     rad=self.radius
     speed=self.x_speed
     speedv=self.speedvar
     xpo=self.rect.x
     ypo=self.rect.y
     all_sprites_list.remove(ball)
     player_sprite.remove(ball)
     ball=Ball(GREEN,rad-30,speed,speedv)
```

```
        ball.rect.x=xpo
        ball.rect.y=ypo
        all_sprites_list.add(ball)
        player_sprite.add(ball)
        self.radius-=30
        self.update()
      elif counterval1>=1000:
        counter1=False
        counterval1=0
        xpot=bar.rect.x
        ypot=bar.rect.y
        all_sprites_list.remove(bar)
        bar=Bar(SILVER,bar_length,20)
        all_sprites_list.add(bar)
        bar.rect.x=xpot
        bar.rect.y=ypot
        specialcondition=False
      elif counterval2>=1000:
        counterval2=0
        counter2=False
        self.x_speed*=2
        self.y_speed*=2
    if len(block_list) <= 0 :    #when all the blocks are demolished,
finish the level
          Start=False
          win=True
    if Level_Score == changespeedinterval :         #for every changespeedinterval value  increment the speed of the ball
        changespeed=True

    if Level_Score%10== 0 and Level_Score>=10 and changespeed: #Level_Score>=10 is not to increment the speed for values less than zero since mod 10 of every value below 11 is 0
        self.x_speed*=self.speedvar
        self.y_speed*=self.speedvar
        changespeed=False
        changespeedinterval+=10
    # finally increment x and y speed
    self.rect.x+=self.x_speed
    self.rect.y-=self.y_speed
hs=0
with open("HighScore.txt",'r') as f:
            hs=f.read()
def introscreen():
  global hs
  global Start
  global emptystr
  global show_score
  global level

  titlefont=pygame.font.SysFont("Comic sans MS",48)
  font=pygame.font.SysFont(None,48)
  title=titlefont.render("Araknoid",True,BLACK,GREEN)
  title_rect=title.get_rect()
```

```
play_text=font.render("Play",True,BLACK,GREEN)
play_rect=play_text.get_rect()
quit_text=font.render("Quit",True,BLACK,GREEN)
quit_rect=quit_text.get_rect()
high_score=font.render("High Score",True,BLACK,GREEN)
high_score_rect=high_score.get_rect()
show_hscore=font.render(emptystr,True,BLACK)
show_hscore_rect=show_hscore.get_rect()
title_rect.center=screen.get_rect().center
title_rect.centery=screen.get_rect().centery - 130
play_rect.center=screen.get_rect().center
quit_rect.centerx=screen.get_rect().centerx
quit_rect.centery=screen.get_rect().centery +60
high_score_rect.centerx=screen.get_rect().centerx
high_score_rect.centery=screen.get_rect().centery +120
show_hscore_rect.centerx=screen.get_rect().centerx
show_hscore_rect.centery=screen.get_rect().centery +200
screen.blit(title,title_rect)
screen.blit(play_text,play_rect)
screen.blit(quit_text,quit_rect)
screen.blit(high_score,high_score_rect)
screen.blit(show_hscore,show_hscore_rect
for event in pygame.event.get():
    mpos=pygame.mouse.get_pos()
    if event.type == pygame.QUIT:
        pygame.quit()
        quit()
        elif event.type==pygame.MOUSEBUTTONDOWN and
event.button == 1:
            if play_rect.collidepoint(mpos) :
                Start=True
                breeak
            elif quit_rect.collidepoint(mpos):
                pygame.quit()
                quit()
            # setting highscore button as ON AND OFF switch


            elif high_score_rect.collidepoint(mpos) and not show_score  :
                 emptystr=str(hs)
                 show_score=True
            elif  high_score_rect.collidepoint(mpos) and show_score  :
                emptystr=str()
                show_score=False
def Start_Game(iron):
    # iron arguments says number of iron block rows
    pygame.mouse.set_visible(False)
    posy=0
    colourlist=[RED,SILVER,PINK,YELLOW,VIOLET,GREEN2]
    power_block_list=[]
    powlist=['bigball','longbar','slowball']
    for i in range(5):
        posx=0
        for j in range(10):
            color= GREY if i==0 else colourlist[random.randint(0,5)]
```

```
                block=Block(color,47,25)
                block.rect.x=posx
                block.rect.y=posy
                #screen.blit(block.image,block.rect)
                block_list.add(block)
                all_sprites_list.add(block)
                if i ==0:
                    iron_block_list.append(block)
                #all_block_rect.append(block.rect)
                rand= random.randint(0,7)
                if (rand <3) and i!=0 :
                    power_block_list.append(block)
                    if rand==0:
                        block.color=WHITE #for larger ball
                    elif rand==1:
                        block.color=PINK2 #for longer bar
                    elif rand==2:
                        block.color=BLACK #for slower ball
                posx+=50
        posy+=28

    all_sprites_list.add(bar)
def GameOver(message,Escape):
    global  highscore
    global total_score
    if Escape:
        score=total_score
    else:
        score=highscore
    basicfont=pygame.font.SysFont('Comic Sans MS',46)
    scorefont=pygame.font.SysFont(None,50)
    text=basicfont.render(message,True,WHITE,BLUE)                    #
render(text,anti-aliasing,color,background)
    textrect=text.get_rect()
    textrect.centerx=screen.get_rect().centerx
    textrect.centery=screen.get_rect().centery-50
    show_score=scorefont.render(str(score),True,RED,GREEN)
    score_rect=show_score.get_rect()
    score_rect.centerx=screen.get_rect().centerx
    score_rect.centery=screen.get_rect().centery
    intro=scorefont.render("Press          Enter          to          Play
Again",True,BLACK,BLUE)
    introrect=intro.get_rect()
    introrect.centerx=screen.get_rect().centerx
    introrect.centery=screen.get_rect().centery+60
    screen.blit(text,textrect)
    screen.blit(show_score,score_rect)
    screen.blit(intro,introrect)
    for event in pygame.event.get():
            if event.type == pygame.KEYDOWN and event.key ==
pygame.K_RETURN :
                return True
def showscore(ball):
      font=pygame.font.SysFont(None,40)
      levelfont=font.render("LEVEL "+str(level),True,BLACK,BLUE)
      lifefont=font.render("LIFE:"+str(ball.life),True,BLACK,BLUE)
```

```
scorefont=font.render("SCORE:"+str(total_score),True,BLACK,BLUE)
    level_rect=levelfont.get_rect()
    life_rect=lifefont.get_rect()
    score_rect=scorefont.get_rect()
    level_rect.centerx=screen.get_rect().centerx
    level_rect.centery=screen.get_rect().centery
    score_rect.centerx=screen.get_rect().centerx+70
    score_rect.centery=screen.get_rect().centery+40
    life_rect.centerx=screen.get_rect().centerx-60
    life_rect.centery=screen.get_rect().centery+40
    screen.blit(levelfont,level_rect)
    screen.blit(scorefont,score_rect)
    screen.blit(lifefont,life_rect)


pygame.init()
screen_width=500
screen_height=500
screen=pygame.display.set_mode([screen_width,screen_height])
pygame.display.set_caption("ARAKNOID")
clock=pygame.time.Clock()
level=1
run=True
fall_list=[]
#is_falling=bool()
#fall_img=pygame.Surface([30,30])
#fall_img.fill(RED)
```

```
#fall_rect=fall_img.get_rect()

def NextStage(speedVar,bar_len,ball_rad,ball_speed):
        global emptystr
        global hs # high score to be stored in file
        global highscore
        global show_score
        global Start
        global win
        global change_speed
        global changespeedinterval
        global speedvar
        global bar
        global Level_Score
        global total_score
        global level
        global run
        global launched
        global fall_list
        global counterval0,counter0
        global counterval1,counter1
        global counterval2,counter2
        global ball
        global specialcondition
        bar_length=bar_len
        global bar_length
        specialcondition=False
       # global fall_rect
```

```
speedvar=speedVar
#bar
bar=Bar(SILVER,bar_len,20)
emptystr=str()
show_score=False
Start=False  if level==1 else True
win=False

#execute at the beginning level only
while not Start and level==1:
    for event in pygame.event.get():
        if  event.type == pygame.QUIT :
            pygame.quit()
            exit()
            break
    screen.fill(BLUE)
    introscreen()
    pygame.display.flip()
block_hit_list=[]
ball=Ball(GREEN,ball_rad,ball_speed,speedvar)
ball.rect.x=screen_width//2 + 20
ball.rect.y=screen_height-45
all_sprites_list.add(ball)
player_sprite.add(ball)

#creating blocks

Start_Game(level):
launched=False
while Start:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
        if event.type==pygame.KEYDOWN and event.key==pygame.K_SPACE and not launched :
            player_sprite.update()
            launched=True
        #for quitting in middle of the running game
        if event.type == pygame.KEYDOWN and event.key == pygame.K_ESCAPE:
            run=False
            changeHighscore(total_score,hs)
            while True:
                screen.fill(BLUE)
                pygame.mouse.set_visible(True)
                if GameOver("You Escaped , Coward !!",True):
                    return
                pygame.display.update()
    if counter0:
        if counterval0<=1000: counterval0+=1
    elif counter1:
        if counterval1<=1000: counterval1+=1
    elif counter2:
        if counterval2<=1000: counterval2+=1
    keys=pygame.key.get_pressed()
    if keys[pygame.K_LEFT] and bar.rect.x>0 :
```

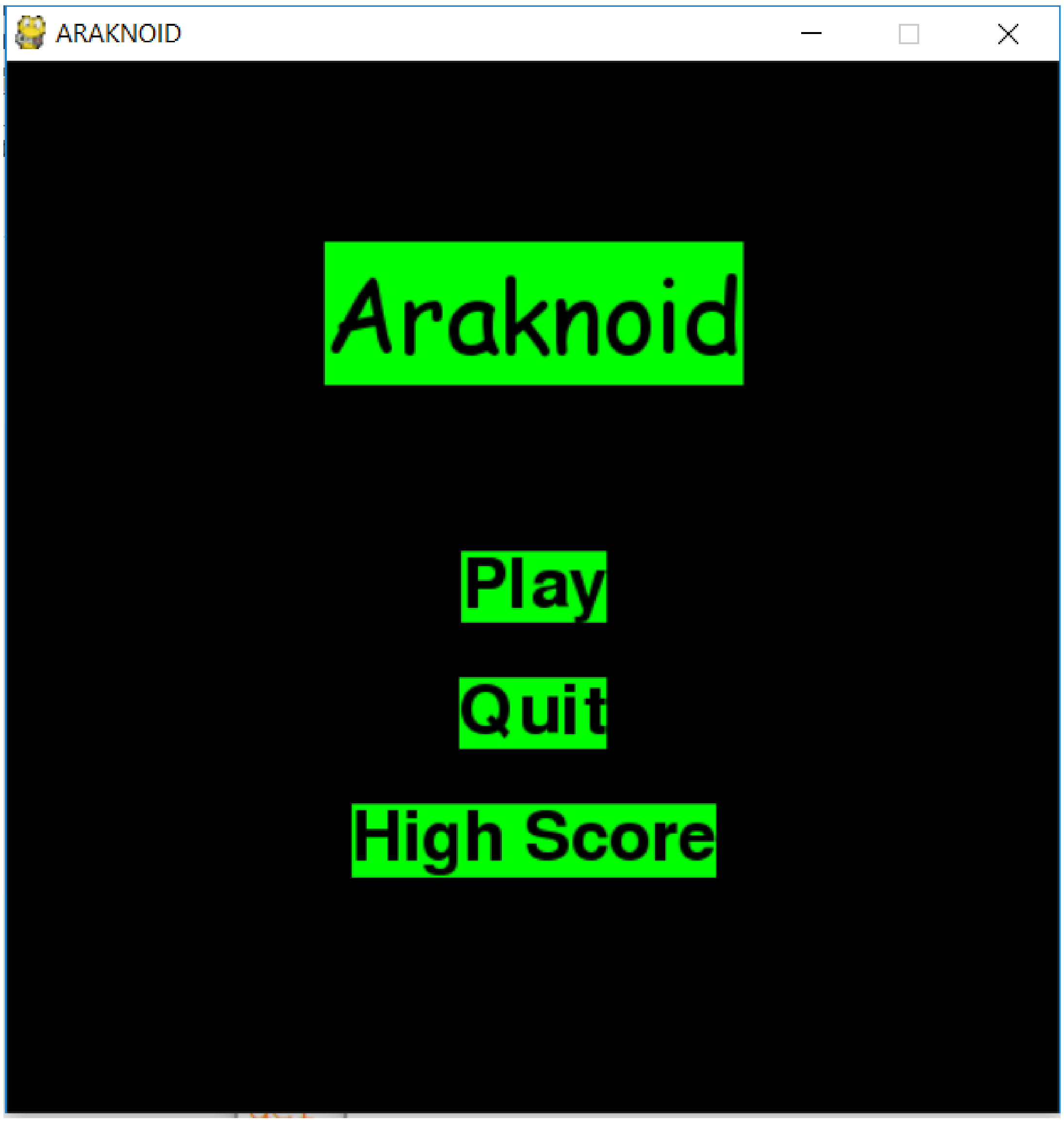
```
            if not launched :
              ball.rect.x-=6   # to move the ball with respect to bar
when it is not launched yet
            bar.rect.x-=7
        if keys[pygame.K_RIGHT] and bar.rect.x<screen_width-
bar.length:
             if not launched :
               ball.rect.x+=6
             bar.rect.x+=7
        screen.fill(BLUE)
        remove=bool()
        iron=[block.rect for block in iron_block_list]
        if ball.rect.collidelist(iron)+1 :          # the
rect1.collidelist(rect2list) will return the index of the total rectangles of
rect2list that collide with rect1
         for block in iron_block_list:
           if ball.rect.colliderect(block.rect):
             remove=False
             block.image.fill(RED)
             iron_block_list.remove(block)
             block.color=RED
             break
         '''for block in power_block_list:
            if ball.rect.colliderect(block.rect):
             remove=False
             block.image.fill(RED)
             power_block_list.remove(block)
             block.color=PINK2'''
            else:
               remove=True

block_hit_list=pygame.sprite.spritecollide(ball,block_list,remove)

            for block in block_hit_list:
                 if block.color !=GREY and len(fall_list)<3:
                    block.rect.width-=35
                    block.rect.height-=5
                    fall_list.append(block)
                 if specialcondition==True: block.color=PINK2
                 sprited_block_rect.append(block.rect)
                 total_score+=1
                 Level_Score+=1
            if launched:
              player_sprite.update()
              sprited_block_rect.clear()
            all_sprites_list.draw(screen)
            player_sprite.draw(screen)

            for block in fall_list:
               pygame.draw.rect(screen,block.color,block.rect)
            showscore(ball)
            clock.tick(60)
            pygame.display.flip()
         while not Start :
            ball.image.fill(BLUE)
            bar.image.fill(BLUE)
            player_sprite.remove(ball)   # del ball
            pygame.mouse.set_visible(True)
            screen.fill(BLUE)
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                     pygame.quit()
            highscore=total_score
            if win:
                if gotonextlevel():
                    break
            else:
                run=False
                changeHighscore(highscore,hs)
                if GameOver('GAME OVER',False):
                        return
            pygame.display.flip()
        Level_Score=0
#NextStage(ch_s,ch_in,bar_len,ball_rad,ball_speed)
def gotonextlevel():
    global level
    if level != 5:
        font=pygame.font.SysFont(None,28)
        nextlev=font.render("Press ENTER to continue",True,BLACK,GREEN)
        nextlev_rect=nextlev.get_rect()
        nextlev_rect.center=screen.get_rect().center
        for event in pygame.event.get():
          if event.type == pygame.KEYDOWN and event.key == pygame.K_RETURN :
                level+=1
                return True
        screen.blit(nextlev,nextlev_rect)
        pygame.display.update()
    else:
        font=pygame.font.SysFont(None,28)
        nextlev=font.render("You Won\n Score :"+str(total_score),True,BLACK,GREEN)
        nextlev_rect=nextlev.get_rect()
        nextlev_rect.center=screen.get_rect().center
        screen.blit(nextlev,nextlev_rect)
        return

def changeHighscore(highscore,hs):
  if highscore>int(hs):
            with open("HighScore.txt",'w') as f:
                f.write(str(highscore))


reset()
def Next():
  global run,level,lostlife,s
  while run and not s.isEmpty() :
    arg=s.pop()
    if level>1:
     lostlife-=1
    NextStage(arg[0],arg[1],arg[2],arg[3])
```

```
while  True:
    if run :
     Next()
    else:
        reset()
```

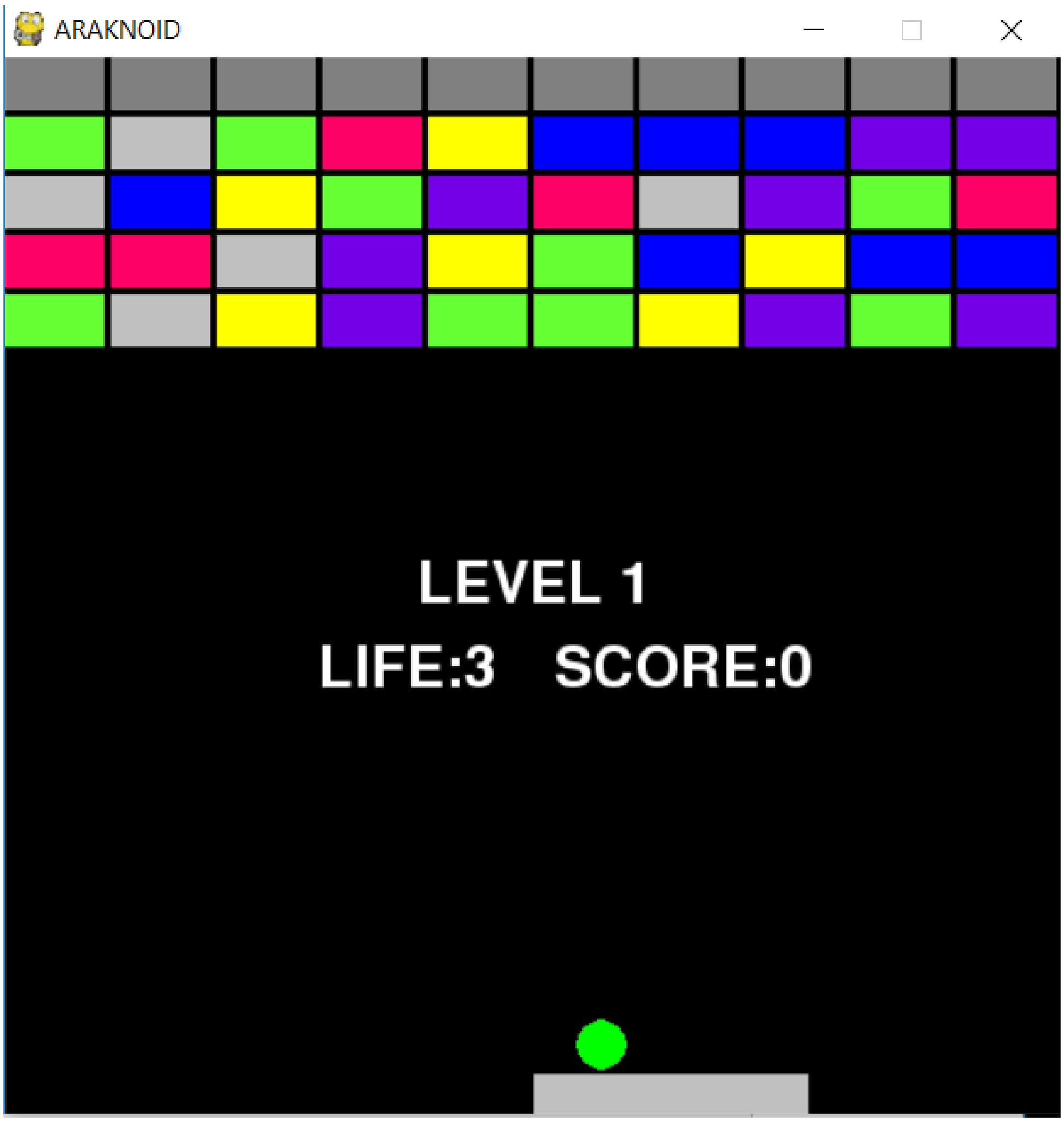
CHAPTER 6

SCREEN SHOTS

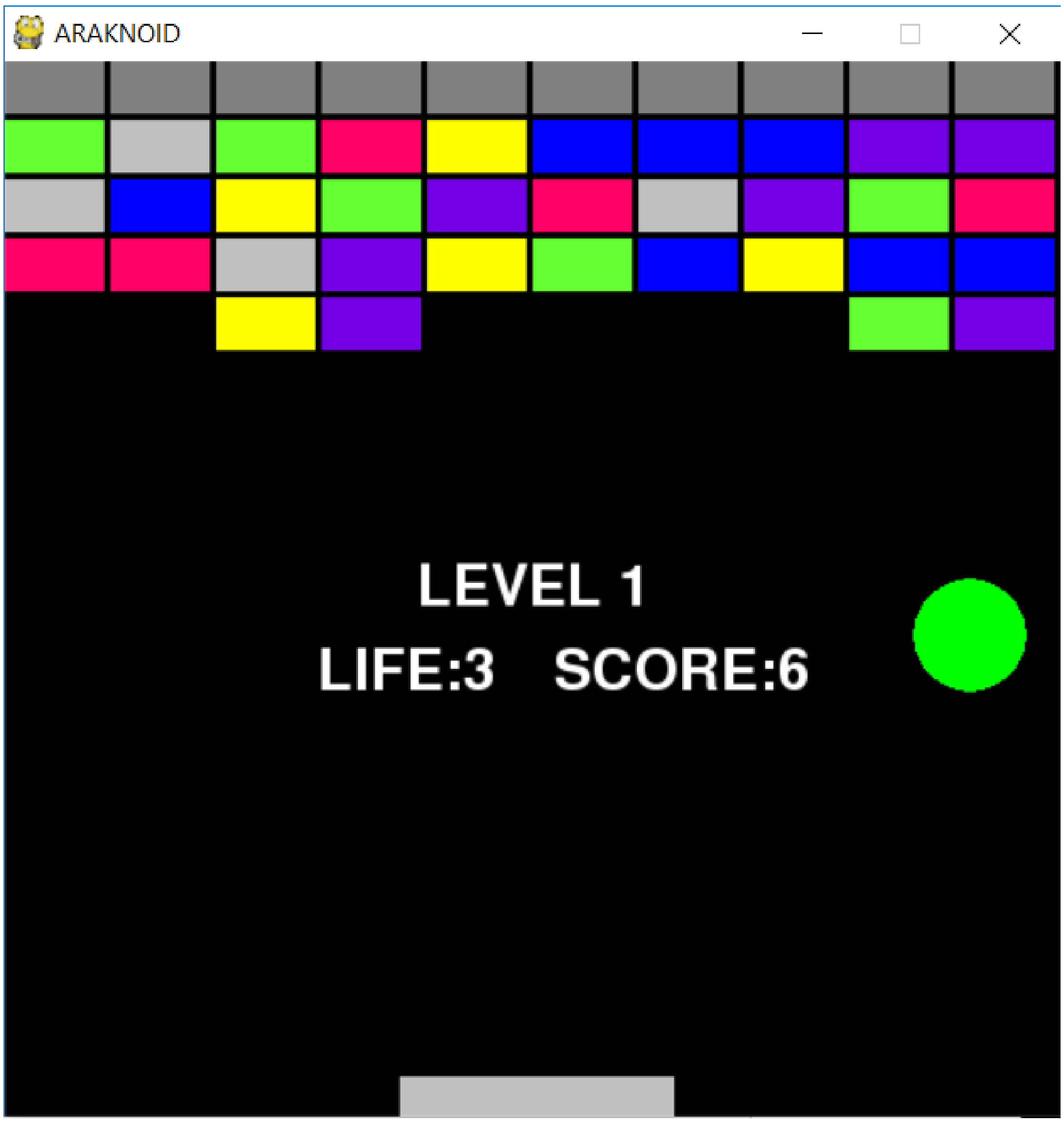
INTRO SCREEEN



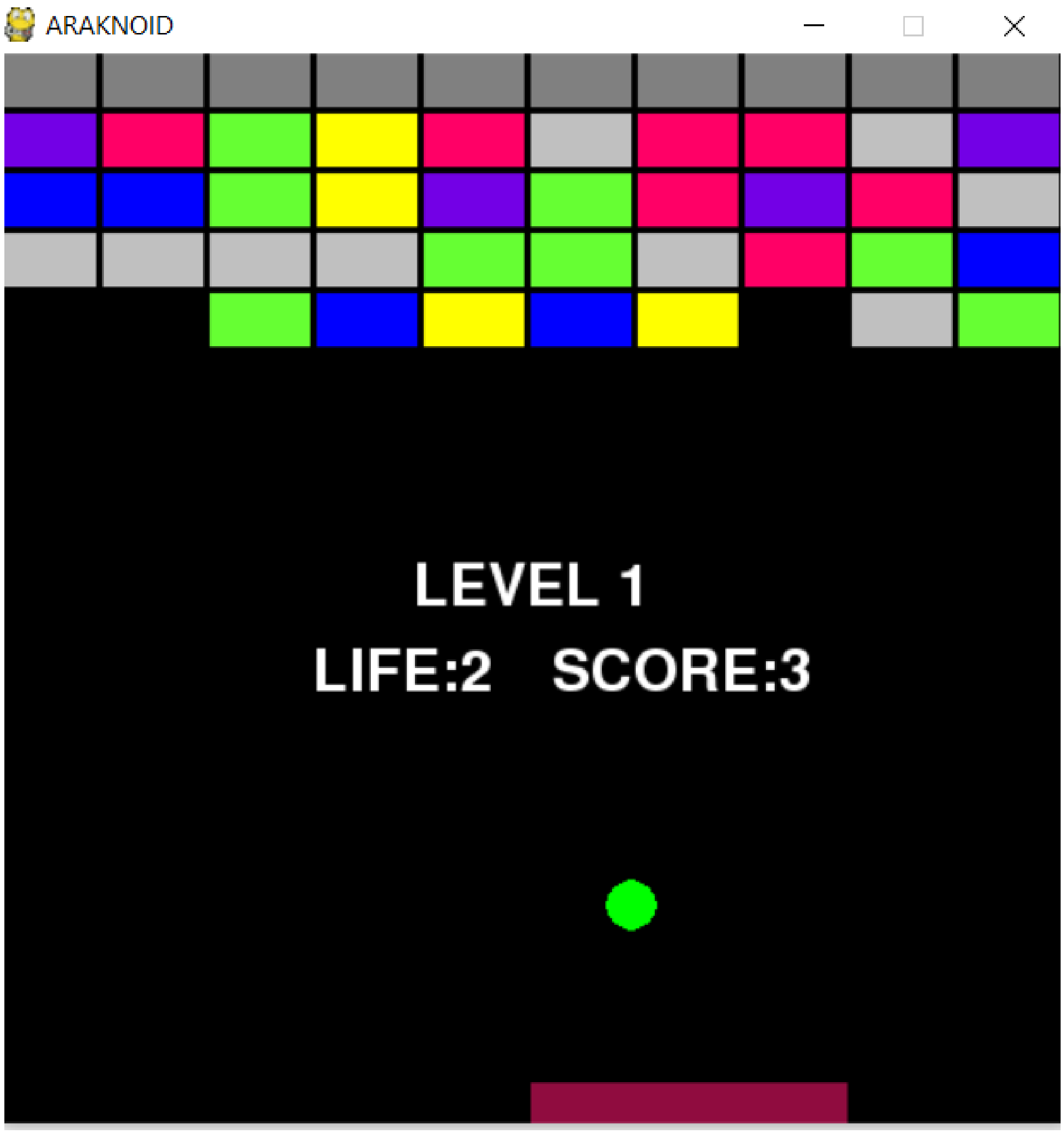
Game Starts



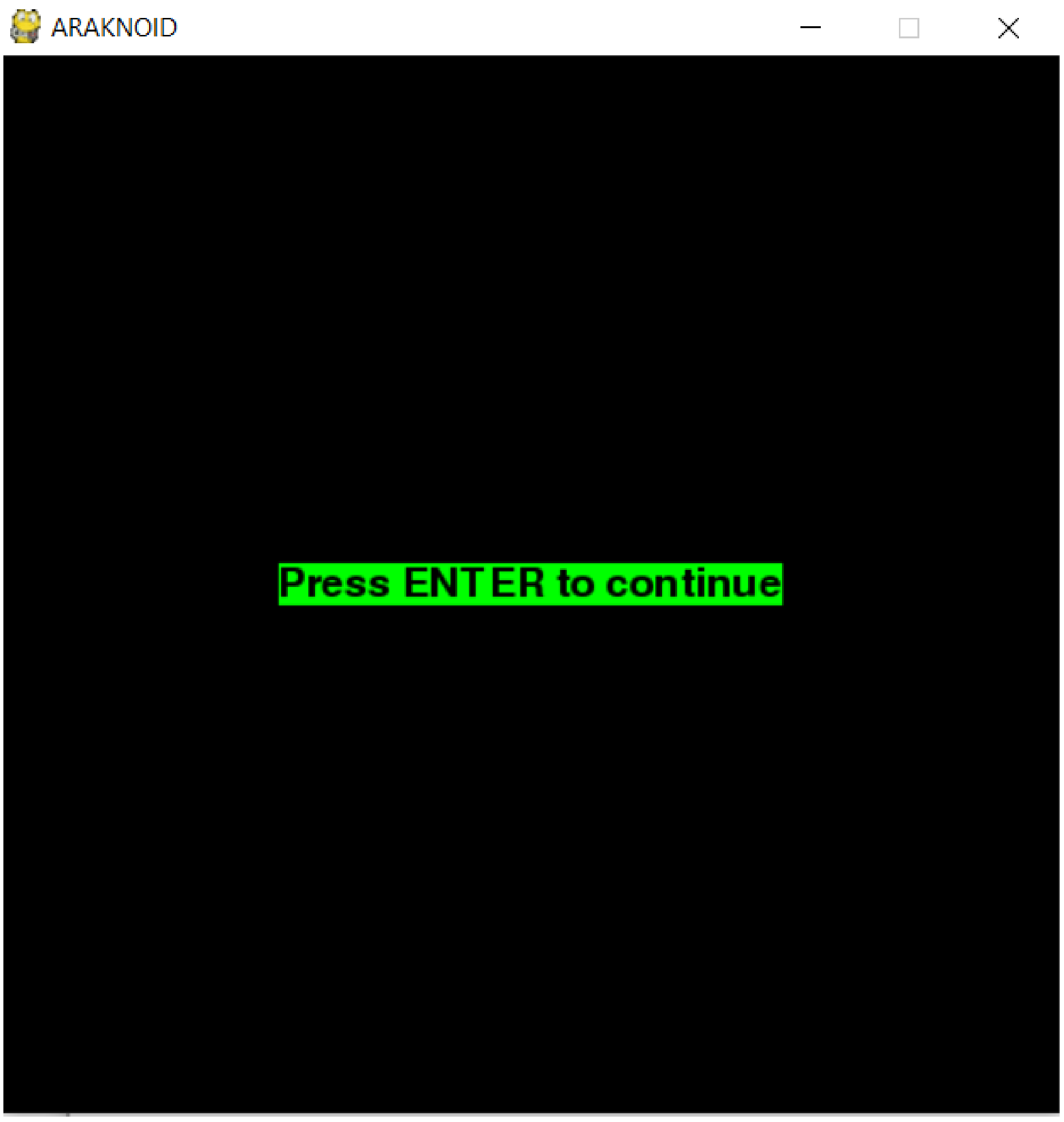
Big Ball Power



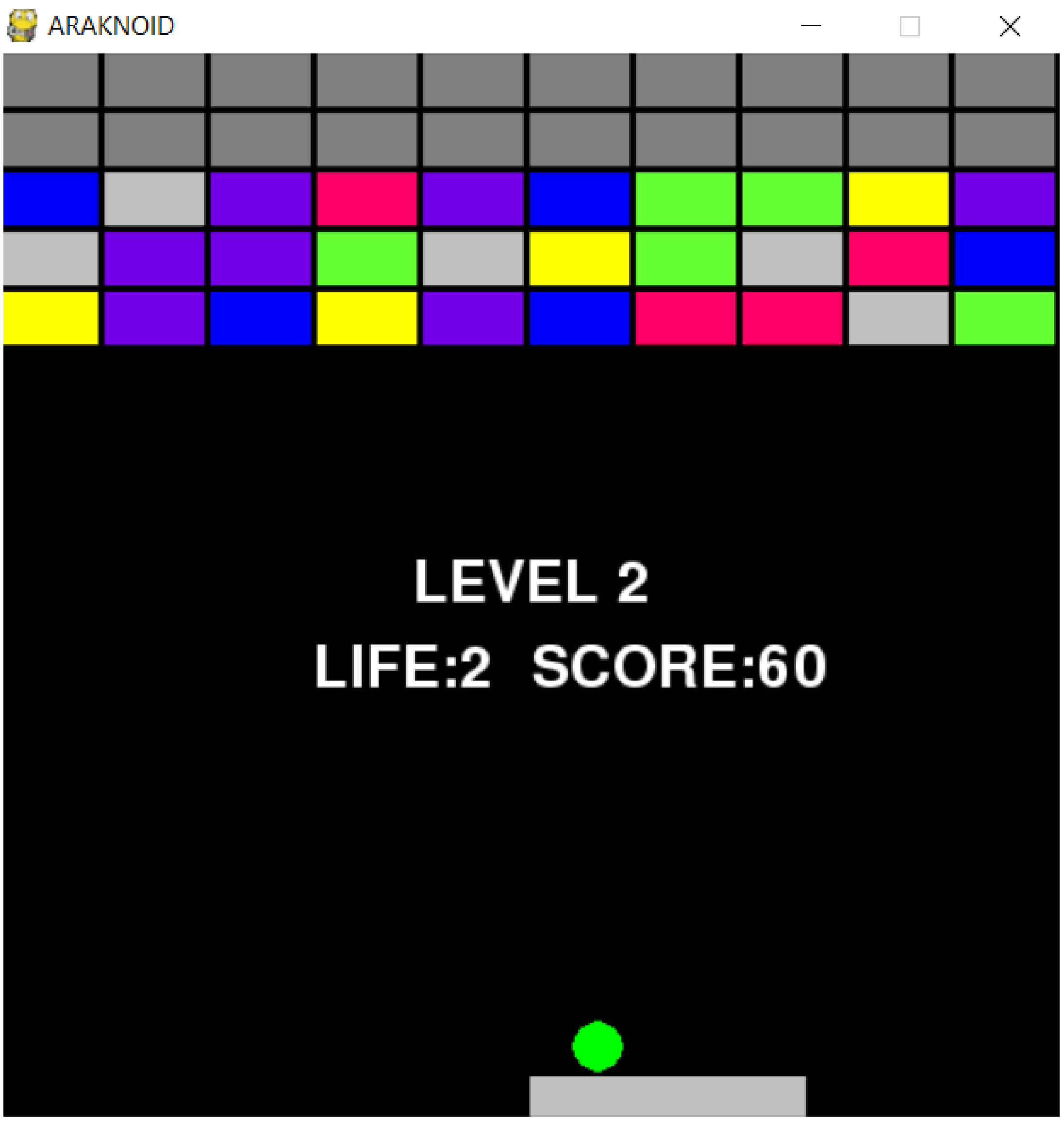
S

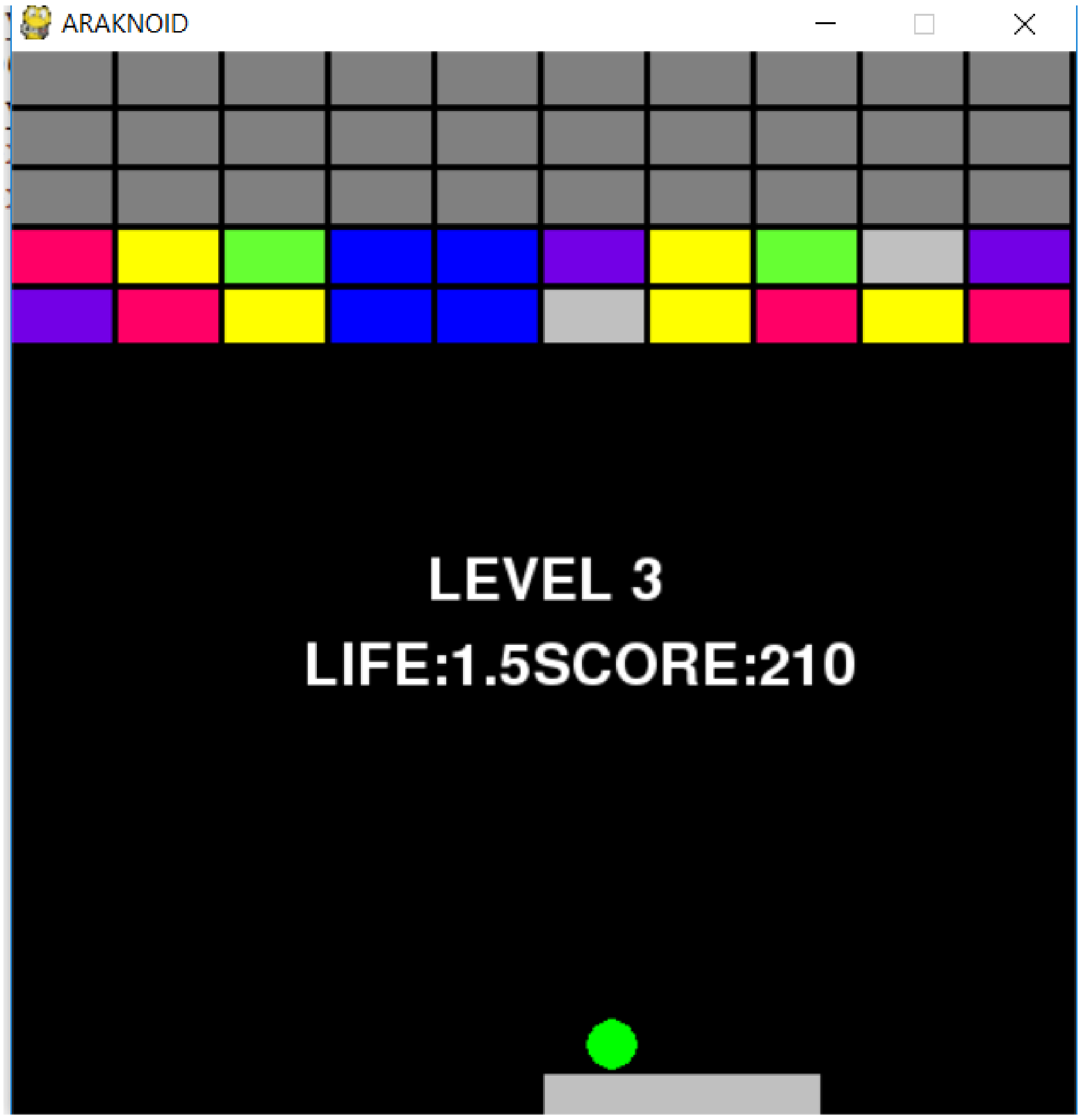


NEXT LEVEL



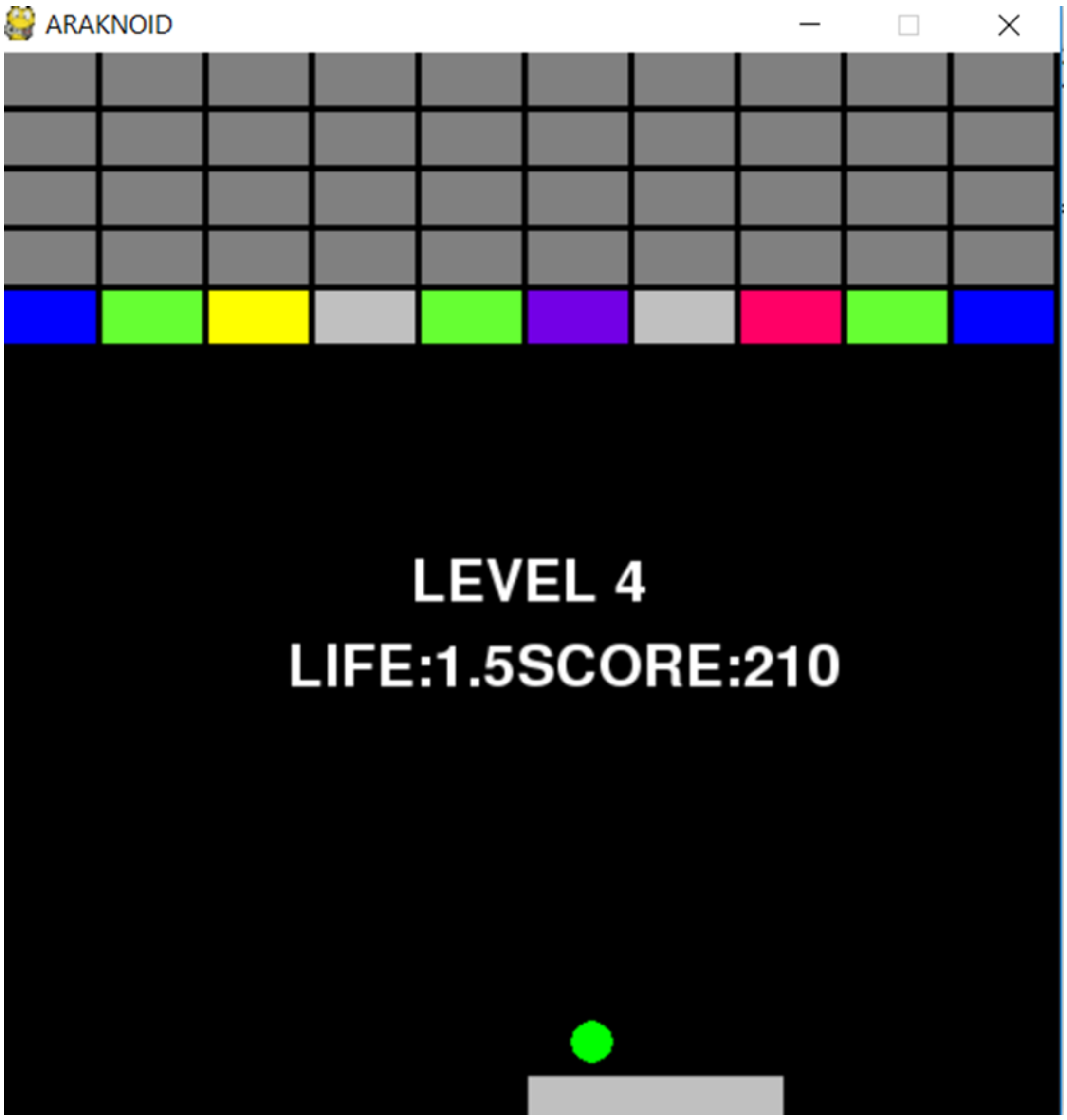
Press ENTER to continue

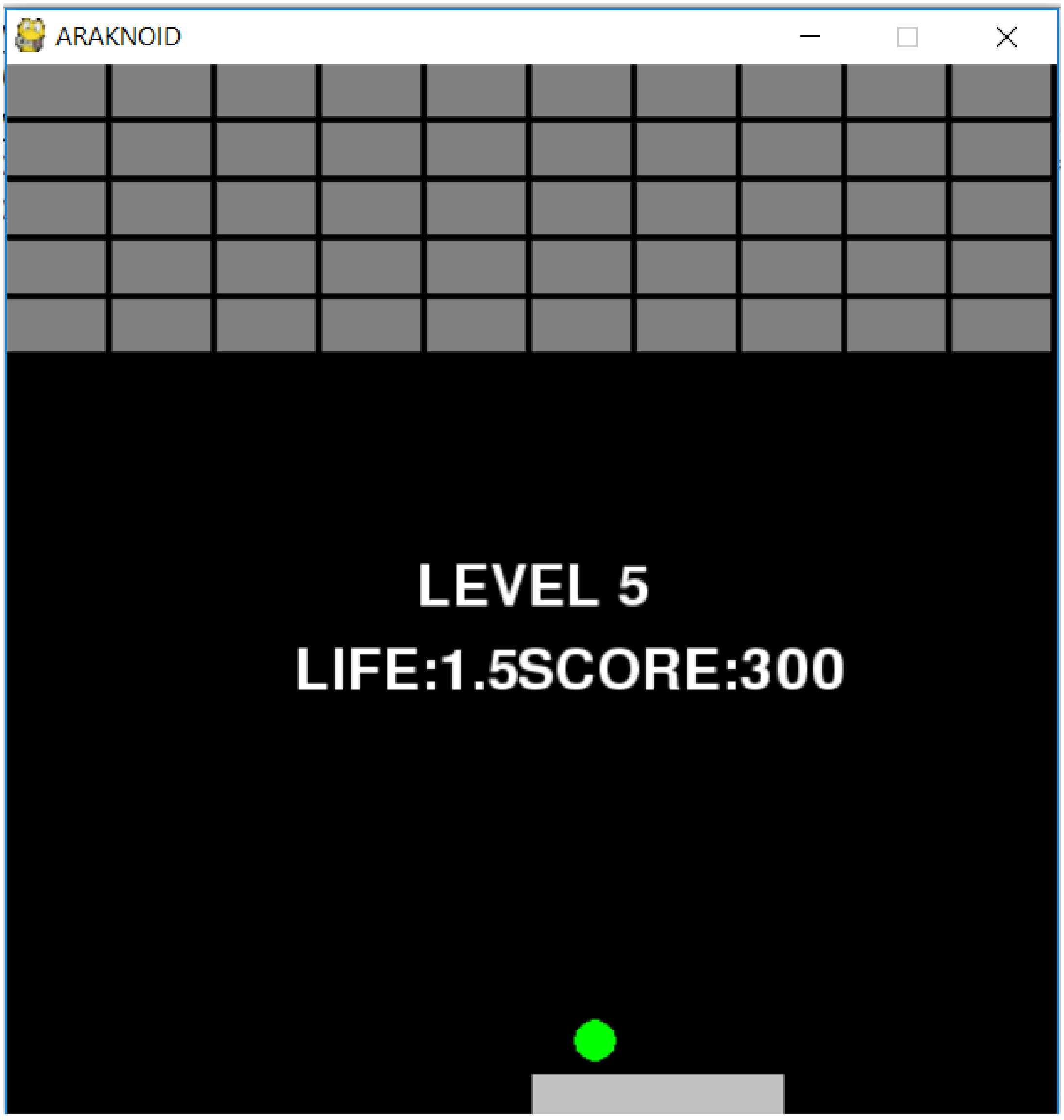

ARAKNOID
LEVEL 2
LIFE:2 SCORE:60

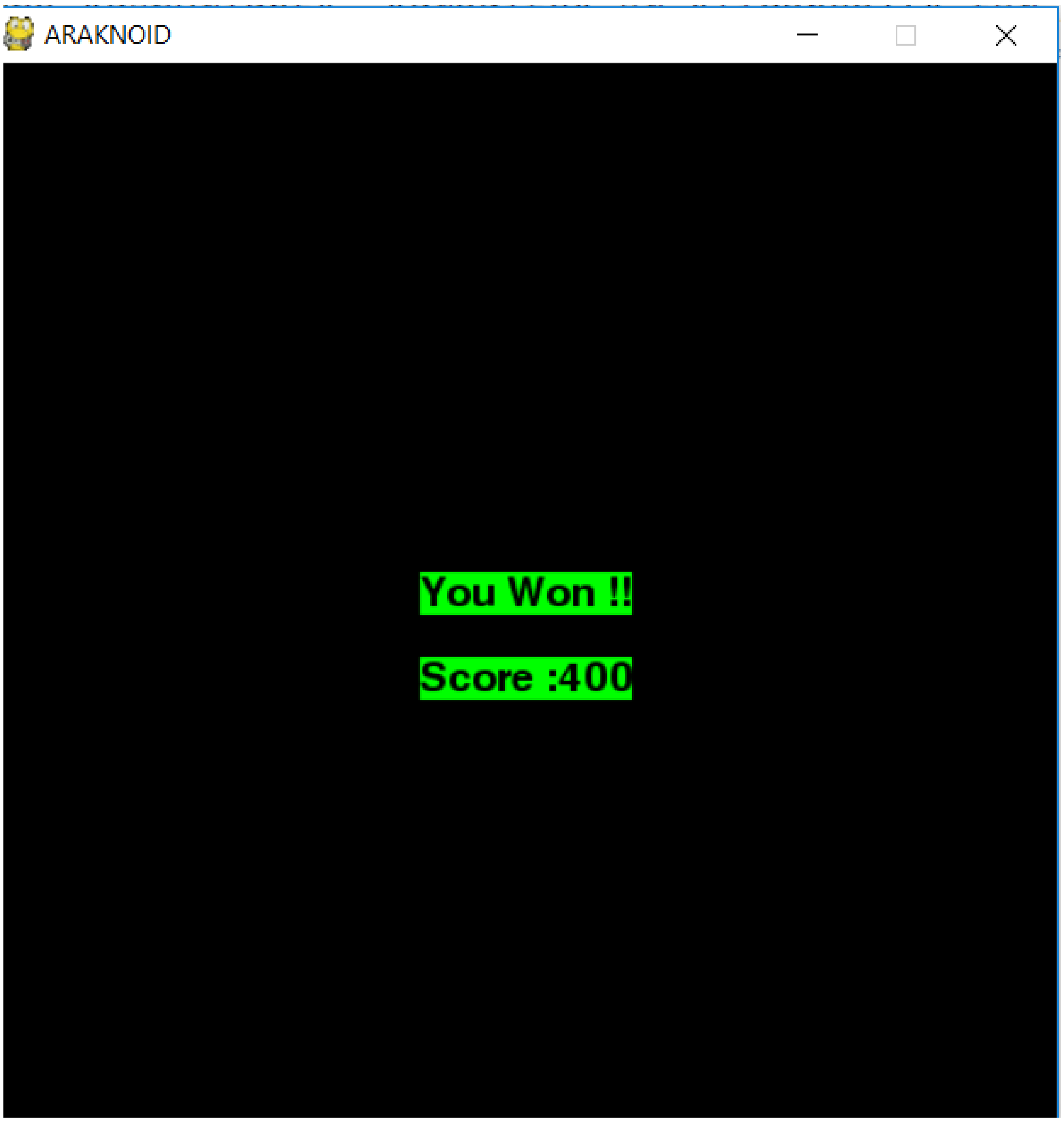


LEVEL 3

LIFE:1.5SCORE:210


ARAKNOID
LEVEL 4
LIFE:1.5SCORE:210


ARAKNOID
LEVEL 5
LIFE:1.5SCORE:300



```
You Won !!

Score :400
```

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

We made this game with only limited functions and levels. In future we can upgrade the gameplay in a more user friendly manner. We will also make this game in 3 dimensional format as here we have did this game only with 2 dimensional manner. In future we are willing to make more powers and different arrangement of blocks in more colourfull way. We are also willing to add more user customizing attributes like selecting the colour of the ball and bar.

REFERENCE

1. https://stackoverflow.com

2. https://programarcadegames.com

3. http://pygametutorials.wikidot.com

4. https://pygame.org

5. https://www.edureka.co/blog/pygame-tutorial

6. https://nerdparadise.com/programming/pygame/