## PROFESSIONAL TRAINING REPORT

**at**

**Sathyabama Institute of Science and Technology**

**(DEEMED TO BE UNIVERSITY)**

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

by

**KOMMANA BHANU SIVA KUMAR**

**(Reg. No. 37110352 )**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SCHOOL OF COMPUTING**

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119, TAMILNADU.**

**AUGUST 2019**

 **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

**(Established under Section 3 of UGC Act, 1956)**

**Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600119**

[**www.sathyabamauniversity**](http://www.sathyabamauniversity)**.ac.in**

**DEPARTMENT OF SCHOOL OF COMPUTING**

## BONAFIDE CERTIFICATE

This is to certify that this Professional Training Report is the bonafide work of **KOMMANA BHANU SIVA KUMAR** (**Reg. No. 37110352**) who underwent the professional training in **“ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING”** under our supervision from DEC 2018 to JAN 2019

**Internal Guide**

**Dr.KALAI ARASI**

**Head of the Department**

**Dr. S. VIGNESHWARI, M.E., Ph.D.**,



**Submitted for Viva voce Examination held on .**

**Internal Examiner External Examiner**

## DECLARATION

I, **KOMMANA BHANU SIVA KUMAR** (**Reg. No: 37110352)** thereby declare that the Project Report entitled **“CAR RACE GAME”** done by me under The guidance of **Dr.KALAI ARASI**  **Dept of CSE** at **PROTREX TECHNOLOGY**  is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

**DATE:**

**PLACE: CHENNAI SIGNATURE OF THE CANDIDATE**

## ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr.T.SASIKALA M.E.,Ph.D.,Dean,** **School of Computer Science**  and **Dr. S. VIGNESHWARI, M.E., Ph.D., Head of the Department, Dept. of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of **Computer Science and Engineering** who were helpful in many ways for the completion of the project.



## ABSTRACT

The project mainly focuses on developing a Dodge racing car game, using a process based upon pygame development; an evolutionary development method. It comprises implementation of game using python programming language.Talking about the gameplay, all the playing methods are too simple all you have to do is just dodge the car from the other cars. Here, the user has to dodge the car from the other cars in order to gain score points. The player has to be as quick as possible because the more you score the point, faster will be the gameplay. Whenever the player touches other cars, crashes or get the car to the end of the display screen, the game is over which shutdowns the program directly. The main objective of this game is to score points as much as possible. A simple GUI is provided for the easy gameplay. The gameplay design is so simple that user won’t find it difficult to use and navigate.

## TABLE OF CONTENTS

**CHAPTER NO. TITLE PAGE NO.**

**LIST OF FIGURES**

**LIST OF ABBREVVIATIONS**

1. **INTRODUCTION**
   1. ABOUT THE PROJECT
   2. PROJECT SCOPE
   3. PYGAME OBJECTIVES

## ALGORITHMS AND METHODS

## OVERVIEW

## FLOWCHART

## PYTHON FEATURES

## 2.3.1 PYGAME OVERVIEW

## 2.3.2 NUMPY

## 2.3.3 TEXTBLOB

1. **SYSTEM IMPLEMENTATION**

3.1 SOFTWARE REQUIREMENT

3.2 PYCHARM INSTALLATION

**4 CODE**

4.1 DISPLAY AND BACKGROUND COLOUR

4.2 BACKGROUND IMAGES

4.3 MOVING THE IMAGE

4.4 OBSTACLES

**5 CONCLUSION AND FUTURE WORK**

5.1 CONCLUSION

5.2 FUTURE WORK

## REFERENCES

## APPENDIX

* SOURCE CODE
* SCREENSHOTS

## 

## LIST OF FIGURES

**FIGURE NO TITLE PAGE NO**

2.2 Flowchart **3**

3.2.1 Downloading pycharm **5**

3.2.2 Selecting next **6**

3.2.3 Choosing path **6**

3.2.4 Selecting 32-bit 0r 64-bit **7**

3.2.5 Choosing Jetbrains **7**

3.2.6 Waiting for installation **8**

3.2.7 Finishing installation **8**

3.2.8 Completing installation **9**

3.2.9 Selecting buttons **9**

3.2.10 Choose “Create New **10**

Project”

4.1 Basic colours

4.2 Loading image

## 

## LIST OF ABBREVIATIONS

API Application Program Interface

GUI Graphical User Interface

IDE Integrated Development Environment

NLP Natural Language Processing

NUMPY Numerical Python

VCS Version Control System

## CHAPTER 1

## INTRODUCTION

### 1.1 ABOUT THE PROJECT

**Introduction:** Developing software applications is a time-consuming process, and with time-consuming processes come high costs. During the last years, several software development methodologies, often known as pygame software development, have become widely used by software developers to address this issue. Many different development methodologies can be more or less good, depending of the task and application type. One of the software development methodologies is the evolutionary software method, which, as the name hints, takes on an evolutionary approach to the problem, and allows the project to evolve through different stages of the project.It show how well this evolutionary approach worked on our project where we choose to develop a computer game and in the world of software development there lots of improvement in the area of Architectural and analytical design and principles. The philosophies and implementation details are changing as the people guiding the development of the application. In this fantastic and yet sometimes complex world of software development there are some tried and true analytical patterns and software development guidelines employed by most architects. Also your design must have an ability to turn towards innovation instead of lending itself to common practices. PYTHON is one such area where analysts must lean on their creative side. The DODGE RACING CAR GAME is a great project. The basic building aim is to provide an interesting game using PYTHON. DODGE RACING CAR GAME is a Browser based game that is designed to play. This finds comfort for their real time work while having quick access within this application. Development leads to enjoyment in players as well as developer.

### 1.2 PROJECT SCOPE

The purpose of making this application is to provide an easiness of accessing a car game designed using python. The user will find the best and simple way to play that is developed in one application that too in a simple way. The invention satisfies the foregoing needs and avoids the drawbacks. Python features a [dynamic](https://en.wikipedia.org/wiki/Dynamic_type) [type](https://en.wikipedia.org/wiki/Dynamic_type) system and automatic [memory](https://en.wikipedia.org/wiki/Memory_management) [management](https://en.wikipedia.org/wiki/Memory_management). It supports multiple [programming](https://en.wikipedia.org/wiki/Programming_paradigm) [paradigms,](https://en.wikipedia.org/wiki/Programming_paradigm) including object-oriented imperative, functional and procedural. It also had a comprehensive standard library.

### 1.3 PYGAME OBJECTIVES

PyGame is a good set of libraries for game development. Also python as a language is very comfortable in writing games especially for beginners. Also the object oriented approach is very handy in game development.

The objective of the game it’s win more points as possible until the car hits the another car and touches the boundaries.The speed can be increased by controlling with direction arrows.

The pygame library is an open-source module for the Python programming language specifically intended to help you make games and other multimedia applications. Built on top of the highly portable SDL (Simple DirectMedia Layer) development library, pygame can run across many platforms and operating systems.

## CHAPTER 2

## ALGORITHMS AND METHODS

### 2.1 OVERVIEW OF PYTHON

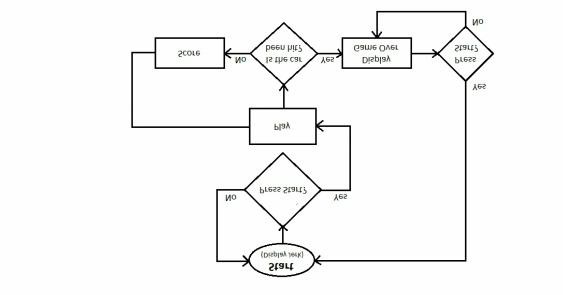
Python is a high-level, interpreted interactive and object-oriented

scripting language.Python is designed to be highly readable.It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

* **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
* **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
* **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.
* **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it.This is similar to PERL and PHP.

### 2.2 FLOWCHART

In fig 2.2, Initially the race gets started by pressing any key in the keyboard, the display turns to play mode and the score gets counted. If the car hits to any other or when it touches the boundaries then the screen turns to “game over”, otherwise the score will be recorded, the screen returns to the start page.



***Fig:2.2 Flow chart of car race***

### 2.3 PYTHON FEATURES

Python is an [interpreter](https://en.wikipedia.org/wiki/Interpreted_language), [high-level](https://en.wikipedia.org/wiki/High-level_programming_language), [general-purpose](https://en.wikipedia.org/wiki/General-purpose_programming_language) [programming](https://en.wikipedia.org/wiki/General-purpose_programming_language) language. Created by [Guido](https://en.wikipedia.org/wiki/Guido_van_Rossum) [van](https://en.wikipedia.org/wiki/Guido_van_Rossum) [Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) and first released in 1991, Python has a design philosophy that emphasizes [code](https://en.wikipedia.org/wiki/Code_readability) [readability](https://en.wikipedia.org/wiki/Code_readability), notably using [significant](https://en.wikipedia.org/wiki/Significant_whitespace) whitespace. It provides constructs that enable clear programming on both small and large scales. Van Rossum led the language community until stepping down as leader in July 2018.

Python interpreters are available for many [operating](https://en.wikipedia.org/wiki/Operating_system) [systems.](https://en.wikipedia.org/wiki/Operating_system) [C](https://en.wikipedia.org/wiki/CPython) [Python](https://en.wikipedia.org/wiki/CPython), the [reference](https://en.wikipedia.org/wiki/Reference_implementation) [implementation](https://en.wikipedia.org/wiki/Reference_implementation) of Python, is [open](https://en.wikipedia.org/wiki/Open-source_software) [source](https://en.wikipedia.org/wiki/Open-source_software) software and has a community-based development model, as do nearly all of Python's other implementations. Python and C Python are managed by the non-profit [Python](https://en.wikipedia.org/wiki/Python_Software_Foundation) Software Foundation.

#### Simple

Python is a simple language. Reading a good Python program feels almost like reading English, although very strict English! This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the languages.

#### Free and Open Source

Python is an example of a FLOSS (Free/LibrÃ© and Open Source

Software). In simple terms, you can freely distribute copies of this software, read it's source code, make changes to it, use pieces of it in new free programs, and that you know you can do these things. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and is constantly improved by a community who just want to see a better Python.

#### High-level Language

When you write programs in Python, you never need to bother about the low-level details such as managing the memory used by your program, etc.

#### Portable

Due to its open-source nature, Python has been ported (i.e. changed to make it work on) to many platforms. All your Python programs can work on any of these platforms without requiring any changes at all if you are careful enough to avoid any system-dependent features.

#### Interpreted

A program written in a compiled language like C or C++ is converted from the source language i.e. C or C++ into a language that is spoken by your computer (binary code i.e. 0s and 1s) using a compiler with various flags and options. When you run the program, the linker/loader software copies the program from hard disk to memory and starts running it.

Python, on the other hand, does not need compilation to binary. You just run the program directly from the source code. Internally, Python converts the source code into an intermediate form called byte codes and then translates this into the native language of your computer and then runs it. All this, actually, makes using Python much easier since you don't have to worry about compiling the program, making sure that the proper libraries are linked and loaded, etc, etc.

#### Object Oriented

Python supports procedure-oriented programming as well as objectoriented programming. In procedure-oriented languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In object-oriented languages, the program is built around objects which combine data and functionality. Python has a very powerful but simplistic way of doing OOP, especially when compared to big languages like C++ or Java.

### 2.3.1 Pygame overview

Pygame is a set of Python modules designed for writing games. It was started by Pete Shiners in 2000. It adds functionality onto the SDL library. It is portable, free, and released under the LGPL. It is supported on many operating systems including Linux, Windows, OS X, and BSD.

### 2.3.2 NUMPY

Numpy is the fundamental package for scientific computing with Python. It contains among other things a powerful N-dimensional array object sophisticated (broadcasting) functions tools for integrating C/C++ and Fortran code useful linear algebra, Fourier transform, and random number capabilities.

### 2.3.3 TEXTBLOB

TEXTBLOB is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.

## CHAPTER 3

## 

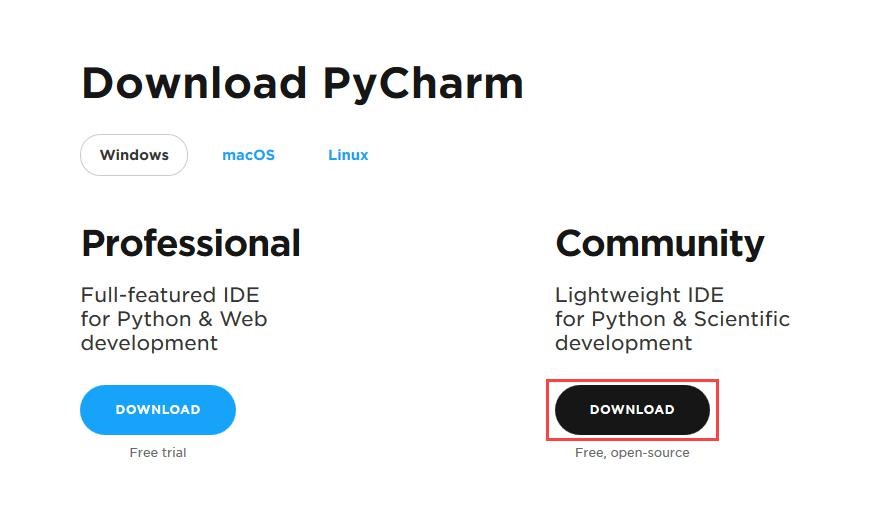
## SYSTEM IMPLEMENTATION

### 3.1 SOFTWARE REQUIREMENT

PyCharm is an [integrated](https://en.wikipedia.org/wiki/Integrated_development_environment) [development](https://en.wikipedia.org/wiki/Integrated_development_environment) [environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) used in computer Programming specifically for the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) language. It is developed by the Czech company [Jet](https://en.wikipedia.org/wiki/JetBrains) [Brains](https://en.wikipedia.org/wiki/JetBrains)[.[6]](https://en.wikipedia.org/wiki/PyCharm) It provides code analysis, a graphical debugger, an integrated unit tester, integration with [version](https://en.wikipedia.org/wiki/Revision_control) [control](https://en.wikipedia.org/wiki/Revision_control) [systems](https://en.wikipedia.org/wiki/Revision_control) (VCSes), and supports web development with [Django](https://en.wikipedia.org/wiki/Django_(web_framework)) as well as Data Science with Anaconda. PyCharm is cross-platform with Windows, mac and Linux versions. The Community Edition is released under the Apache License and there is also Professional Edit with extra features – released under a proprietary license.

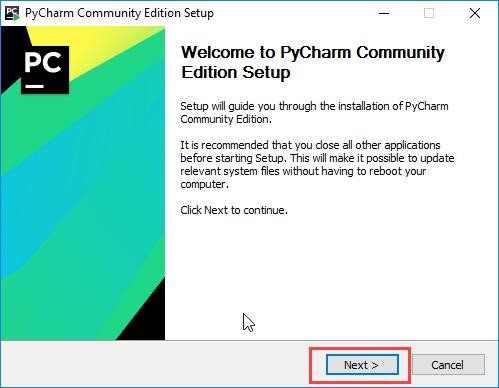
### 3.2 PYCHARM INSTALLATION

**Step 1:** To download PyCharm visit the website https://www.jetbrains.com/pycharm/download/ and Click the "DOWNLOAD" link under the Community edition.



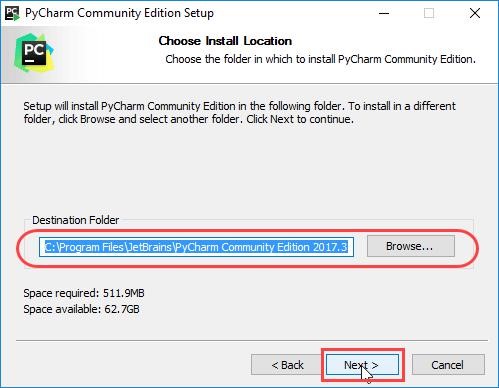
***Fig:3.2.1 Downloading pycharm***

**Step 2:** Once the download is complete, run the exe to install PyCharm. The setup wizard should have started. Click “Next”.



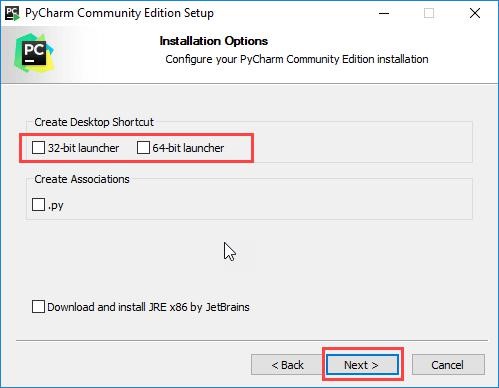
***Fig:3.2.2 Selecting next***

**Step 3:** On the next screen, Change the installation path if required. Click “Next”.



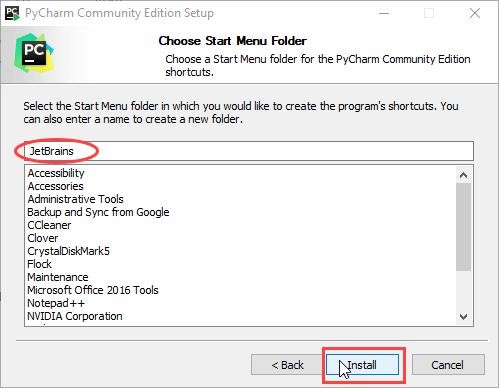
***Fig:3.2.3 Choosing path***

**Step 4:**On the next screen, you can create a desktop shortcut if you want. select either ‘32-bit launcher’ or ’64-bit launcher’. You can also select ‘Create Association' checkbox to make Python programs open in Pycharm by default. After that, click on “Next”.



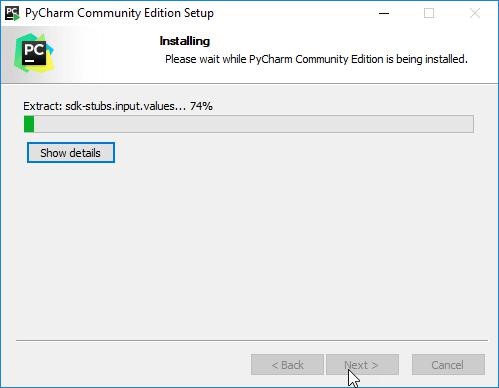
***Fig:3.2.4 selecting 32-bit or 64-bit***

**Step 5**: Choose the start menu folder. Keep selected Jet Brains and click on install

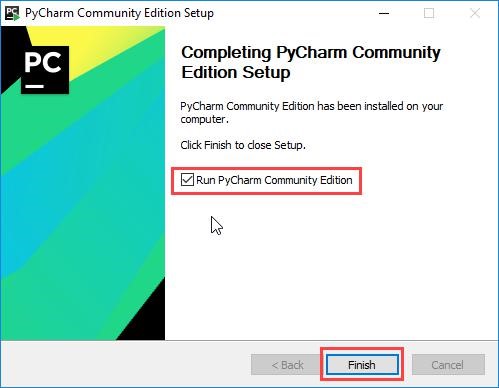


***Fig:3.2.5 Choosing jetbrains***

**Step 6:** Wait for the installation to finish.

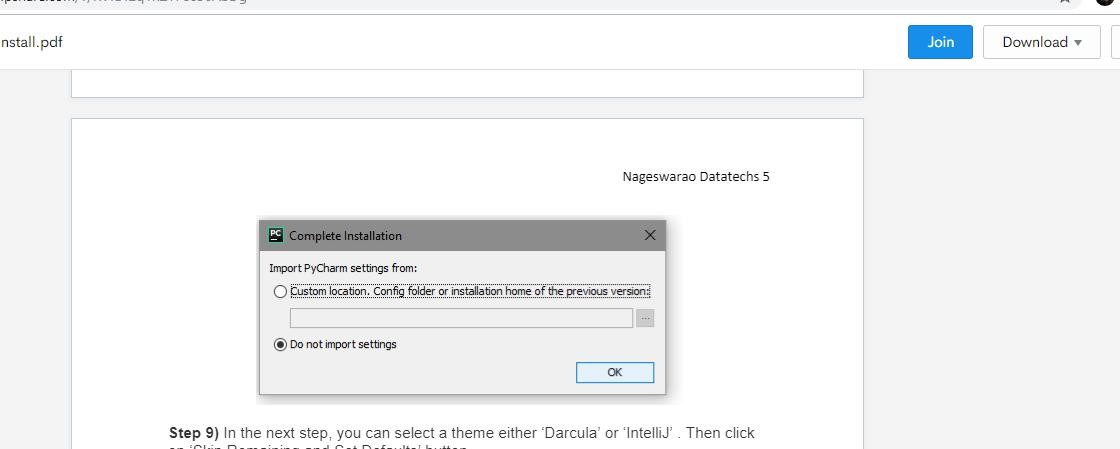


***Fig:3.2.6 Waiting for installation***

**Step 7:** Once installation finished, you should receive a message screen that Pycharm is installed. If you want to go ahead and run it, click the “run Pycharm Community Edition” box first and click “Finish”. 

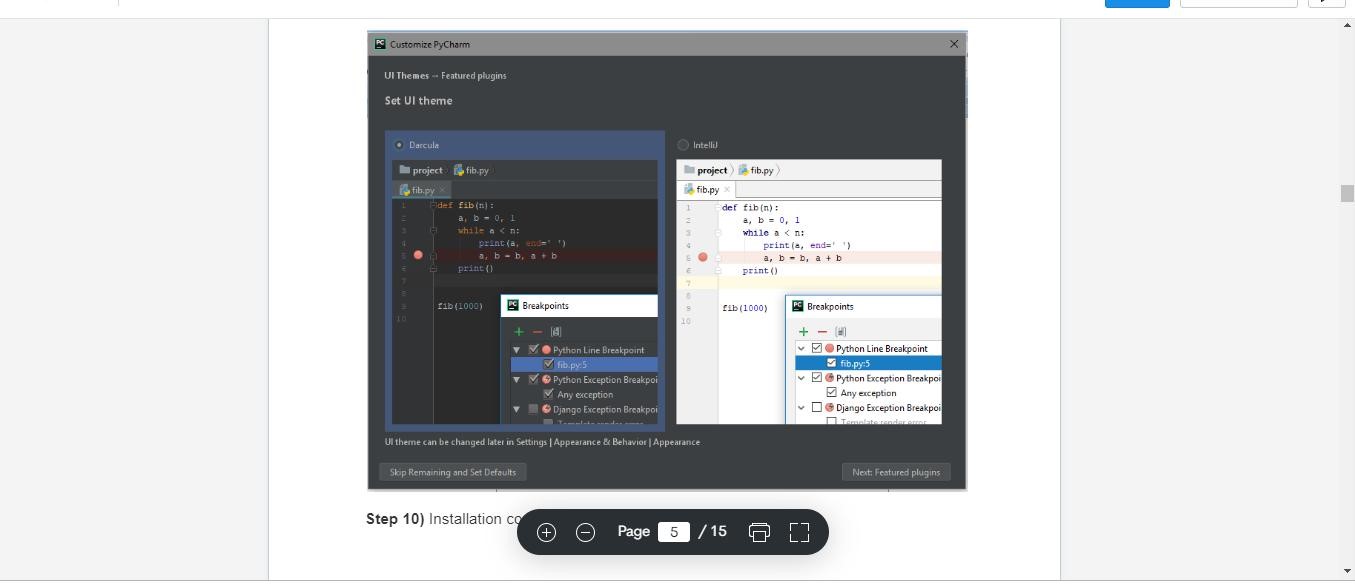
***Fig:3.2.7 Finishing installation.***

**Step 8:**After you click on ”Finish”, the Following screen will appear. Select, Do not import settings” and click on, ”OK”.



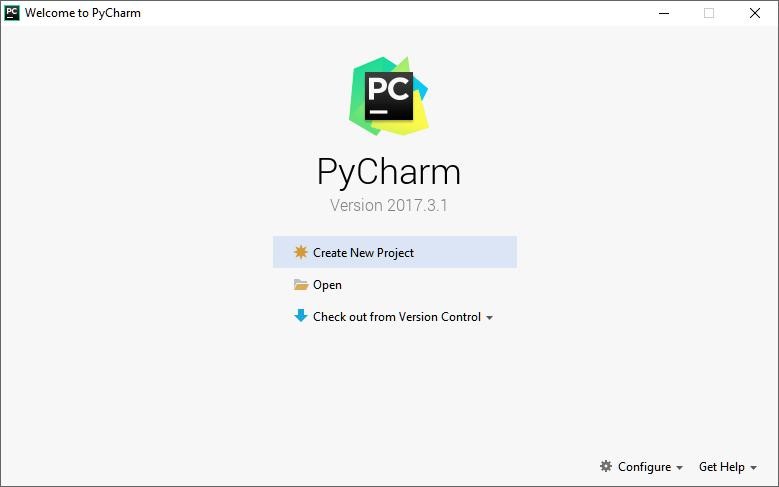
***Fig :3.2.8 Selecting complete installation.***

**Step 9:** In the next step, you can select a theme either “ Dracula” or “IntelliJ”. Then click on Skip Remaining and Set Defaults “button”.



***Fig:3.2.9 Selecting buttons*** .

**Step 10:** Installation completes with the following screen.



***Fig:3.2.10 Choose ”Create New Project”.***

## CHAPTER 4 CODE

### 4.1 DISPLAY AND BACKGROUND COLOUR

Now,We should create a window with width and height.And then we add background colour and text colour using “rgb colour”

A screenshot of a cell phone

Description automatically generated

**Fig:4.1 basic colours**

These are the basic colours .By using these colours we can set our background or text colours by using those rbg colour codes.

Now , to set our window we give the height and width of the window .Then a window will be created and the background colour is set to that window.

windowwidth=800

windowheight=600

Then a window is created with width 800 and height 600.

### 4.2 BACKGROUND IMAGES

To the window created with the background ,we add to add the sides and the images to that window .So,to add images we have to download those images and then copy those images to the pycharm project page.

And ,then first we have to add the sides to that window and then add the player car image and the add the obstacles.

First download all the images needed and copy all the images to

c drive and goto user and goto user\_name and then to pycharm projects and then to the game and paste the images along with the code.

In pycharm you can see the images on left side of the page that you pasted and those images can be used in the code as shown in the below figure.

A screenshot of a computer screen

Description automatically generated

**Fig:4.2 loading images**

### 4.3 MOVING THE IMAGE

We use the keys to move the object from one position to another position like front,back,left,right.. we use “up” key or “a” key to go front and “down” key or “s” to move down and “left” key or “a” key to go left and “right” key or ”d” key to go right.

if event.type == KEYDOWN:  
 if event.key == ord('z'):  
 reverseCheat = True  
 if event.key == ord('x'):  
 slowCheat = True  
 if event.key == K\_LEFT or event.key == ord('a'):  
 moveRight = False  
 moveLeft = True  
 if event.key == K\_RIGHT or event.key == ord('d'):  
 moveLeft = False  
 moveRight = True  
 if event.key == K\_UP or event.key == ord('w'):  
 moveDown = False  
 moveUp = True  
 if event.key == K\_DOWN or event.key == ord('s'):  
 moveUp = False  
 moveDown = True  
   
  
if event.type == KEYUP:  
 if event.key == ord('z'):  
 reverseCheat = False  
 score = 0  
 if event.key == ord('x'):  
 slowCheat = False  
 score = 0  
 if event.key == K\_ESCAPE:  
 terminate()  
  
 if event.key == K\_LEFT or event.key == ord('a'):  
 moveLeft = False  
 if event.key == K\_RIGHT or event.key == ord('d'):  
 moveRight = False  
 if event.key == K\_UP or event.key == ord('w'):  
 moveUp = False  
 if event.key == K\_DOWN or event.key == ord('s'):  
 moveDown = False

### 4.4 OBSTACLES

We already added the obstacles and ther images .Now,we use the window size and the images pixel size to find the obstacles crash.From the total window we take x and y axis and from ther pixels and then we take the image pixel size and according to that we compare and then we declare the obstacles strike and ther add speed and add rate can be managed.

if obstaclesAddCounter == ADDNEWobstaclesRATE:  
 obstaclesAddCounter = 0  
 obstaclesSize = 30

newobstacles ={

'rect': pygame.Rect(random.randint(140, 485),obstaclesSize, 23, 47)

'speed': random.randint(obstaclesMINSPEED, obstaclesMAXSPEED),  
 'surface': pygame.transform.scale(random.choice(sample), (23, 47))

}

obstacles.append(newobstacles)

sideLeft = {

'rect': pygame.Rect(0, 0, 126, 600),  
 'speed': random.randint(obstaclesMINSPEED, obstaclesMAXSPEED),  
 'surface': pygame.transform.scale(wallLeft, (126, 599)),

}  
 obstacles.append(sideLeft)

sideRight = {'rect': pygame.Rect(497, 0, 303, 600),  
 'speed': random.randint(obstaclesMINSPEED, obstaclesMAXSPEED),  
 'surface': pygame.transform.scale(wallRight, (303, 599)),  
 }

obstacles.append(sideRight)  
  
if moveLeft and playerRect.left > 0:  
 playerRect.move\_ip(-1 \* PLAYERMOVERATE, 0)

if moveRight and playerRect.right < WINDOWWIDTH:  
 playerRect.move\_ip(PLAYERMOVERATE, 0)

if moveUp and playerRect.top > 0:  
 playerRect.move\_ip(0, -1 \* PLAYERMOVERATE)

if moveDown and playerRect.bottom < WINDOWHEIGHT:  
 playerRect.move\_ip(0, PLAYERMOVERATE)

## CHAPTER 5 CONCLUSION AND FUTURE WORK

### 5.1 CONCLUSION

Simple Car Dodge Game project is written in Python. The project file contains asset files, python scripts and image files. The gameplay Graphics is good enough and the controls are too simple for the users. Talking about the gameplay, all the playing methods are too simple all you have to do is just dodge the car from the other cars. Here, the user has to dodge the car from the other cars in order to gain score points. The player has to be as quick as possible because the more you score the point, faster will be the gameplay. Whenever the player touches other cars, crashes or get the car to the end of the display screen, the game is over which shutdowns the program directly. The main objective of this game is to score points as much as possible. A simple GUI is provided for the easy gameplay. The gameplay design is so simple that user won’t find it difficult to use and navigate.

Different images and sound files are used in the development of this minigame project. In order to run the project, you must have installed [Python](https://www.python.org/downloads/release/python-365/), on your PC.

### 5.2 FUTURE WORK

Being more obsessed to the games, started developing initial game car race. So that it would be an initiative to my future work for developing a new game. The gaming industry has an array of career choices to offer. It makes up for a large segment on the employment front too. The most common myth about game industry jobs is the lack of options. However game careers are not limited to game design and game development.

## REFERENCES

[1] [https://www.w3schools.com/python/default.asp](https://www.w3schools.com/python/default.asp%20%20) [2]<https://www.youtube.com/watch?v=SZUNUB6nz3g>

1. Head first Python: A Brain-Friendly Guide, by paul Barry.
2. Python crash course: A hands-on, project-Based introduction to programming,

By Eric Matthes.

[5]Python Programming: A introduction to computer Science, by john Zelle.

## APPENDIX

### a.SOURCE CODE

import pygame, random, sys, os, time

from pygame.locals import \*

WINDOWWIDTH = 800

WINDOWHEIGHT = 600

TEXTCOLOR = (0, 255, 255)

BACKGROUNDCOLOR = (119,118,110)

FPS = 40

obstaclesMINSIZE = 10

obstaclesMAXSIZE = 15

obstaclesMINSPEED = 8

obstaclesMAXSPEED = 8

ADDNEWobstaclesRATE = 8

PLAYERMOVERATE = 7

count = 3

topScore = 0

def terminate():

pygame.quit()

sys.exit()

def waitForPlayerToPressKey():

while True:

for event in pygame.event.get()

if event.type == QUIT:

terminate()

if event.type == KEYDOWN:

if event.key == K\_ESCAPE:

terminate()

return

def playerHasHitobstacles(playerRect, obstacles):

for obs in obstacles:

if playerRect.colliderect(obs['rect']):

return True

return False

def drawText(text, font, surface, x, y):

textobj = font.render(text, 1, TEXTCOLOR)

textrect = textobj.get\_rect()

textrect.topleft = (x, y)

surface.blit(textobj, textrect)

pygame.init()

mainClock = pygame.time.Clock()

windowSurface = pygame.display.set\_mode((WINDOWWIDTH, WINDOWHEIGHT))

pygame.display.set\_caption('car race')

pygame.mouse.set\_visible(False)

playerImage = pygame.image.load('car1.png')

car3 = pygame.image.load('car3.png')

car4 = pygame.image.load('car4.png')

playerRect = playerImage.get\_rect()

obstacleImage = pygame.image.load('car2.png')

sample = [car3, car4, obstacleImage]

wallLeft=pygame.image.load('left.png')

wallRight = pygame.image.load('right1.png')

font = pygame.font.SysFont(None, 42)

drawText('PRESS ANY KEY TO START THE GAME!', font, windowSurface, (WINDOWWIDTH / 3) - 137, (WINDOWHEIGHT / 3) + 80)

pygame.display.update()

waitForPlayerToPressKey()

zero = 0

while (count > 0):

obstacles = []

score = 0

playerRect.topleft = (WINDOWWIDTH / 2, WINDOWHEIGHT - 50)

moveLeft = moveRight = moveUp = moveDown = False

reverseCheat = slowCheat = False

obstaclesAddCounter = 0

while True:

score += 1

for event in pygame.event.get ():

if event. type == QUIT:

terminate ()

if event. type == KEYDOWN:

if event.key == ord('z'):

reverseCheat = True

if event.key == ord('x'):

slowCheat = True

if event.key == K\_LEFT or event.key == ord('a'):

moveRight = False

moveLeft = True

if event.key == K\_RIGHT or event.key == ord('d'):

moveLeft = False

moveRight = True

if event.key == K\_UP or event.key == ord('w'):

moveDown = False

moveUp = True

if event.key == K\_DOWN or event.key == ord('s'):

moveUp = False

moveDown = True

if event.type == KEYUP:

if event.key == ord('z'):

reverseCheat = False

score = 0

if event.key == ord('x'):

slowCheat = False

score = 0

if event.key == K\_ESCAPE:

terminate()

if event.key == K\_LEFT or event.key == ord('a'):

moveLeft = False

if event.key == K\_RIGHT or event.key == ord('d'):

moveRight = False

if event.key == K\_UP or event.key == ord('w'):

moveUp = False

if event.key == K\_DOWN or event.key == ord('s'):

moveDown = False

if event.key == ord('p'):

resume()

if not reverseCheat and not slowCheat:

obstaclesAddCounter += 1

if obstaclesAddCounter == ADDNEWobstaclesRATE:

obstaclesAddCounter = 0

obstaclesSize = 30

newobstacles = {

'rect': pygame.Rect(random.randint(140, 485), 0 - obstaclesSize, 23, 47),

'speed': random.randint(obstaclesMINSPEED, obstaclesMAXSPEED),

'surface': pygame.transform.scale(random.choice(sample), (23, 47)),

}

obstacles.append(newobstacles)

sideLeft = {

'rect': pygame.Rect(0, 0, 126, 600),

'speed': random.randint(obstaclesMINSPEED, obstaclesMAXSPEED),

'surface': pygame.transform.scale(wallLeft, (126, 599)),

}

obstacles.append(sideLeft)

sideRight = {

'rect': pygame.Rect(497, 0, 303, 600),

'speed': random.randint(obstaclesMINSPEED, obstaclesMAXSPEED),

'surface': pygame.transform.scale(wallRight, (303, 599)),

}

obstacles.append(sideRight)

if moveLeft and playerRect.left > 0:

playerRect.move\_ip(-1 \* PLAYERMOVERATE, 0)

if moveRight and playerRect.right < WINDOWWIDTH:

playerRect.move\_ip(PLAYERMOVERATE, 0)

if moveUp and playerRect.top > 0:

playerRect.move\_ip(0, -1 \* PLAYERMOVERATE)

if moveDown and playerRect.bottom < WINDOWHEIGHT:

playerRect.move\_ip(0, PLAYERMOVERATE)

for obs in obstacles:

if not reverseCheat and not slowCheat:

obs['rect'].move\_ip(0, obs['speed'])

elif reverseCheat:

obs['rect'].move\_ip(0, -5)

elif slowCheat:

obs['rect'].move\_ip(0, 1)

for obs in obstacles[:]:

if obs['rect'].top > WINDOWHEIGHT:

obstacles.remove(obs)

font = pygame.font.SysFont(None, 38)

windowSurface.fill(BACKGROUNDCOLOR)

drawText('Score: %s' % (score), font, windowSurface, 128, 0)

drawText('Top Score: %s' % (topScore), font, windowSurface, 128, 21)

drawText('Rest Life: %s' % (count), font, windowSurface, 128, 41)

windowSurface.blit(playerImage, playerRect)

for obs in obstacles:

windowSurface.blit(obs['surface'], obs['rect'])

pygame.display.update()

if playerHasHitobstacles(playerRect, obstacles):

if score > topScore:

topScore = score

break

mainClock.tick(FPS)

count = count - 1

time.sleep(1)

font = pygame.font.SysFont(None, 52)

if (count == 0):

drawText('Game Over', font, windowSurface, (WINDOWWIDTH / 3) + 40, (WINDOWHEIGHT / 3) + 70)

drawText('Press any key to play again.', font, windowSurface, (WINDOWWIDTH / 3) - 110, (WINDOWHEIGHT / 3) + 95)

pygame.display.update()

time.sleep(2)

waitForPlayerToPressKey()

count = 3

### b.SCREENSHOTS

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated