PROJECT REPORT ON

PYTHON

**UNIVERSITY OF ENGINEERING AND MANAGEMENT, KOLKATA**

**Prepared by:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Stream | Semester | Year |
| Debjit  Chattopadhyay | **CSE** | **2nd** | **1st** |
| Aritra  Ghosh | **CSE** | **2nd** | **1st** |
| Purnava  Roy | **CSE** | **2nd** | **1st** |

SIMPLE CARD GAME

IN

PYTHON

**Prepared for:**

Python Department

Under the guidance of:

**Deepa Ma’am.**

**Table of Contents**

Acknowledgement

Abstract

Project Scope

Requirement Specification

Screenshots ..

Code ..

Future Scope of Improvements .

References…………………………………………………………………………………………………………………………………………

Project Certificate

**Acknowledgement**

I take this opportunity to express my profound gratitude and deep regards to my faculty for her exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by her time to time shall carry me a long way in the journey of life on which I am about to embark.

I am obliged to my project team members for the valuable information provided by them in their respective fields. I am grateful for their cooperation during the period of my assignment.

**Abstract**

A simple card game that compares the numeric values of the cards to determine the winner.

The player and the bot will be given 3 random cards each. The player can play a card by using the left click button of the mouse on that particular card.

The bot will also play a card from its hand. The card which has the greater value wins the hand. The score is updated as per win or loss. The game continues till the deck is empty.

The primary project goal is:

* To deliver a user friendly experience of a card game using python graphics, concept of lists, if-else statements and basic concepts of OOP. It is easy to use and understand.

**Requirement Specification**

Domain Description

We used Python idle in our project and imported tkinter class for implementing a GUI Interface.

Problem Definition

The basic idea was to develop a simple user interface and create a card game using the basics of programming. It also reveals the greater prospects that Python can provide in the field of game development using python graphics.

Functional Requirements

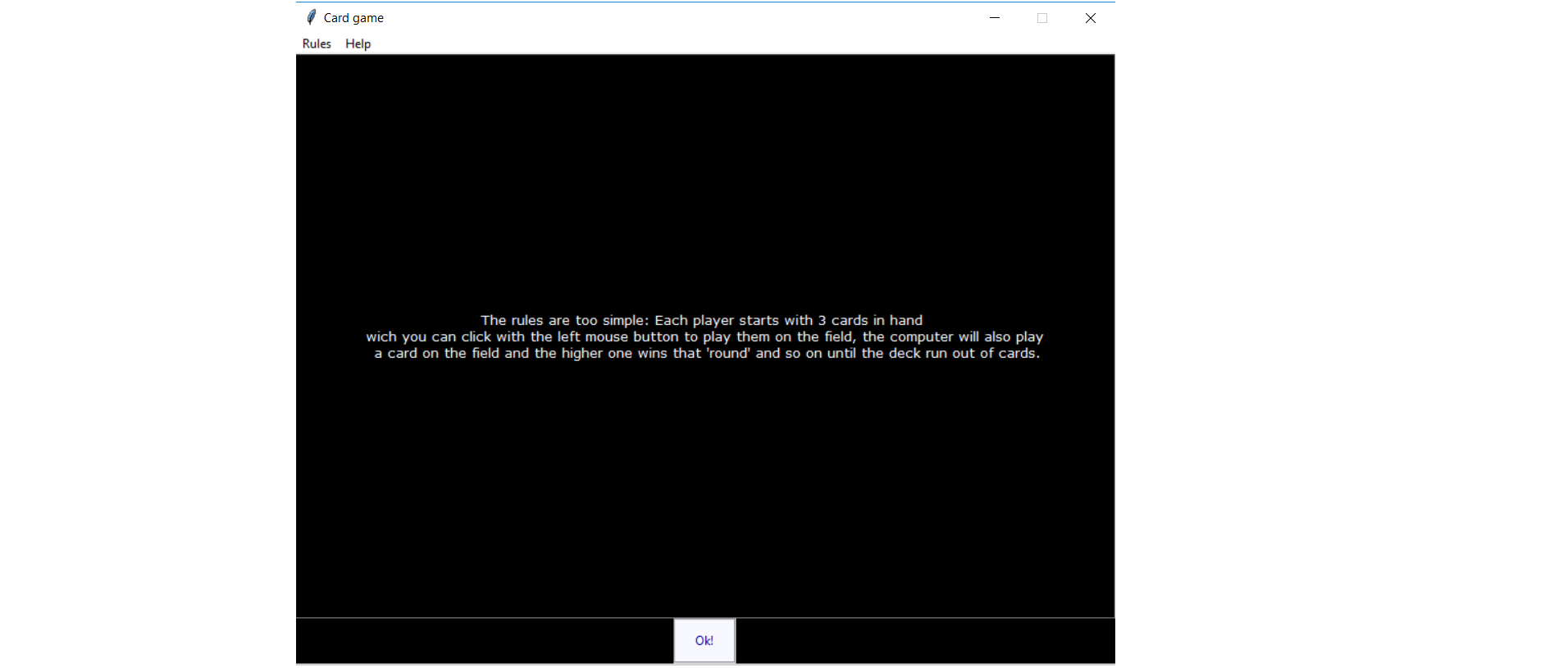
With the most recent developments in game engine software that provides and developes more realistic and high end games, Python has a considerable amount of contribution to develop the UI in 1st generation games like Space Invaders, Minesweeper, etc. it was a humble attempt by us to design a simple UI and create a card game with tkinter.



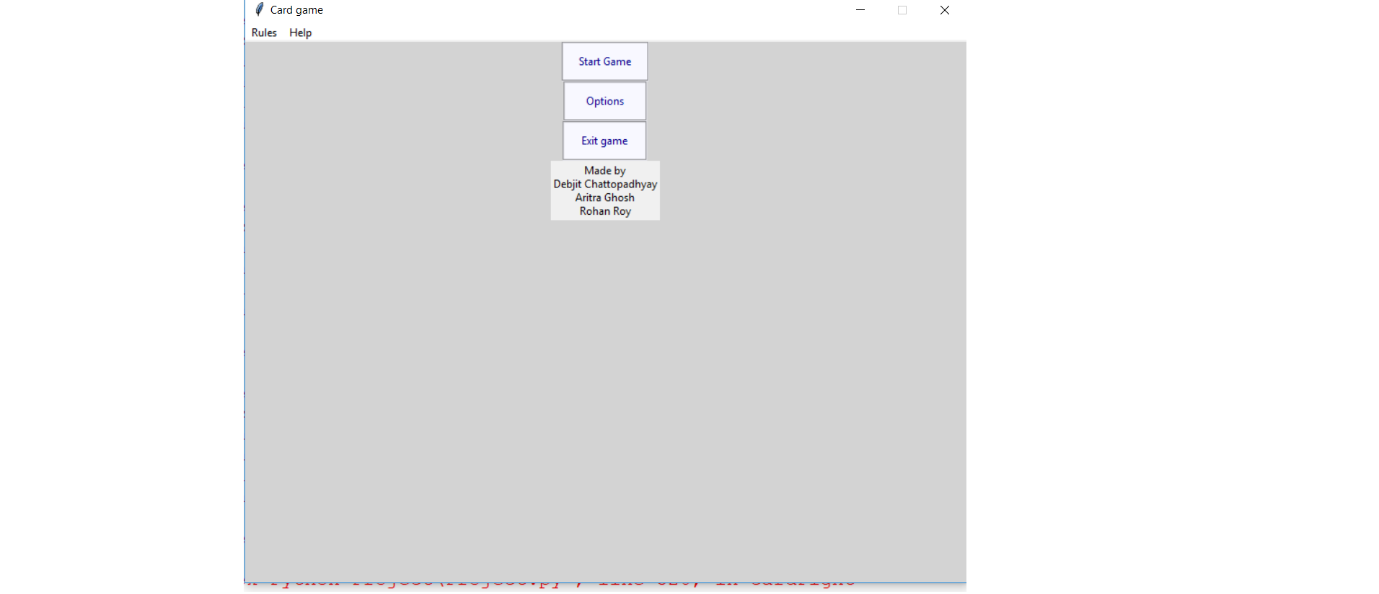
This user-friendly software will provide the user a good experience while playing with cards and understanding the huge potential that Python has in game development.

**SCREENSHOTS**

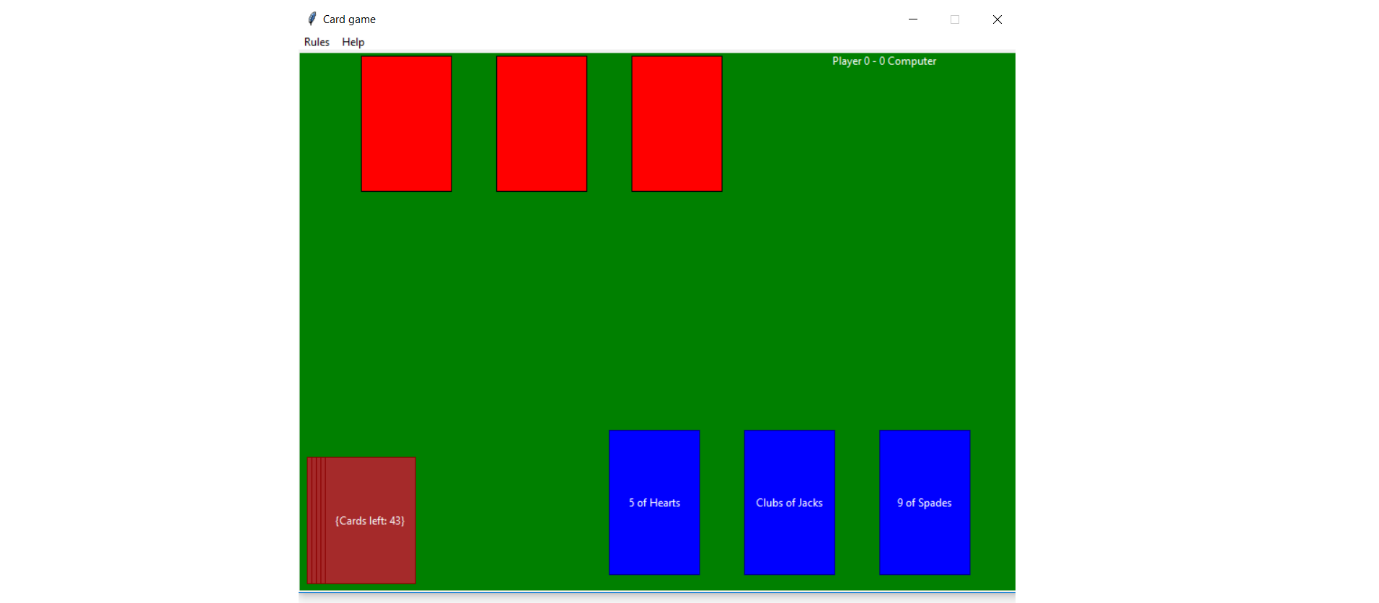
**Rules Screen**

****

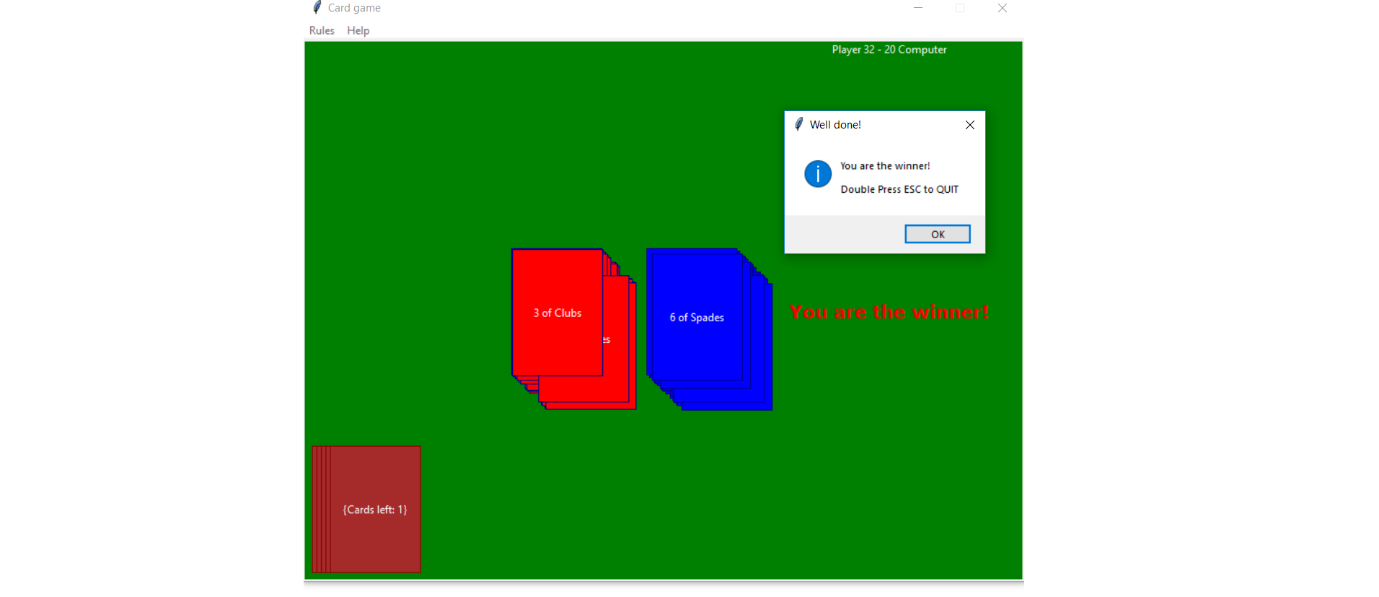
**Welcome Screen**



**Game Screen**

****

**Result Screen**

****

**Code :-**

**Cards.py**

deck = []

def makeDeck():

suits = ["Spades","Diamonds","Hearts","Clubs"]

for suit in suits:

for value in range(2,14):

if value == 11:

a = "Jacks"

deck.append(suit +" of "+a)

elif value == 12:

a = "Queens"

deck.append(suit +" of "+a)

elif value == 13:

a = "Kings"

deck.append(suit +" of "+a)

else:

deck.append(str(value) +" of "+ suit )

deck.append("Ace")

**vars.py**

BACKGROUND = "lightgrey"

distCard = 100

playerDraw = True

botDraw = False

#Other cards have value of 1

cardValues = {'2 of Spades':5, 'Jack of Spades ':2, 'Queen of Spades':3, 'King of Spades':4,

'2 of Diamonds':6,'Jack of Diamonds':3, 'Queen of Diamonds':4, 'King of Diamonds':5,

'2 of Hearts':7, 'Jack of Hearts':4, 'Queen of Hearts':5, 'King of Hearts':6,

'2 of Clubs':8,'Jack of Clubs':5, 'Queen of Clubs':6, 'King of Clubs':7,

'Ace':15}

**Project.py(Main Code)**

"""

Card game made only with Tkinter in Python, by Debjit Chattopadhyay\nAritra Ghosh\nPurnava Roy July,2018;

"""

from tkinter import \*

import tkinter.messagebox

import vars as var

import cards as Deck

import sys, random

class Window(object):

def \_\_init\_\_(self):

self.root = Tk()

self.root.geometry(("800x600"))

self.root.title("Card game")

try:

self.root.wm\_iconbitmap("icon.ico")

except:

pass

self.root.bind("<Escape>" , sys.exit)

# Creating a menu

mainMenu = Menu(self.root)

self.root.config(menu = mainMenu)

subMenu = Menu(mainMenu, tearoff = 0)

mainMenu.add\_cascade(label = "Rules", command = self.gameRules)

mainMenu.add\_cascade(label = "Help", menu = subMenu)

subMenu.add\_command(label = "Known bugs", command = self.knownBugs)

subMenu.add\_command(label = "Report a bug", command = self.reportBug)

subMenu.add\_separator()

subMenu.add\_command(label = "How to play?", command = self.howToPlay)

self.root.resizable(False,False)

self.root['bg'] = var.BACKGROUND

#Starts the game rules screen...

self.gameRulesScreen()

# PRE LOADED FRAMES

self.frameMenuBar = Frame(self.root, bg = "blue")

self.frameGame = Frame(self.root)

#canvas for the game

self.gameScreen = Canvas(self.frameGame,width=800,height=600,bg="green")

# Field list (Field is where the cards are played)

self.field = ["0","0"] # Each for player, and this is the last crad played.

self.root.mainloop()

#""" Menu functions """#

def knownBugs(self):

newWindow = Toplevel()

newWindow.wm\_title("Known bugs")

text = Label(newWindow, text="""These are the known bugs:

> Clicking on the name of the cards to play them don't work.

> If play all the cards too fast the bot can't keep up and will not draw.""")

text.pack(side="top", fill="both", expand=True, padx=100, pady=100)

def reportBug(self):

newWindow = Toplevel()

newWindow.wm\_title("Report a bug")

text = Label(newWindow, text = "Reporting bugs helps the developers to improve the game.")

self.reportEntry = Entry(newWindow)

ok = Button(newWindow, text = "Send", command = self.getReport)

text.pack()

self.reportEntry.pack()

ok.pack()

def getReport(self):

#Idealy i wanted the player to send this to a online database not to the game folder;

print(">",self.reportEntry.get())

file = open("report.txt","a")

file.write("\n" + self.reportEntry.get())

file.close()

def howToPlay(self):

#TODO

print("Work in progress...")

def gameRules(self):

#TODO

print("Work in progress...")

def mainMenu(self):

self.frameRules.destroy()

self.frameTitle = Frame(self.root, bg = var.BACKGROUND)

self.frameTitle.pack()

self.start\_but = Button(self.frameTitle,text="Start Game",

pady= 10,

command = self.play,

padx = 15,fg='darkblue', bg='ghostwhite', relief=GROOVE)

self.options\_but = Button(self.frameTitle,

text="Options",

command = self.options,

pady= 10,padx = 21,fg='darkblue', bg='ghostwhite', relief=GROOVE)

self.exit\_but = Button(self.frameTitle,

text="Exit game",

command = self.exit,

pady= 10,padx = 17,fg='darkblue', bg='ghostwhite', relief=GROOVE)

self.developer = Label(self.frameTitle,

text = "Made by\nDebjit Chattopadhyay\nAritra Ghosh\nRohan Roy")

self.start\_but.pack()

self.options\_but.pack()

self.exit\_but.pack()

self.developer.pack(side = BOTTOM)

#""" Game functions """#

def status(self):

#Display cards on deck

self.cards\_on\_deck = len(Deck.deck)

self.gameScreen.create\_text((80,520),text = ("Cards left: %i"%self.cards\_on\_deck,),fill="white",tag="gameStats")

self.gameScreen.create\_text((650,10),text = ("Player %i - %i Computer"%(self.player\_points,self.bot\_points)),fill="white",tag="gameStats")

def update(self):

if self.gameRun:

self.gameScreen.delete("gameStats","pts")

self.status()

self.botAI()

if len(Deck.deck) <= 0:

self.winner()

self.win = "OVER"

self.root.after(200, self.update) #increasing this might cause a bug, but increases FPS. (10 is fine)

else:

pass

def play(self):

#This will start the game and some variables.

#Vars

self.bot\_hand = []

self.player\_hand= []

self.player\_points = 0

self.bot\_points = 0

self.gameRun = True

#

#delete menus

self.frameTitle.destroy()

self.frameGame.pack()

self.gameScreen.pack()

#creating scenario

self.status()

self.startingCards()

self.update()

#binds

self.gameScreen.tag\_bind('card0',"<Button-1>",self.placeCard)

self.gameScreen.tag\_bind('card1',"<Button-1>",self.placeCard)

self.gameScreen.tag\_bind('card2',"<Button-1>",self.placeCard)

def drawCard(self):

if len(Deck.deck) <= 3 :

return

self.cardDrawn = random.choice(Deck.deck)

if var.playerDraw:

self.b = self.player\_hand[0] == 0

self.c = self.player\_hand[1] == 0

self.d = self.player\_hand[2] == 0

if self.b:

#Card get removed from deck and its added to the hand.

self.player\_hand[0] = self.cardDrawn

Deck.deck.remove(self.cardDrawn)

#Drawing card on screen with text for position 0.

self.gameScreen.create\_rectangle((345,420),(445,580),fill="blue",tag="card0",outline="darkblue")

self.cardText = self.gameScreen.create\_text((395,500),text = ("%s"%self.cardDrawn),fill="white",tag="cardText0")

return

elif self.c:

self.player\_hand[1] = self.cardDrawn

Deck.deck.remove(self.cardDrawn)

#Drawing card on screen with text for position 1.

self.gameScreen.create\_rectangle((345+150,420),(445+150,580),

fill="blue",tag="card1",outline="darkblue")

self.cardText = self.gameScreen.create\_text((395+150,500),

text = ("%s"%self.cardDrawn),fill="white",tag="cardText1")

return

elif self.d:

self.player\_hand[2] = self.cardDrawn

Deck.deck.remove(self.cardDrawn)

#Drawing card on screen with text for position 2.

self.gameScreen.create\_rectangle((345+300,420),(445+300,580),

fill="blue",tag="card2",outline="darkblue")

self.cardText = self.gameScreen.create\_text((395+300,500),

text = ("%s"%self.cardDrawn),fill="white",tag="cardText2")

return

def botDraw(self):

if len(Deck.deck) <= 0 :

return

else:

self.cardDrawn0 = random.choice(Deck.deck)

Deck.deck.remove(self.cardDrawn0)

if len(Deck.deck) <= 0 :

return

self.cardDrawn1 = random.choice(Deck.deck)

Deck.deck.remove(self.cardDrawn1)

if len(Deck.deck) <= 0 :

return

self.cardDrawn2 = random.choice(Deck.deck)

Deck.deck.remove(self.cardDrawn2)

self.bot\_hand[0] = self.cardDrawn0

self.gameScreen.create\_rectangle((20+50,5),(120+50,155),

fill="red",tag="cardBOT0",outline="darkred")

self.bot\_hand[1] = self.cardDrawn1

self.gameScreen.create\_rectangle((20+200,5),(120+200,5+150),

fill="red",tag="cardBOT1",outline="darkred")

self.bot\_hand[2] = self.cardDrawn2

self.gameScreen.create\_rectangle((20+350,5),(120+350,5+150),

fill="red",tag="cardBOT2",outline="darkred")

self.bot\_hand.reverse()

def botAI(self):

#TODO

#The simplest AI possible :)

if self.bot\_hand[2] != 0:

self.botPlay = self.bot\_hand[2]

return

elif self.bot\_hand[1] != 0:

self.botPlay = self.bot\_hand[1]

return

elif self.bot\_hand[0] != 0:

self.botPlay = self.bot\_hand[0]

return

else:

self.botDraw()

def placeCard(self,event):

#Player's move

if (event.y >= 420 and event.y <= 580):

#To make it more dynamic

desv = random.randint(-20,20)

#card left

if (event.x >= 345 and event.x <= 445):

self.gameScreen.delete("card0","cardText0")

self.card = self.gameScreen.create\_rectangle((400+desv,250+desv),(500+desv,390+desv),

fill = "blue",

tag="card\_on\_field",

outline="darkblue")

self.cardText = self.gameScreen.create\_text((450+desv,320+desv),

fill="white",

tag="cardText\_on\_field",

text= self.player\_hand[0])

#remove from status and...

#place on the field , index 0 for player

#then to compare with index 1 wich is the bot played card.

self.field[0] = self.player\_hand[0]

self.player\_hand[0] = 0

elif (event.x >= 495 and event.x <= 595):

#card middle

self.gameScreen.delete("card1","cardText1")

self.card = self.gameScreen.create\_rectangle((400+desv,250+desv),(500+desv,390+desv),

fill = "blue",

tag="card\_on\_field",

outline="darkblue")

self.cardText = self.gameScreen.create\_text((450+desv,320+desv),

fill="white",

tag="cardText\_on\_field",

text= self.player\_hand[1])

#remove from status

self.field[0] = self.player\_hand[1]

self.player\_hand[1] = 0

elif (event.x >= 645 and event.x <= 745):

#card right

self.gameScreen.delete("card2","cardText2")

self.cardField = self.gameScreen.create\_rectangle((400+desv,250+desv),(500+desv,390+desv),

fill = "blue",

tag="card\_on\_field",

outline="darkblue")

self.cardTextField = self.gameScreen.create\_text((450+desv,320+desv),

fill="white",

tag="cardText2",

text= self.player\_hand[2])

#remove from status

self.field[0] = self.player\_hand[2]

self.player\_hand[2] = 0

else:

print("Player: not in that x")

#Draws 3 cards automaticly if there is no cards in your hand

if self.player\_hand == [0,0,0]:

for i in range(3):

self.drawCard()

#Bot's moves when player moves first.

if self.field[0] != 0:

desv = random.randint(-20,20)

#card left

if self.botPlay == self.bot\_hand[0]:

self.gameScreen.delete("cardBOT0")

self.card = self.gameScreen.create\_rectangle((250+desv,250+desv),(350+desv,390+desv),

fill = "red",

tag="card\_on\_field",

outline="darkblue")

self.cardText = self.gameScreen.create\_text((300+desv,320+desv),

fill="white",

tag="cardText\_on\_field",

text= self.bot\_hand[0])

#remove from status

self.field[1] = self.bot\_hand[0]

self.bot\_hand[0] = 0

elif self.botPlay == self.bot\_hand[1]:

#card middle

self.gameScreen.delete("cardBOT1")

self.card = self.gameScreen.create\_rectangle((250+desv,250+desv),(350+desv,390+desv),

fill = "red",

tag="card\_on\_field",

outline="darkblue")

self.cardText = self.gameScreen.create\_text((300+desv,320+desv),

fill="white",

tag="cardText\_on\_field",

text= self.bot\_hand[1])

#remove from status

self.field[1] = self.bot\_hand[1]

self.bot\_hand[1] = 0

elif self.botPlay == self.bot\_hand[2]:

#card right

self.gameScreen.delete("cardBOT2")

self.cardField = self.gameScreen.create\_rectangle((250+desv,250+desv),(350+desv,390+desv),

fill = "red",

tag="card\_on\_field",

outline="darkblue")

self.cardTextField = self.gameScreen.create\_text((300+desv,320+desv),

fill="white",

tag="cardText2",

text= self.bot\_hand[2])

#remove from status

self.field[1] = self.bot\_hand[2]

self.bot\_hand[2] = 0

self.cardFight()

def startingCards(self):

a = 295

#Draw player hand

for i in range(3):

deckCard = random.choice(Deck.deck)

print(deckCard)

#Drawing cards on hand

self.card = self.gameScreen.create\_rectangle((50 + a,120 + 300),(150 + a,150 + 430),

fill="blue",

tag="card%i"%i,

outline="darkblue")

self.cardText = self.gameScreen.create\_text((100 + a,200 + 300),

text = ("%s"%deckCard),

tag = "cardText%i"%i,

fill="white")

#Sending to status

self.player\_hand.append(deckCard)

Deck.deck.remove(deckCard)

a += 150

#Draw bot hand

a = 50

for i in range(3):

deckCard = random.choice(Deck.deck)

print(deckCard)

#

self.card1 = self.gameScreen.create\_rectangle((20 + a,5 + 0),(120 + a,5 + 150),

fill="red",

tag="cardBOT%i"%i,

)

#

a += 150

self.bot\_hand.append(deckCard)

Deck.deck.remove(deckCard)

self.bot\_hand.reverse()

#Deck on screen

for i in range(5):

self.gameScreen.create\_rectangle((10+5\*i,450),(110+ 5\*i,590),fill="brown",tag="deck",outline="darkred")

def cardFight(self):

#Comparing and searching for the value on the string

self.player\_card = self.field[0]

self.bot\_card = self.field[1]

#

if True:

if self.player\_card in var.cardValues:

cardValue\_player = var.cardValues.get(self.player\_card)

else:

for i in self.player\_card:

if i.isnumeric():

num = i

num = int(num)

if num <= 7 and num >= 0:

cardValue\_player = 1

elif num > 7 and num <= 10:

cardValue\_player = 2

#

if True:

if self.bot\_card in var.cardValues:

cardValue\_bot = var.cardValues.get(self.bot\_card)

else:

for i in self.bot\_card:

if i.isnumeric():

num = i

num = int(num)

if num >= 0 and num <= 7 :

cardValue\_bot = 1

elif num > 7 and num <= 10:

cardValue\_bot = 2

#Add points

if cardValue\_player > cardValue\_bot:

self.player\_points += cardValue\_player

self.gameScreen.create\_text((700,300),text=("+" + str(cardValue\_player)),fill="blue",tag="pts",font=("Verdana","15","bold"))

elif cardValue\_player == cardValue\_bot:

self.player\_points += cardValue\_player

self.bot\_points += cardValue\_bot

self.gameScreen.create\_text((700,300), text=("+" + str(cardValue\_player)),fill="blue",tag="pts",font=("Verdana","15","bold"))

self.gameScreen.create\_text((700,400), text=("+" + str(cardValue\_bot)),fill="red",tag="pts",font=("Verdana","15","bold"))

else:

self.bot\_points += cardValue\_bot

self.gameScreen.create\_text((700,400), text=("+" + str(cardValue\_bot)), fill="red", tag="pts", font=("Verdana","15","bold"))

def gameRulesScreen(self):

self.frameRules = Frame(self.root,bg = "black")

self.frameRules.pack()

self.ok\_but = Button(self.frameRules,

text="Ok!",

command = self.mainMenu,

pady= 10,padx = 17,fg='darkblue', bg='ghostwhite', relief=GROOVE)

self.rules = Label(self.frameRules,bd = 1, relief = SUNKEN,

pady = 250, padx = 250,

text="The rules are too simple: Each player starts with 3 cards in hand "

"\n which you can click with the left mouse button to play them on the field, the computer will also play "

"\n a card on the field and the higher one wins that 'round' and so on until the deck run out of cards."

,fg="ghostwhite",bg="black",font=("Verdana",10))

self.rules.pack()

self.ok\_but.pack()

def winner(self):

self.win = self.player\_points > self.bot\_points

if self.win:

self.textAlert("You are the winner!",650,300)

tkinter.messagebox.showinfo("Well done!","You are the winner!\n\nDouble Press ESC to QUIT")

game\_run = False

if self.win == "OVER":

print("game end...")

elif self.win == False:

self.textAlert("Bot is the winner!",100,300)

tkinter.messagebox.showinfo("You still played well....","Bot is the winner!\n\nDouble Press ESC to QUIT")

game\_run = False

#""" game settings functions """#

def options(self):

self.frameTitle.destroy()

self.frameOptions = Frame(self.root,bg = var.BACKGROUND)

self.frameOptions.pack()

self.bg\_var = False

self.bg\_but = Button(self.frameOptions, text="Nocturne Mode", command = self.nocturneMode)

self.mainmenu\_but = Button(self.frameOptions,

text="Return to main menu",

command = self.changeMenu, fg="white",bg="darkgrey",font=("Verdana","15","bold"))

self.bg\_but.pack()

self.mainmenu\_but.pack(side=BOTTOM)

def changeMenu(self):

self.frameOptions.destroy()

self.mainMenu()

def textAlert(self,text,x,y,color="red"):

self.text = self.gameScreen.create\_text((x,y),text=text,fill=color,font=("Verdana","15","bold"))

def nocturneMode(self):

self.bg\_var = not self.bg\_var

if self.bg\_var:

self.root['bg'] = 'black'

self.frameOptions['bg'] = 'black'

self.gameScreen['bg'] = 'darkgreen'

var.BACKGROUND = "black"

else:

self.root['bg'] = 'lightgrey'

self.frameOptions['bg'] = 'lightgrey'

self.gameScreen['bg'] = 'green'

var.BACKGROUND = "lightgrey"

def exit(self):

quit()

#""" """#

if \_\_name\_\_ == "\_\_main\_\_":

Deck.makeDeck()

Window()

**Future Scope of Improvements:**

* The project is a humble attempt to create a computerized playing bot versus a real user. Decision making skills and and light touch of AI is also portrayed by the behavior of the bot.
* Presently, there are a few bugs and errors in the game, which can be surely resolved by future developments.

**References**

* https://stackoverflow.com
* [https://docs.python.org](https://docs.python.org/)/
* https://github.com