Report On

Guess the Number Game

Submitted in partial fulfillment of the requirements of the Mini project in

Semester IV of Second Year Computer Engineering

by

Prachi Raut (Roll No. 23)

Sayyed Abdus Sami (Roll No.32)

Mentor

Prof. Sneha Mhatre

**Vidyavardhini's College of Engineering & Technology**

**Department of Computer Engineering**



**(A.Y. 2023-24)**

**Vidyavardhini's College of Engineering & Technology**

**Department of Computer Engineering**

**CERTIFICATE**

This is to certify that the Project entitled **“Guess the Number Game”** is a bonafide work of **Prachi Raut (Roll No. 23), Sayyed Abdus Sami (Roll No. 32)** submitted to the University of Mumbai in partial fulfillment of the requirement for the Course project in semester IV of Second Year **“Computer Engineering”.**

|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Prof. Sneha Mhatre  Mentor |  |
| --- | --- | --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Dr Megha Trivedi  Head of Department |  | Dr. H.V. Vankudre  Principal |

**Abstract**

The "Guess Number Game" is an interactive game developed using Python's Tkinter library. The game allows players to guess a randomly generated number within a specified range, with three difficulty levels: Easy (1-20), Medium (1-50), and Hard (1-100). Players are presented with a series of graphical user interfaces (GUIs) that guide them through the game, from entering their name to selecting a difficulty level and making guesses. Each level comes with a different number of hearts/lives, indicating the number of attempts a player has. The game provides real-time feedback through messages and updates the heart/life count upon incorrect guesses. After completing or losing the game, players are given the option to play again or exit.

**Content**

Introduction 1

Block Diagram 2

Module and Description 3

Brief Description of Technology 4

Code 5

Result and Conclusion 6

Reference 7

**Introduction**

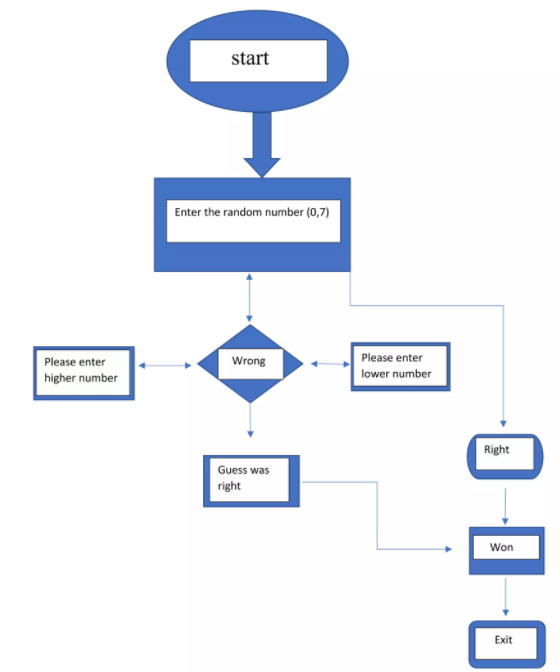
Welcome to the "Guess Number Game", an engaging and entertaining game designed to challenge your guessing skills! This game is an interactive experience where players attempt to guess a randomly selected number within a predefined range. Whether you're a beginner or an experienced player, there's a level for everyone - Easy, Medium, and Hard.

To get started, simply enter your name on the first page and select a difficulty level that suits you. As you progress through the game, you'll notice a heart/life count that indicates the number of attempts you have left. Make sure to use your guesses wisely!

The game offers real-time feedback, guiding you with hints to help you narrow down your guesses. If you guess the number correctly, you win! But be careful, if you run out of hearts/lives, you lose. Don't worry though, you can always play again for another chance to beat your score.

With its user-friendly interface and dynamic gameplay, the "Guess Number Game" promises hours of fun and excitement. So, are you ready to take on the challenge and test your guessing skills? Let's play!

**Block Diagram**



**Module and Description**

This program is a simple "Guess the Number" game using the Tkinter library in Python. It consists of multiple pages or stages where the user can interact with the game. Here's a brief overview of each page and description:

**First Page (Entry Page):**

User is greeted with a welcome message and prompted to enter their name.

Upon entering their name and clicking "Submit", the user proceeds to the second page.

**Second Page (Level Selection):**

After entering the name, the user is greeted with a personalized message.

The user can select from three difficulty levels: Easy, Medium, or Hard.

Clicking on a difficulty level takes the user to the third page where the game starts.

**Third Page (Game Page):**

The user is shown the range of numbers they should guess, based on the selected difficulty.

A heart emoji and the number of remaining lives (hearts) are displayed.

The user can input their guess in a text box and submit it.

Depending on the guess, the user is given feedback through message boxes:

If the guess is correct, a congratulatory message is displayed, and the user proceeds to the forth page.

If the guess is wrong, a message indicates whether the correct number is higher or lower than the guess. The user also loses a heart.

If all hearts are lost, a message informs the user that they lost and reveals the correct number. The user then proceeds to the forth page.

**Forth Page (Play Again or Exit):**

After winning or losing the game, the user is asked if they want to play again.

Clicking "Yes" restarts the game by going back to the first page.

Clicking "No" exits the game.

Additionally, the game keeps track of the user's wins and losses by writing them to a file named "log.txt".

**Brief Description of Technology**

Brief explanation of the technologies and concepts used:

**Tkinter:**

Tkinter is Python's standard GUI (Graphical User Interface) library.

It provides a set of tools and widgets to create desktop applications with graphical interfaces.

You've used it to create windows, labels, buttons, and text input areas.

**Random:**

The random module is used for generating random numbers.

You've used random.randint() to generate random numbers for the game.

**Messagebox:**

tkinter.messagebox is used to display message boxes for notifications like errors or information messages.

It provides methods like showerror, showinfo to display different types of messages to the user.

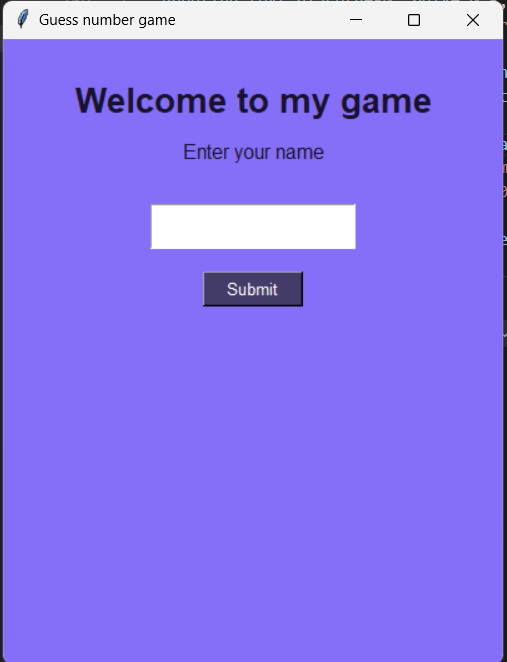
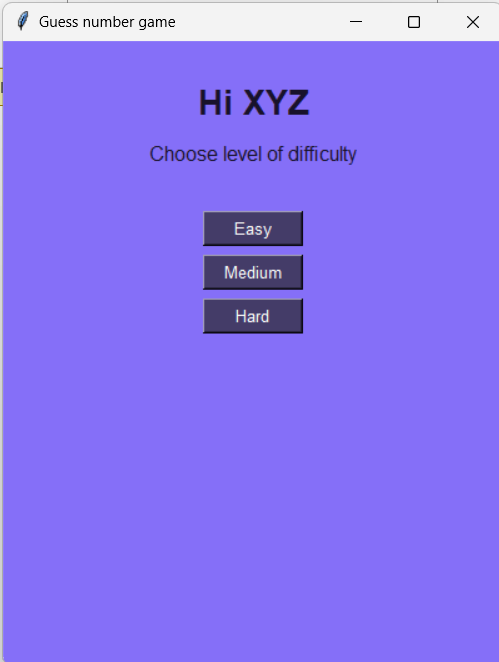
**File I/O:**

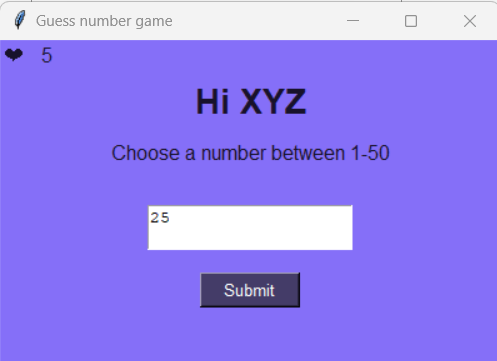
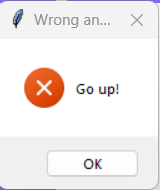
The write\_to\_file() function uses Python's built-in file handling to write the player's name and game result to a text file named "log.txt".

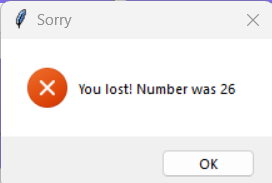
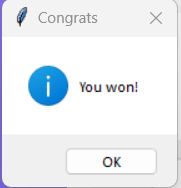
**Code**

| from tkinter import \*  import random  from tkinter import messagebox  def set\_number(level):  hearts = 0  number = 0  heading\_text = ""  if (level == "easy"):  heading\_text = " 1-20"  hearts = 4  number = random.randint(1,20)  elif (level == "medium"):  heading\_text = " 1-50"  hearts = 5  number = random.randint(1,50)  else:  heading\_text = " 1-100"  hearts = 7  number = random.randint(1,100)  return heading\_text,hearts, number  def forth\_page():  def yes\_action():  root.destroy()  first\_page()  def no\_action():  root.destroy()  exit()  root = Tk()  root.title("Guess number game")  root.geometry("400x500")  root['background']='#856ff8'  root.minsize(400,500)  root.maxsize(400,500)  greeting\_text = Label(root, text="Do to want to play again?", font= ('Helvetica 20 bold'), bg= '#856ff8', fg='#18122B')  greeting\_text.pack()  greeting\_text.place(y=50, relx= 0.5, anchor='center')  btn\_yes = Button(root, text = 'Yes', command= lambda: yes\_action(),  font= ('Helvetica 10 normal'), bg="#443C68", fg="#ffffff")  btn\_yes.pack()  btn\_yes.place(y = 120, width=80, relx= 0.5, anchor='center')  btn\_no = Button(root, text = 'No', command= lambda: no\_action(),  font= ('Helvetica 10 normal'), bg="#cc3549")  btn\_no.pack()  btn\_no.place(y = 155, width=80, relx= 0.5, anchor='center')  root.mainloop()  def third\_page(level, player\_name):  HEARTS = []  heading\_text,hearts, number = set\_number(level)  HEARTS.append(hearts)  def write\_to\_file(condition):  file = open("log.txt","a")  text = player\_name + " " + condition + '\n'  file.write(text)  file.close()  def submit():  inp = inputtxt.get(1.0, "end-1c")  result = play\_game(inp)  if(HEARTS[0] <= 0):  messagebox.showerror("Sorry", "You lost! Number was "+ str(number))  write\_to\_file("lost")  root.destroy()  forth\_page()  if(result == 1):  messagebox.showinfo("Congrats", "You won!")  write\_to\_file("won")  root.destroy()  forth\_page()  elif(result == 2):  messagebox.showerror("Wrong answer", "Go down!")  else:  messagebox.showerror("Wrong answer", "Go up!")  def play\_game(inp):  if (int(inp) == number):  return 1  HEARTS[0] -= 1  hearts\_text['text'] = str(HEARTS[0]) # This line update lives of user after guess a number wrong  if(int(inp) > int(number)):  return 2  else: return 3  root = Tk()  root.title("Guess number game")  root.geometry("400x500")  root['background']='#856ff8'  root.minsize(400,500)  root.maxsize(400,500)  greeting\_text = "Hi " + player\_name  greeting\_text = Label(root, text=greeting\_text, font= ('Helvetica 20 bold'), bg= '#856ff8', fg='#18122B')  greeting\_text.pack()  greeting\_text.place(y=50, relx= 0.5, anchor='center') | text2 = Label(root, text='Choose a number between' + heading\_text, font= ('Helvetica 12 normal'),  bg= '#856ff8', fg='#18122B')  text2.pack()  text2.place(y=90, relx= 0.5, anchor='center')  emoji = Label(root, text="\u2764\ufe0f", font= ('Helvetica 12 normal'), bg= '#856ff8', fg='#18122B')  emoji.pack()  emoji.place(anchor='nw')  hearts\_text = Label(root, text=str(hearts), font= ('Helvetica 12 normal'), bg= '#856ff8', fg='#18122B')  hearts\_text.pack()  hearts\_text.place(x = 30, anchor='nw')  inputtxt = Text(root,height = 2,width = 20)  inputtxt.pack()  inputtxt.place(y = 150, relx= 0.5, anchor='center')  btn\_easy = Button(root, text = 'Submit', command= lambda: submit(),  font= ('Helvetica 10 normal'), bg="#443C68", fg="#ffffff")  btn\_easy.pack()  btn\_easy.place(y = 200, width=80, relx= 0.5, anchor='center')  root.mainloop()  def second\_page(palyer\_name):  def game\_window(level):  root.destroy()  third\_page(level, palyer\_name)  root = Tk()  root.title("Guess number game")  root.geometry("400x500")  root['background']='#856ff8'  root.minsize(400,500)  root.maxsize(400,500)  greeting\_text\_ = "Hi " + palyer\_name  greeting\_text = Label(root, text=greeting\_text\_, font= ('Helvetica 20 bold'), bg= '#856ff8', fg='#18122B')  greeting\_text.pack()  greeting\_text.place(y=50, relx= 0.5, anchor='center')  level\_text = Label(root, text='Choose level of difficulty', font= ('Helvetica 12 normal'), bg= '#856ff8', fg='#18122B')  level\_text.pack()  level\_text.place(y=90, relx= 0.5, anchor='center')  btn\_easy = Button(root, text = 'Easy',command=lambda: game\_window("easy"),  font= ('Helvetica 10 normal'), bg="#443C68", fg="#ffffff")  btn\_medium = Button(root, text = 'Medium',command=lambda: game\_window("medium"),  font= ('Helvetica 10 normal'), bg="#443C68", fg="#ffffff")  btn\_hard = Button(root, text = 'Hard',command=lambda: game\_window("hard"),  font= ('Helvetica 10 normal'), bg="#443C68", fg="#ffffff")  btn\_easy.pack()  btn\_easy.place(y = 150, width=80, relx= 0.5, anchor='center')  btn\_medium.pack()  btn\_medium.place(y = 185, width=80, relx= 0.5, anchor='center')  btn\_hard.pack()  btn\_hard.place(y = 220, width=80, relx= 0.5, anchor='center')  root.mainloop()  def first\_page():  def submit():  inp = inputtxt.get(1.0, "end-1c")  root.destroy()  second\_page(inp)  root = Tk()  root.title("Guess number game")  root.geometry("400x500")  root['background']='#856ff8'  root.minsize(400,500)  root.maxsize(400,500)  greeting\_text = Label(root, text='Welcome to my game', font= ('Helvetica 20 bold'), bg= '#856ff8', fg='#18122B')  greeting\_text.pack()  greeting\_text.place(y=50, relx= 0.5, anchor='center')  level\_text = Label(root, text='Enter your name', font= ('Helvetica 12 normal'), bg= '#856ff8', fg='#18122B')  level\_text.pack()  level\_text.place(y=90, relx= 0.5, anchor='center')  inputtxt = Text(root,height = 2,width = 20)  inputtxt.pack()  inputtxt.place(y = 150, relx= 0.5, anchor='center')  btn\_easy = Button(root, text = 'Submit',command=lambda: submit(),  font= ('Helvetica 10 normal'), bg="#443C68", fg="#ffffff")  btn\_easy.pack()  btn\_easy.place(y = 200, width=80, relx= 0.5, anchor='center')  root.mainloop()  first\_page() |
| --- | --- |

**Result**

** **

** **

** **

**Conclusion**

Guessing game worked correctly as it followed most of the objectives. It told you if your guess was too high or too low and if your guess was right it told you that also. It had a limit

between 1-100 so that it wasn't too hard for people to guess. As well as this the final,

completed guessing game didn't come up with error messages as it ran through the flow chart each time a number was entered which shows that it was correct. It was only a basic version and wasn't expanded but it worked fine.