# TechVidvan’s Number Guessing Game

#### **About Number Guessing Game:**

Guessing a number is a fun game that involves us to relate the clues the system gives us. This fun, yet provoking game is a simple way to keep our mind stimulated.

#### **Python Number Guessing Game Project:**

We will create a simple number guessing game using python and display the scores the player gets, if he/she wins. This project will serve as an introduction to functions and creating GUI using Tkinter.

#### **Project Prerequisites:**

The project makes use of tkinter for GUI, random module for generating random targets and clues. Since we use Tkinter which is a built-in GUI library in python, we check it by importing it

python

>>> import tkinter

If the library is installed, it will not show an error. If you face any error whilst importing Tkinter, refer to [Tkinter Installation](https://tkdocs.com/tutorial/install.html)

#### **Download Number Guessing Game code:**

You can download the source code for the Number Guessing Game in the given link: [Number Guessing Game](https://drive.google.com/file/d/1DeVDmBTsIMylnoRyZr3i8cJnYFmjBjl5/view?usp=sharing)

#### **Project File Structure:**

There are many GUI libraries supported by python such as PyQT5, Kivy, Pyside2 etc. Tkinter is widely used by many developers and is easy for beginners to practise.

Let’s have a look at the steps to build the project:

1. Importing modules: tkinter, random and declaring variables
2. Declaring clue functions
3. Declaring and defining guess and target number functions
4. Creating the user input interface

Now for the implementation in detail.

Feel free to play with the values and change your input methods.

##### **Importing necessary modules:**

#TechVidvan's Number Guessing Game

#Import modules

import random

from tkinter import \*

from tkinter import messagebox

#Declare and initialise variables

target = 0

score = 10

guess=0

**Code Explanation:**

* **import random:**  Random library generates a random number or a random subset in a string. We use randint function of random
* **from tkinter import \*:** To create the user interface, we import Tkinter. Tkinter contains provisions to build the interface with widgets such as buttons, input entry fields etc
* **from tkinter import messagebox:** To display the user with prompts, we use messagebox. Using this module, we can show warnings, errors, information etc to the user
* **Target, score, guess:** Declare the variables to set the target value and guess from the user in the forthcoming functions. Score is set to 10 which also implies the user gets 10 chances before he loses

##### **Declaring clue functions:**

#Define the clue functions

def add():

return "The sum of target and guess is " + str(guess+target)

def sub():

return "The difference of target and guess is " + str(target-guess)

def multiplication():

return "The product of target and guess is " + str(guess\*target)

def division():

return "The division of target by guess is " + str(target/guess)

def greater\_lesser():

if target < guess:

return 'Target is less than the guess'

elif target > guess:

return 'Target is greater than the guess'

**Code explanation:**

* **add():** Returns the sum of the guess and the target
* **sub():** Returns the difference of the guess and the target
* **multiplication():** Returns the product of the guess and the target
* **division():** Returns the division of target by guess
* **greater\_lesser():** Tells if the target is less than the guess and vice versa

##### **Declaring and defining guess and target number functions:**

#Create the random clue generator

def clues():

switcher = {

0: add(),

1: sub(),

2: multiplication(),

3: division(),

4: greater\_lesser()

}

return switcher.get(random.randint(0,4))

**Code explanation:**

* **clues():** Declare the function clues
* **switcher:** Define a dictionary switcher to simulate switch case where the values of the keys represent the clue functions
* **return switcher.get(random.randint(0,4)):** Take a random key from the switcher to generate a random clue using get() function

#Generate the target value

def generate\_target\_number():

global target

target = random.randint(1,10)

messagebox.showinfo(message="Random Number Generated; Start Guessing!! STARTING SCORE=10")

#Disable the random number button until game ends

random\_number\_button['state'] = DISABLED

#Activate the guessing button

guess\_button['state'] = NORMAL

**Code explanation:**

* **generate\_target\_number():** Declare the function generate\_target\_number
* **global target:** Make the variable global to preserve its value across other functions, ie not limiting to local scope (within the function)
* **target = random.randint(1,10):** Using randint, assign a random integer to target. The parameters are the lower and upper range, both inclusive
* **messagebox.showinfo():** To display the user with a prompt after generating the target value, we display a prompt using messagebox. Since this is an information, we use showinfo of messagebox with the message parameter set to a message.
* **random\_number\_button['state'], guess\_button['state']:** After getting a target, we disable the random number button so the user plays a fair game without changing the target. Enable guess button to allow the user to guess the number after getting a target. Disable turns off the button and Normal allows to click the button. The state of the button attribute is altered to achieve this.

#Guess and score by reading user input

def guess\_and\_score():

#Make variables global for access across functions

global score

global guess

try:

guess =0

#Read if user submitted an input

guess = int(guess\_entry.get())

except:

messagebox.showerror(message="Enter a number to guess and play")

return

#If target and guess are the same, print score and prompt to user

if guess == target:

messagebox.showinfo(message="Congratulations!!! You guessed the number correct. Your score is "+str(score))

#Enable random number button to play a new game and disable guessing button

random\_number\_button['state'] = NORMAL

guess\_button['state'] = DISABLED

return

#If the user runs out of guesses

elif score == 0:

messagebox.showwarning(message="Out of Guesses Buddy! Better luck next time );")

return

#Call the guessing functions to give the clues

else:

score -= 1

message=clues()

messagebox.showinfo(message=message)

**Code explanation:**

* **guess\_and\_score:** Declare the function guess\_and\_score
* **global score, global guess:** Make the variable global to preserve its value across other functions, ie not limiting to local scope (within the function)
* **guess =0,guess = int(guess\_entry.get()):** Initialise guess to 0 and read the user given input. We put this in a try…except block to raise an error if the user did not specify with a guess. Raise the error prompt using messagebox.showerror()
* **if guess == target:** Display the score to the player if he guesses it correct and disable the guess button. Enable the random number button to play another game
* **elif score == 0:** If the score is 0, it implies the user has run out of tries. Prompt the user with 0 score and show 0 moves to go
* **Else block:** The else block calls for clues. It calls clues functions which will call the functions created earlier. Display the clue using messagebox

##### **Creating the user input interface:**

#Create the user interface, specify the dimensions of the application window

window = Tk()

window.geometry("350x200")

window.title("TechVidvan's Number Guessing Game")

#Mention the title of the app

title\_label = Label(window, text="TechVidvan's Number Guessing Game\nGuess a number between 1 to 50", font=('Ubuntu Mono',12))

title\_label.pack()

#Generate random number

random\_number\_button = Button(window, text="Generate Random Number", command=generate\_target\_number)

random\_number\_button.pack()

#Read User input

guess\_label = Label(window, text="Enter your guess: ")

guess\_label.pack()

guess\_entry = Entry(window, width=3)

guess\_entry.pack()

#Start guessing

guess\_button = Button(window, text="Guess Me", command=guess\_and\_score, state=DISABLED)

guess\_button.pack()

#Exit and close the app

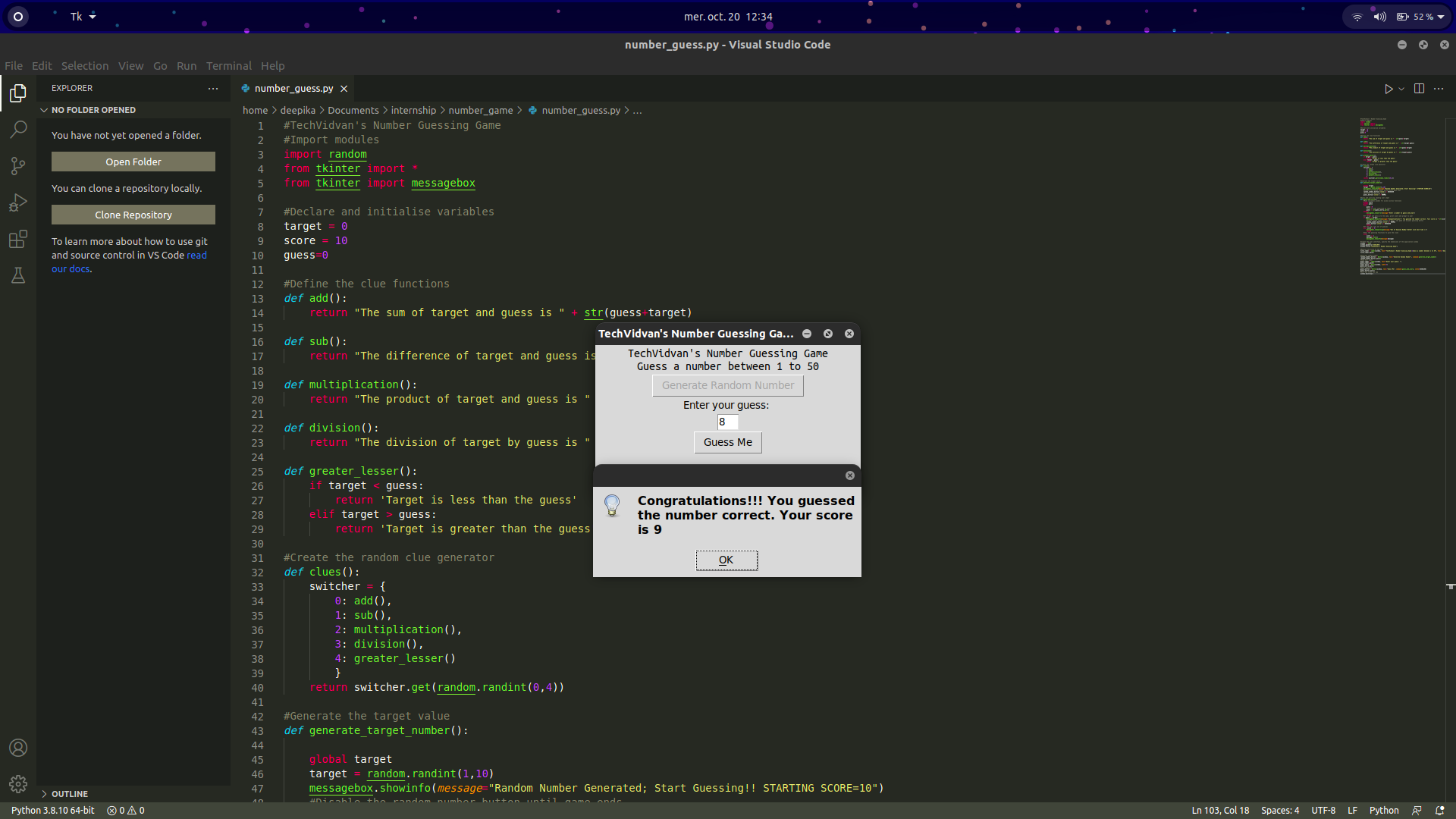
window.mainloop()

**Code explanation:**

* **window = Tk():** Create a window by assigning the tkinter class constructor to it. We can now add widgets and entry fields to our app
* **window.geometry("350x200"):** Set the dimensions of the application window using geometry. The parameters are width and height
* **window.title("TechVidvan's Number Guessing Game"):** Set a title for the titlebar of the application using title parameter
* **title\_label, guess\_label:** To add any non editable text to the window, we use labels. The parameters are window and the text to display.
* **Random\_number\_button, guess\_button:** Buttons invoke a function when the user clicks on them. The command parameter contains the function to call. The other parameters are the name of the button and the window it will sit on. Use state parameter to set on or off a button
* **pack():** Placing a widget on a window makes it visible. So to position it, we use pack which puts it in the center of each row. First element is in the first row, second in the second row and so on
* **window.mainloop()**: Any line of command pertaining to the GUI will not be visible after mainloop. This line indicates the end of the application interface and allows to terminate the application when user closes it

#### **Project output:**

Play the game by starting the random number generator



#### **Summary**

Thus, we created a simple project to understand and create our functions to provide clues and calculate scores of the games. We also created a GUI using tkinter to provide an interface