-:Number gassing game Explaination:-

Step by step :

1]Import required Module :

1)import random(by default module)
2)from tkinter import *(pip install tk)
3)from tkinter import messagebox

2]Declare Varible :
target = 0
score = 10
guess=0

Code explaination :

** import random: Random library generates a random number or a random subset in a string. We use ra ndint function of random
** from tkinter import *: To create the user interface, we import Tkinter. Tkinter contains provisions to buil d 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 mo dule, 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 forthc oming functions. Score is set to 10 which also implies the user gets 10 chances before he loses

3]Write Clue Function :

```
        #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

4]Generate Target Number:

```
def generate_target_number():
  global target
  target = random.randint(1,10)
  messagebox.showinfo(message="Random Number Generated; Start Guessing!! STARTING SCORE=1
0")
  #Disable the random number button until game ends
  random_number_button['state'] = DISABLED
  #Activate the guessing button
  guess_button['state'] = NORMAL

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)



#Create the user interface, specify the dimensions of the application window for Python Number Guessin
g project
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 Explination :

** 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 loc al 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 me ssage parameter set to a message.
** random_number_button['state'], guess_button['state']: After getting a target, we disable the random nu mber 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 t he button. The state of the button attribute is altered to achieve this.
** 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 li miting 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 u sing 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 e arlier. Display the clue using messagebox
** window = Tk(): Create a window by assigning the tkinter class constructor to it. We can now add widge ts and entry fields to our app
** window.geometry("350×200"): Set the dimensions of the application window using geometry. The para meters 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 c
ommand parameter contains the function to call. The other parameters are the name of the button and t
he 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 li
ne indicates the end of the application interface and allows to terminate the application when user close
s it