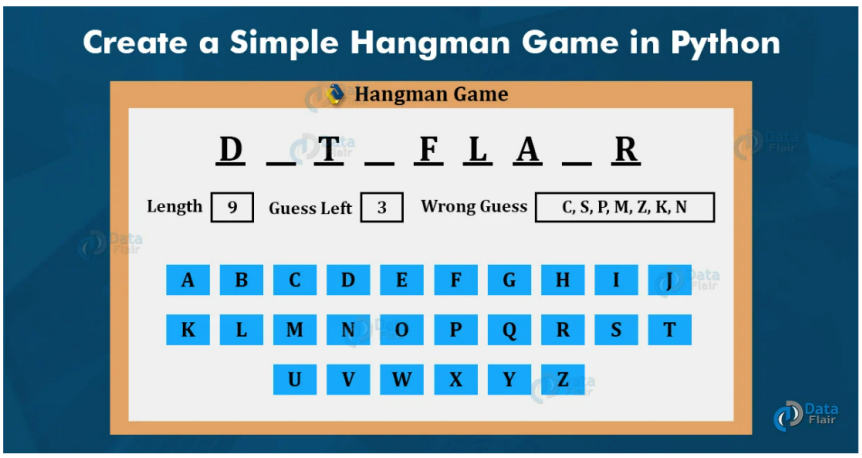
Python Projects +
Python Django Projects +
Machine Learning Proje... +
Deep Learning Projects +
AI Projects +
Python Interview Questi... +
Python Quiz +

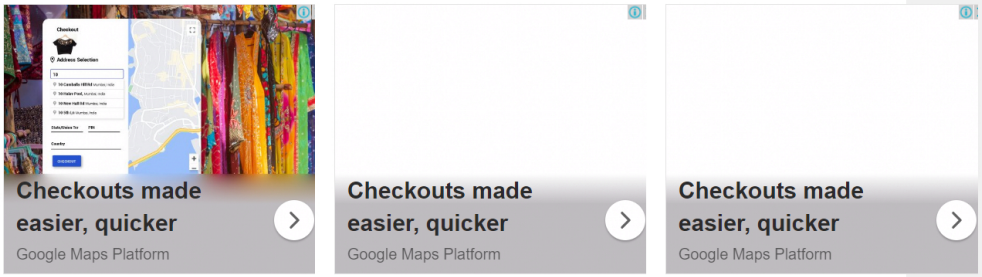
Hangman Game in Python – Simple Game Project for Beginners

Free Python course with 35 real-time projects Start Now!!

Rather than playing games developed by others, let's develop a game in python. Work on python hangman game and master the most popular programming language on the planet



Hangman Game in Python



About Hangman

Going back to our old school days, some of the pen-paper games were always a top for our leisure time. Hangman was one, other than some chit games, to guess words according to the guesses determined and as soon as they lost all their wrong guesses, they were hanged (not really, but on paper 😉). That is an old way, now to play Hangman. The new advancement in technologies allows us to play hangman using our own computer also without any other player. How? Let's find out further

Python Hangman Project

The objective of our project is to implement the hangman game using Python. It doesn't require any specific modules other than random and time. Python loops and functions are enough to build this game here.

Project Prerequisites

This project requires good knowledge of Python which includes defining functions and managing for/while loops. The functions that we use here contain arguments that are defined in a global scope which can be further used in other functions to improve game quality. It can also be used to provide different steps when required to execute upon conditions by the for and while loops.

Download Hangman Python Code

Please download the source code of Python Hangman: Python Hangman Project

Project File Structure

First, let's check the steps to build the Hangman game in Python:

1. Importing the random and time modules.
2. Defining functions with specific global arguments.
3. Implementing loops to execute the program.
4. Passing the function in the program to run.

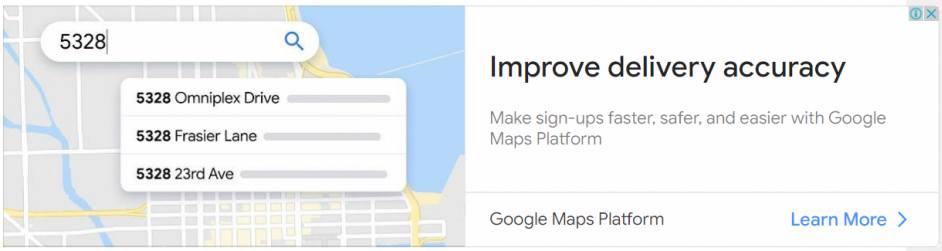
So that is basically what we will do in this Python project. Let's start.

1. Importing the random and time libraries:

```
import random
import time

```

```
# Initial Steps to invite in the game:
print("\nWelcome to Hangman game by DataFlair\n")
name = input("Enter your name: ")
print("Hello " + name + "! Best of Luck!")
time.sleep(2)
print("The game is about to start!\n Let's play Hangman!")
time.sleep(3)
```

Code Explanation:



- **Import random:** This is used to randomly choose an item from a list [] or basically a sequence.
- **Import time:** This module is used to import the actual time from your pc to use in the program.
- **Time.sleep():** This is used to halt the execution of the program for a few seconds. It is a fun way to put the user of the game in short suspense.

2. Define the main function:

```
def main():
    global count
    global display
    global word
    global already_guessed
    global length
    global play_game
    words_to_guess = ["january","border","image","film","promise","kids","lungs","doll","rhyme","damage","plants"]
    word = random.choice(words_to_guess)
    length = len(word)
    count = 0
    display = '_' * length
    already_guessed = []
    play_game = ""
```

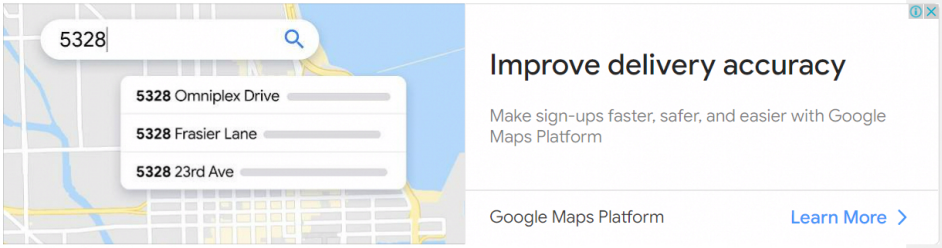
Code Explanation:

- We define the main function that initializes the arguments: global count, global display, global word, global already_guessed, global length and global play_game. They can be used further in other functions too depending on how we want to call them.
- **Words_to_guess:** Contains all the Hangman words we want the user to guess in the game.
- **Word:** we use the random module in this variable to randomly choose the word from words_to_guess in the game.
- **Length:** len() helps us to get the length of the string.
- **Count:** is initialized to zero and would increment in the further code.
- **Display:** This draws a line for us according to the length of the word to guess.
- **Already_guessed:** This would contain the string indices of the correctly guessed words.

3. Develop a loop to execute the game again:

```
# A loop to re-execute the game when the first round ends:

def play_loop():
    global play_game
    play_game = input("Do You want to play again? y = yes, n = no \n")
    while play_game not in ["y", "n","Y","N"]:
        play_game = input("Do You want to play again? y = yes, n = no \n")
    if play_game == "y":
        main()
    elif play_game == "n":
        print("Thanks For Playing! We expect you back again!")
        exit()
```

Code Explanation:



- **Play_loop:** This function takes in the argument of play_game.
- **Play_game:** We use this argument to either continue the game after it is played once or end it according to what the user suggests.
- While loop is used to execute the play_game argument. It takes the parameter, y=yes and n=no. If the user gives an input of something else other than y/n, it asks the question again for the appropriate answer. If the user inputs “y”, the game restarts, otherwise the game ends.

4. Initialize conditions for hangman game:

```
# Initializing all the conditions required for the game:
def hangman():
    global count
    global display
    global word
```

```
    global already_guessed
    global play_game
    limit = 5
    guess = input("This is the Hangman Word: " + display + " Enter your guess: \n")
    guess = guess.strip()
    if len(guess.strip()) == 0 or len(guess.strip()) >= 2 or guess <= "9":
        print("Invalid Input, Try a letter\n")
        hangman()
```

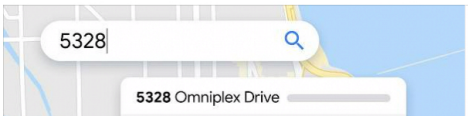
Code Explanation:

- We call all the arguments again under the **hangman() function**.
- **Limit:** It is the maximum guesses we provide to the user to guess a particular word.
- **Guess:** Takes the input from the user for the guessed letter. **Guess.strip()** removes the letter from the given word.
- If loop checks that if no input is given, or two letters are given at once, or a number is entered as an input, it tells the user about the invalid input and executes hangman again.

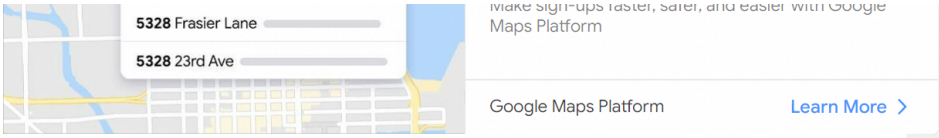
5. The rest of the whole hangman program combined together:

```
    elif guess in word:
        already_guessed.extend([guess])
        index = word.find(guess)
        word = word[:index] + "_" + word[index + 1:]
        display = display[:index] + guess + display[index + 1:]
        print(display + "\n")

    elif guess in already_guessed:
        print("Try another letter.\n")

    else:
        count += 1

        if count == 1:
            time.sleep(1)
            print("   _____ \n"
                  "  |      \n"
                  "  |      \n"
                  "  |      \n"
                  "  |      \n"
                  "  |      \n"
                  "  |      \n"
                  "__|__\n")
            print("Wrong guess. " + str(limit - count) + " guesses remaining\n")

        elif count == 2:
            time.sleep(1)
            print("   _____ \n"
                  "  |     | \n"
                  "  |     |\n"
                  "  |      \n"
                  "  |      \n"
                  "  |      \n"
                  "  |      \n"
                  "__|__\n")
            print("Wrong guess. " + str(limit - count) + " guesses remaining\n")

        elif count == 3:
           time.sleep(1)
           print("   _____ \n"
                 "  |     | \n"
                 "  |     |\n"
                 "  |     | \n"
                 "  |      \n"
                 "  |      \n"
                 "  |      \n"
                 "__|__\n")
           print("Wrong guess. " + str(limit - count) + " guesses remaining\n")

        elif count == 4:
            time.sleep(1)
            print("   _____ \n"
                  "  |     | \n"
                  "  |     |\n"
                  "  |     | \n"
                  "  |     O \n"
                  "  |      \n"
                  "  |      \n"
                  "__|__\n")
            print("Wrong guess. " + str(limit - count) + " last guess remaining\n")

        elif count == 5:
            time.sleep(1)
            print("   _____ \n"
                  "  |     | \n"
                  "  |     |\n"
                  "  |     | \n"
                  "  |     O \n"
                  "  |    /|\ \n"
                  "  |    / \ \n"
                  "__|__\n")
            print("Wrong guess. You are hanged!!!\n")
            print("The word was:",already_guessed,word)
            play_loop()

    if word == '_' * length:
        print("Congrats! You have guessed the word correctly!")
        play_loop()

    elif count != limit:
        hangman()


main()

hangman()
```

Code Explanation:



Improve delivery accuracy



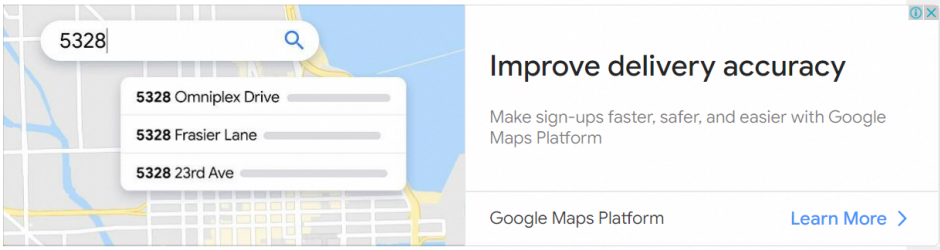
- If the letter is correctly guessed, index searches for that letter in the word.
- Display adds that letter in the given space according to its index or where it belongs in the given word.
- If we have already guessed the correct letter before and we guess it again, It tells the user to try again and does not lessen any chances.
- If the user guessed the wrong letter, the hangman starts to appear which also tells us how many guesses are left. Count was initialized to zero and so with every wrong guess its value increases with one.
- Limit is set to 5 and so (limit- count) is the guesses left for the user with every wrong input. If it reaches the limit, the game ends, showing the right guesses (if any) and the word that was supposed to be guessed.
- If the word is guessed correctly, matching the length of the display argument, the user has won the game.
- Play_loop asks the user to play the game again or exit.
- Main() and hangman() would start again if the play_loop executes to yes.

Hurray!! The game is all set to play and pass your leisure time by executing and playing Hangman on your own 😄

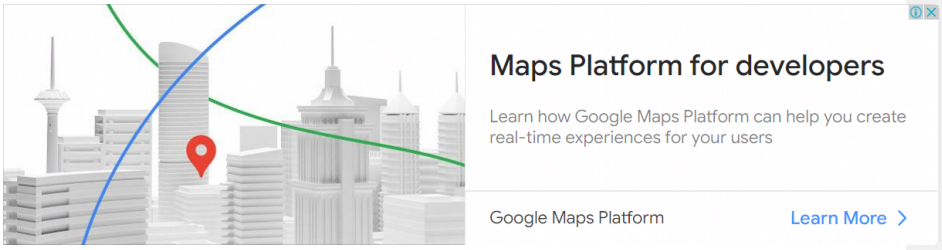
Project output

Enter Your Name:

Hangman Game Starts:



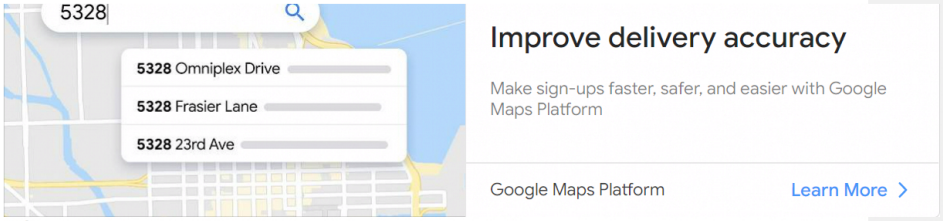
Now, play the Hangman game which you developed. Kudos!



Summary

With this project in Python, we have successfully developed the Hangman game. We used the popular time and random modules to render our program. Executing different functions and using loops helped us with a better understanding of python basics.

We work very hard to provide you quality material



Reply

bugra August 13, 2020 at 6:28 pm

so we can try to fix that problem right? 😀

Reply

Mohamed October 9, 2020 at 2:34 am

This is how I made it work:

```
#Begin code

shape_list=list(shape)
word_list=list(your_word)
p=0
for x in word_list:
if x==f:
n=word_list.index(x,p)
p=n+1
shape_list[n]=f
shape="".join(shape_list)

#End code
```

Reply

DataFlair Team October 29, 2020 at 10:23 pm

Thanks for helping the community, keep learning, keep growing

Reply

Zika August 18, 2020 at 10:33 am

what you can do is, create a function that checks how many times a character appears, and saves the characters in a list. Afterward, call the function at the point in the program where it finds the character in the word and loop through the number of times the character appears in the word to replace the '_' with the word.

in the code above the lines:
index = word.find(guess)
word = word[:index] + '_' + word[index + 1:]
display = display[:index] + guess + display[index + 1:]

the above code finds a single character in the word, you can use a loop to run this code the number of times the character appears to replace the underscores with the character.

Reply

DataFlair Team October 29, 2020 at 9:48 pm

Thanks for helping the community, keep learning keep helping keep mastering 😊

Reply

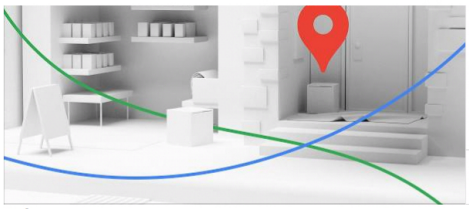
Ajay Basumatary August 20, 2020 at 11:24 pm

Thanks for the code ....

Reply

Gino de Reuver August 25, 2020 at 3:20 am

the code doesn't work for me. I get an indentationerror for this line: 'elif guess in word:' can someone help me with this. I even just copied this part of the code.



Predict delivery with accuracy

Quickest or shortest? Learn how Google Maps Platform optimises your last mile delivery

Google Maps Platform Learn More

Reply

DataFlair Team October 29, 2020 at 9:45 pm

Please download the code and then run, copying and pasting usually result in improper indentation.

Reply

Raju kumar August 26, 2020 at 3:32 pm

sir problem occured in line no :- 58

elif guess in word:

Reply

DataFlair Team October 29, 2020 at 9:46 pm

This might be an indentation error, the indentation should match with previous if statement

Reply

Sara Amaya February 26, 2022 at 2:00 pm

I just erase the previous space when writing the previous if like this:

```
if count ==1:
time.sleep(1)
print(" _ .....
if count ==1:
time.sleep(1)
print(" _.....
```

I did also remove the spaces on the next elif/s and now is not in red color on the code.

Reply

Nobita September 22, 2020 at 9:28 am

#try this i have removes few lines from code and its working if any issue founds let me know

```
import random
import time

print("HANGMAN GAME")
name = input("Enter your name : ")
print("Hello " + name + " Best Of Luck!")
time.sleep(2)
print("The Game about to START!\n\t Let's Play Hangman")
time.sleep(3)

def main():
global count
global display
global word
global already_guessed
global length
global play_game

words_to_guess = ["january","border","image","film","promise","kids","lungs","doll","rhyme","damage","plants"]
word = random.choice(words_to_guess)
length = len(word)
count = 0
display = '_ ' * length
already_guessed = []
play_game = ""

def play_loop():
global play_game
play_game = input("Do you want to play again ? Y = yes , N = no")



while play_game not in ['Y','y','n','N']:
play_game = input("Do you want to play again ? Y = yes , N = no")
if play_game == 'y' or 'Y':
main()
elif play_game == 'n' or 'N':
```

```
elif play_game == 'n' or 'N':
print("thanks for playing! come back again")
exit()

def hangman():
global count
global display
global word
global already_guessed
global play_game
limit = 5
guess = input("This is the Hangman Word: " + display + " Enter your guess: \n")
guess = guess.strip()
#if len(guess.strip()) == 0 or len(guess.strip()) >= 2 or guess <= "9":
# print("Invalid Input, Try a letter\n")
# hangman()
if guess in word:
already_guessed.extend([guess])
index = word.find(guess)
word = word[:index] + "_" + word[index + 1:]
display = display[:index] + guess + display[index + 1:]
print(display + "\n")
elif guess in already_guessed:
print("Try another letter.\n")
else:
count += 1
if count == 1:
time.sleep(1)
print(" _____ \n"
" | \n"
" | \n"
" | \n"
" | \n"
" | \n"
" | \n"
"__|__\n")
print("Wrong guess. " + str(limit – count) + " guesses remaining\n")
elif count == 2:
time.sleep(1)
print(" _____ \n"
" | | \n"
" | |\n"
" | \n"
" | \n"
" | \n"
" | \n"
"__|__\n")
print("Wrong guess. " + str(limit – count) + " guesses remaining\n")
elif count == 3:
time.sleep(1)
print(" _____ \n"
" | | \n"
" | |\n"
" | | \n"
" | \n"
" | \n"
" | \n"
"__|__\n")
print("Wrong guess. " + str(limit – count) + " guesses remaining\n")
elif count == 4:
time.sleep(1)
print(" _____ \n"
" | | \n"
" | |\n"
" | | \n"
" | O \n"
" | \n"
" | \n"
"__|__\n")
print("Wrong guess. " + str(limit – count) + " last guess remaining\n")
elif count == 5:
time.sleep(1)
print(" _____ \n"
" | | \n"
" | |\n"
" | | \n"
" | O \n"
" | /|\ \n"
" | / \ \n"
"__|__\n")
print("Wrong guess. You are hanged!!!\n")
print("The word was:",already_guessed,word)
play_loop()
if word == '_' * length:
print("Congrats! You have guessed the word correctly!")
play_loop()
elif count != limit:
hangman()

main()
```

hangman()

Reply

DataFlair Team October 29, 2020 at 9:47 pm

Thanks for helping the community, keep learning keep helping keep mastering 😉

Reply

thanika October 26, 2020 at 9:00 am

elif guess in word:
loop=word.count(guess)
while(loop>0):
already_guessed.extend([guess])
index = word.find(guess)
word = word[:index] + "_" + word[index + 1:]
display = display[:index] + guess + display[index + 1:]
loop-=1
print(display + "\n")

Reply

Farshad November 5, 2020 at 2:27 pm

Hello, thank you so much for these clean codes.
Could you please explain why you used main() in this piece if code?
if play_game == "y":
main()

Reply

DataFlair Team December 8, 2020 at 9:51 pm

main function declares and initializes all the required variables to play the game which is why we have called the main function when the user wants to play the game (when the user enters 'y').

Reply

AlexM January 13, 2021 at 4:26 am

Hi everybody. I am having a recurring problem: After it tells me "The game is about to start, Let's play hangman", the program stops running and nothing else happens. I'm using pycharm and this appears: Process finished with exit code 0
But I have all the rest of the codes afterwards. So I cannot play the game. Do you know what's going on?

Reply

Ibrahim Alshahari March 1, 2021 at 4:18 am

Make sure to call the functions
main()
hangman()
At the end of the code.

Reply

DataFlair Team August 3, 2021 at 8:21 pm

It seems you are not calling the function after the definition, just call main() and hangman function. For more clarification try downloading the code provided and run it.

Reply

ubant March 7, 2021 at 6:46 pm

My code in Tkinter is someone wants to compare their:

```
from tkinter import *
from random import randint

root = Tk()
root.title("Hang Man")
root.configure(bg="#303030")
# root.geometry("800×500")

# Word to guess section
words_to_guess = ["january", "border", "image", "film", "promise", "kids", "lungs", "doll", "rhyme", "damage", "plants"]
word = words_to_guess[randint(0, 10)]
word_screen = word
# print(word)
guessed_word = len(word) * "_"
empty_word = Label(root, font=("Calibri", 70), text=" ".join(guessed_word), bg="#303030", fg="white")
empty_word.grid(column=1, row=0, columnspan=5, pady=20)
# Guess specify section
lengthName = Label(root, font=("Calibri", 20), text="Length", bg="#303030", fg="white")
lengthLabel = Label(root, font=("Calibri", 20), text=" " + str(len(word)) + " ", bg="#303030", fg="white",
borderwidth=2, relief="ridge")
lengthName.grid(column=0, row=1, padx=(20, 5))
lengthLabel.grid(column=1, row=1)
guessLeft = 10
wrongGuess = []
GuessLeftName = Label(root, font=("Calibri", 20), text="Guess Left", bg="#303030", fg="white")
GueswsLeftLabel = Label(root, font=("Calibri", 20), text=" " + str(guessLeft) + " ", bg="#303030", fg="white", borderwidth=2,
relief="ridge")
GuessLeftName.grid(column=2, row=1, padx=(20, 5))
GueswsLeftLabel.grid(column=3, row=1)
WrongGuessName = Label(root, font=("Calibri", 20), text="Wrong Guess", bg="#303030", fg="white")
WrongGuessLabel = Label(root, font=("Calibri", 20), text=", ".join(wrongGuess), bg="#303030", fg="white",
borderwidth=2, relief="ridge")
WrongGuessName.grid(column=4, row=1, padx=(20, 5))
WrongGuessLabel.grid(column=5, row=1, padx=(0, 25))

# Alphabet section
alphB = Canvas(root, width=800, bg="#303030", highlightthickness=0)
alphB.grid(column=0, row=2, columnspan=6, pady=20, padx=(30, 0))

x = 0
y = 0
z = 0
letters = ["A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U", "V"
```

```
letters = ["A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U", "V",
"W", "X", "Y", "Z"]
for a in range(len(letters)):
z += 1
y += 1
if z == 21:
y += 2
b = Button(alphB, font=("Calibri", 20, "bold"), text=letters[z – 1], width=3, bg="#1d3357", fg="white",
relief="groove", borderwidth=2, highlightcolor="black")
b.config(command=lambda t=letters[a], b=b: letterB(t, b))
b.grid(column=y, row=x, padx=5, pady=10)
if y == 10:
x += 1
y = 0

def letterB(text, b):
global guessed_word, word, guessLeft
if text.lower() in word:
for e in word:
if e == text.lower():
guessed_word = guessed_word[:word.index(e)] + e + guessed_word[word.index(e)+1:]
word = word[:word.index(e)] + "-" + word[word.index(e)+1:]
# print(guessed_word)
empty_word.config(text=" ".join(guessed_word))
if guessed_word == word_screen:
empty_word.destroy()
lengthLabel.destroy()
lengthName.destroy()
GuessLeftName.destroy()
GueswsLeftLabel.destroy()
WrongGuessName.destroy()
WrongGuessLabel.destroy()
alphB.destroy()
endScreen = Label(root, font=("Calibri", 50), text=f"You won !\nWord to guess was \"{word_screen}\"",
bg="#303030", fg="white")
endScreen.grid(padx=200, pady=200)

else:
guessLeft -= 1
GueswsLeftLabel.config(text=" " + str(guessLeft) + " ")
wrongGuess.append(text)
WrongGuessLabel.config(text=", ".join(wrongGuess))
if guessLeft == 0:
empty_word.destroy()
lengthLabel.destroy()
lengthName.destroy()
GuessLeftName.destroy()
GueswsLeftLabel.destroy()
WrongGuessName.destroy()
WrongGuessLabel.destroy()
alphB.destroy()
endScreen = Label(root, font=("Calibri", 50), text=f"You lost :(\nWord to guess was \"{word_screen}\"", bg="#303030",
fg="white")
endScreen.grid(padx=200, pady=200)

mainloop()
```

Reply

DataFlair Team August 3, 2021 at 8:22 pm

Your efforts are appreciated

Reply

ShradhaPol August 3, 2021 at 1:04 pm

Very nice

Reply

Kel August 10, 2021 at 7:13 pm

I'm running into the following issue: "NameError: global name 'already_guessed' is not defined" It goes away if I define already_guessed within hangman(), but it would wipe the list clean every cycle.

Reply

Tejaswini Akketi November 17, 2021 at 11:07 am

Hi,

I am new to Python and I can't understand the below block of code. Can anyone help me out please ?

```
print(" _____ \n"
" | | \n"
" | |\n"
" | | \n"
" | \n"
" | \n"
" | \n"
"__|__\n")
```

Reply

knonx November 25, 2021 at 8:14 pm

The code runs when special character are entered like ~ or |, it accepts it as a valid letter.

My fix is to use isalpha() method on the string.

```
guess = input("This is the Hangman Word: " + display + " Enter your guess: \n")
guess = guess.strip()
if len(guess.strip()) == 0 or len(guess.strip()) >= 2 or not guess.isalpha():
print("Invalid Input, Try a letter\n")
```

also, i think it would be nice if the codes are written in to conform with latest python version.

like using the f string method to concatenate strings and string with int
print(f"Hello{variable}") is more clean than print("Hello" + variable)
I like your platform, I've learnt a lot from it and i'm learning... Kudos to DATAFLAIR TEAM

Reply

mahdi December 10, 2021 at 3:36 pm

Hello
I think there is a problem in your code .
In play_loop function if user wants to play again you gotta call main function and hangman function together but you have only called main function in your code .
If it is wrong tell me why please .
Thanks for your challenging codes .

Reply

Amihay Goldenberg April 30, 2022 at 6:50 pm

well, reading this comment after 5 Hrs. work to find what is wrong with my code..... filling like I don't want to code anymore 🙂

Reply

Abhijeet Kumar July 21, 2022 at 3:40 pm

This is the Hangman Word: _____Enter your guess:
Plants
Invalid Input, Try a letter

Reply

bugra yildirim July 26, 2022 at 1:15 am

These are my contrubitions;

for finding all letters in the word:

index = ([pos for pos, char in enumerate(word) if char == guess])
for index in index:
word = word[:index] + "_" +word[index+1:]
display = display[:index] + guess + display[index+1:]

for looking the original word;

def main():
global word_org
word_org = word
.....
.....

print("The word was:",word_org)

for playing again:
if play_game.lower() == "y":
main()
hangman()

Reply

LEAVE A REPLY

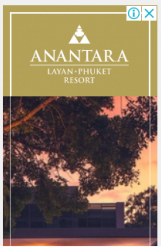
Comment *

Name *

Email *

Post Comment

Python Projects +
Python Django Projects +
Machine Learning Proje... +
Deep Learning Projects +
AI Projects +
Python Interview Questi... +
Python Quiz +



DataFlair © 2022. All Rights Reserved.