IP Project

Introduction

This project uses Python. Python is an interpreted, high-level and general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace.

Modules used:
random
webbrowser
time

Python is a general-purpose interpreted, interactive, object-oriented, andhigh-level programming language. It was created by Guido van Rossumduring 1985- 1990. Its language constructs and object-oriented approachaim to help programmers write clear, logical code for small and large-scale projects. It uses English keywords frequently where as other languages usepunctuation, and it has fewer syntactical constructions than other languages.

Characteristics of Python
Following are important characteristics of Python Programming −
1, It supports functional and structured programming methods as well as OOP.
2, It can be used as a scripting language or can be compiled to byte- code for building large applications.
3, It provides very high-level dynamic data types and supports dynamic type checking.
4, It supports automatic garbage collection.
5, It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Applications of Python
As mentioned before, Python is one of the most widely used language over the web.
Easy-to-learn − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
Easy-to-read − Python code is more clearly defined and visible to the eyes.
Easy-to-maintain − Python' source code is fairly easy-to-maintain.
A broad standard library − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
Interactive Mode − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
Portable − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

Extendable − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
Databases − Python provides interfaces to all major commercial databases.
GUI Programming − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
Scalable − Python provides a better structure and support for large programs than shell scripting.

WEBBROWSER MODULE
Introduction
According to Python's standard documentation, the webbrowser module provides a high-level interface to allow displaying Web-based documents to users. This topic explains and demonstrates proper usage of the webbrowser module.
Syntax
webbrowser.open(url, new=0, autoraise=False)
webbrowser.open_new(url)

webbrowser.open_new_tab(url)
webbrowser.get(usage=None)
webbrowser.register(name, constructor, instance=None)

Parameters
webbrowser.open()
url the URL to open in the web browser
webbrowser.open_new()
url the URL to open in the web browser
webbrowser.open_new_tab()
url the URL to open in the web browser
webbrowser.get()
using the browser to use
webbrowser.register()
url browser name
constructor path to the executable browser (help)
instance An instance of a web browser returned from
the webbrowser.get() method

Example
To simply open a URL, use the webbrowser.open() method:
import webbrowser
webbrowser.open('http://stackoverflow.com')
If a browser window is currently open, the method will open a new tab at the specified URL. If no window is open, the method will open the operating system's default browser and navigate to the URL in the parameter. The open method supports the following parameters:

url - the URL to open in the web browser (string) [required]
new - 0 opens in existing tab, 1 opens new window, 2 opens new tab
(integer) [default 0]

autoraise - if set to True, the window will be moved on top of the other
application's; windows (Boolean) [default False]
Note, the new and autoraise arguments rarely work as the majority of modern browsers refuse these commmands. Webbrowser can also try to open URLs in new windows with the open_new method:

```
import webbrowser
webbrowser.open_new('http://stackoverflow.com')
```

This method is commonly ignored by modern browsers and the URL is usually opened in a new tab. Opening a new tab can be tried by the module using the open_new_tab method:

```
import webbrowser
webbrowser.open_new_tab('http://stackoverflow.com')
```

RANDOM MODULE
Functions in the random module depend on a pseudo- random number generator function random(), which generates a random float number between 0.0 and 1.0.

1.random.random(): Generates a random float number between 0.0 to 1.0. The function doesn't need any arguments.

```
import random
random.random()
>>>0.645173684807533
```

2.random.randint(): Returns a random integer between the specified integers.

```
import random
random.randint(1,100)
>>>95
random.randint(1,100)
>>>49
```

3.random.randrange(): Returns a randomly selected element from the range created by the start, stop and step arguments. The value of start is 0 by default. Similarly, the value of step is 1 by default.

```
random.randrange(1,10)
>>>2
random.randrange(1,10,2)
>>>5
random.randrange(0,101,10)
>>>80
```

4.random.choice(): Returns a randomly selected element from a non-empty sequence. An empty sequence as an argument raises an IndexError.

import random
random.choice('computer')
>>>39;
random.choice([12,23,45,67,65,43])
>>>45

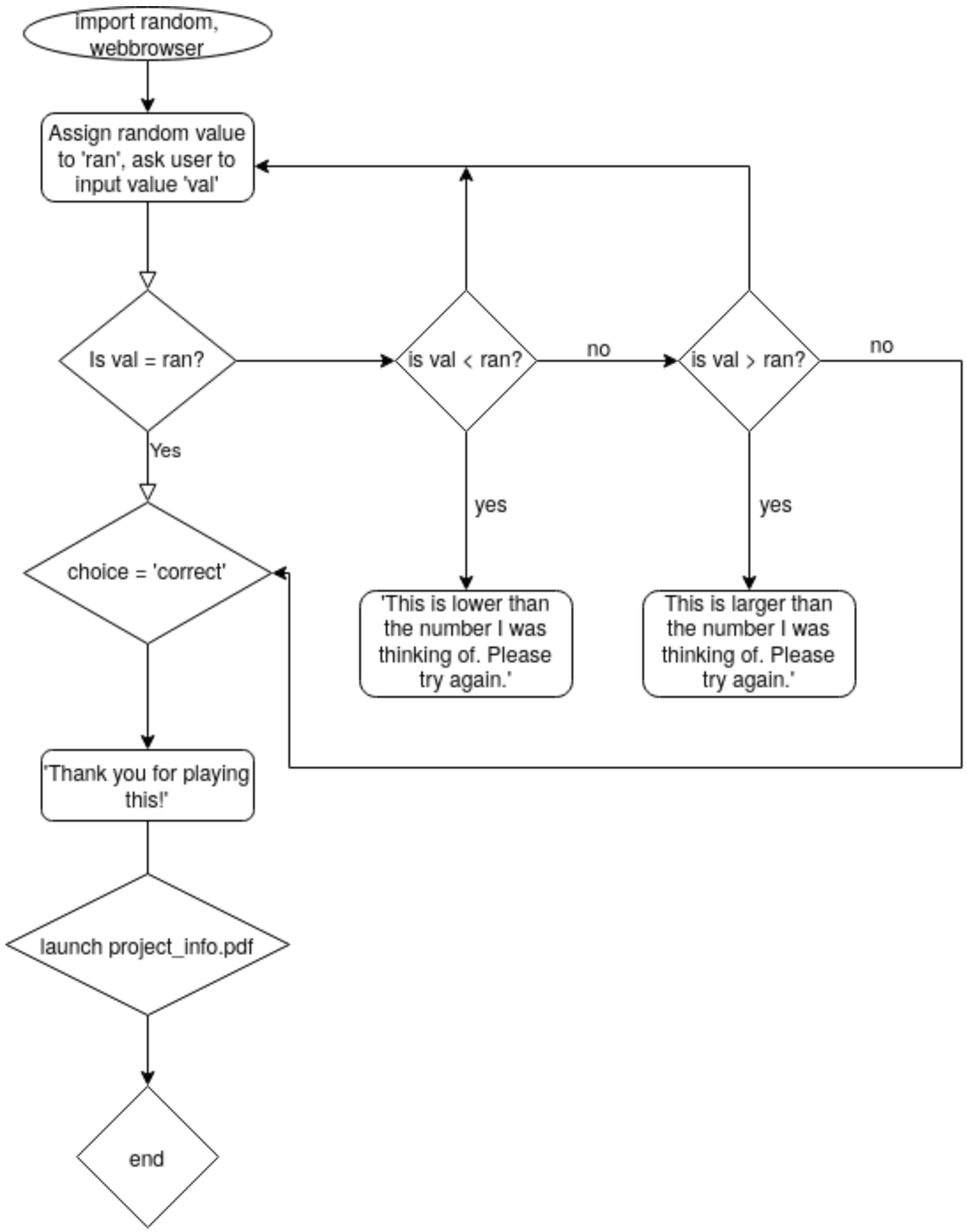
random.choice((12,23,45,67,65,43))
>>>67
5.random.shuffle(): This functions randomly reorders the
elements in a list.
numbers=[12,23,45,67,65,43]
random.shuffle(numbers)
numbers
>>>[23,12,43,65,67,45]
random.shuffle(numbers)
numbers
>>>[23,43,65,45,12,67

Objective

This code includes guessing a random number between 1-100.
For the working of this code modules such as webbrowser and random are used.
Program will be in loop until the user guesses the right number.
Once the guess is successful. The program will end.
After this, the user will proceed to project
Documentation.

How it works

The code assigns a random value to variable ‘ran’ using the random module. The user is then asked to enter a value of their choice between 1 and 100. The value entered is assigned to var ‘val’. If ‘val’ is greater than ‘ran’, the code prints out saying that the number is greater than the one entered, and vice versa. When the user finally gets the number right, it prints out saying that the number is right and that the game has finished. The code will then launch project_info.pdf (this file) using the webbrowser module.


import random, webbrowser
Assign random value to 'ran', ask user to input value 'val'
Is val = ran?
Yes
is val < ran?
no
is val > ran?
no
yes
yes
choice = 'correct'
'This is lower than the number I was thinking of. Please try again.'
This is larger than the number I was thinking of. Please try again.'
'Thank you for playing this!'
launch project_info.pdf
end

System requirements

OS: Must be able to boot successfully
RAM: Above 516 MB
GPU: Not required
CPU: Must exist
Look, even Grandma Ivanka’s PC from the Soviet Era will be able to run this. Don’t worry.

Bibliography

1, https://www.wikipedia.org
2, https://docs.python.org
3, https://stackoverflow.com

A project by:
Siddharth
Adhishri
Janhavi
(12D)