AZURE SKYNET
TECHNOLOGY | KNOWLEDGE | INNOVATION

[PYTHON PROGRAMMING 2022]

[Summer Internship Project Report]

Address
M-4, 1st Floor, Old DLF Colony, Sector 14, Gurugram, Haryana 122001
Azure Skynet Solutions Private Limited, address

Aashish Tharu Gamuwa
aashishtharu819@gmail.com

Acknowledgement

While I was preparing this project file, various information that I found helped me in chapter *Password security* and I am glad that I was able to complete this project and understand many things. Through preparation of this computer science project was an immense learning experience and I inculcated many personal qualities during this process like responsibility, punctuality, confidence and others.

I would like to thank to my teachers who supported me all the time, cleared my doubts and to my parents who also played a big role in finalization of my project file. I am taking this opportunity to acknowledge their support and I wish that they keep supporting me like this in the future.

A project is a bridge between theoretical and practical learning and with this thinking I worked on the project and made it successful due to timely support and efforts of all who helped me.

Once again, I would like to thank my classmates and my friends also for their encouragement and help in designing and making my project creative. I am in debt of all these. Only because of them I was able to create my project and make it good and enjoyable experience.

Content

- Acknowledgement
- Inteoduction
- Project 1 (basic)-Casear Cipher
- Project 2(Intermediate)-Tiny URL
- Project 3(Advanced)-Epidemeic Simulator
- Conclusion
- Refrence

Introduction

A Brief Introduction :

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

Learning Outcomes :

- Describe the characteristics of Python as a high-level, general-purpose programming language.
- Python is widely used in major domains such as web development, AI/ML, and data analytics.
- Explain the popularity of Python among beginners and professionals in various domains.
- Demonstrate how to get started with Python programming using essential resources and IDEs.
- Recognize real-world applications of Python in organizations like NASA, Netflix, and Instagram.
- Outline the steps to become proficient in Python programming and resources for further learning.

Project-1

Basic – Caesar Cipher

Introduction :

In this technique, each character is substituted by a letter certain fixed number position it's later or before the alphabet. For example - Alphabet B is replaced by two positions down D. D would become F and so on. This method is named after popular frictional characters Julius Caesar, who used it to communicate with officials.

There is an algorithm used to implement it. Let's understand the following.

Feature of Caesar Cipher Algorithm

This algorithm consists of a few features that are given below.

- **This technique is quite simple to apply encryption.**
- **Each text is replaced by the fixed number of position down or up with the alphabet.**
- **It is a simple type of substitute cipher.**

There is an integer value required to define each latter of the text that has been moved down. This integer value is also known as the shift.

We can represent this concept using modular arithmetic by first transmuting the letter into numbers, according to the schema, A = 0, B = 1, C = 2, D = 3…….. Z = 25.

The following mathematical formula can be used to shift n letter.

How to decrypt?

The decryption is the same as encryption. We can create a function that will accomplish shifting in the opposite path to decrypt the original text. However, we can use the cyclic property of the cipher under the module.

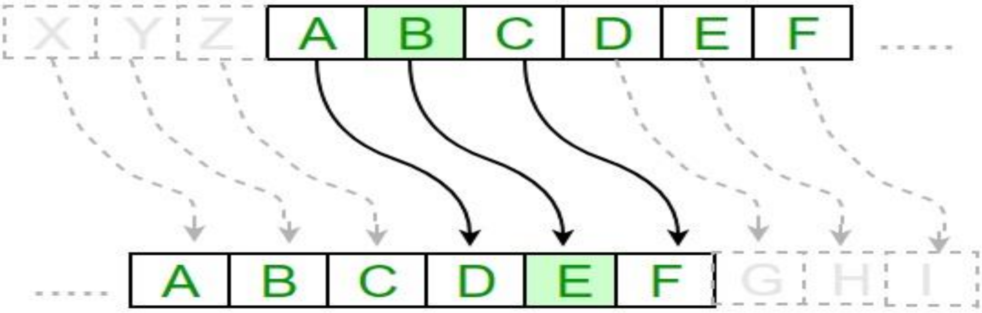
Cipher(n) = De-cipher(26-n)

The same function can be used for decryption. Instead, we will modify the shift value such that shifts = 26 - shift.

$$\boxed{En\ (x)=(x+n)\bmod 26}$$

(Encryption Phase with Shift n)

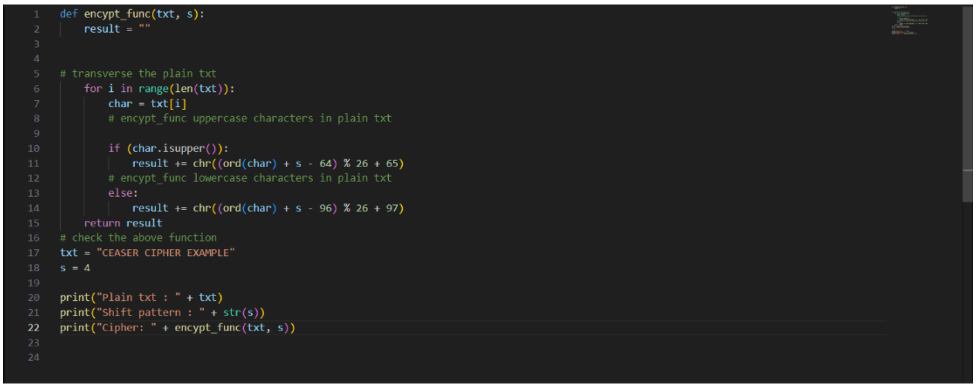
$$\boxed{Dn(x)=(x+n)\bmod 26}$$



This Photo by Unknown Author is licensed under CC BY-SA

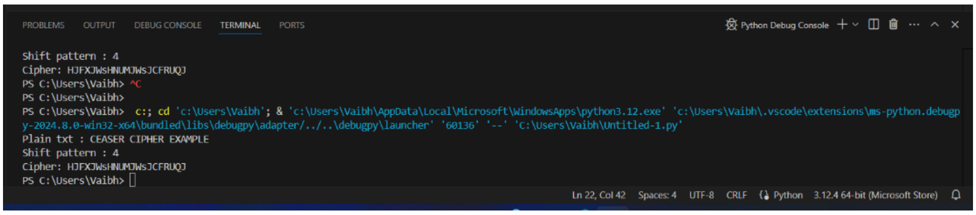
Let's understand the following example -

Example –(Programe 1)



```
def encypt_func(txt, s):
    result = ""


# transverse the plain txt
    for i in range(len(txt)):
        char = txt[i]
        # encypt_func uppercase characters in plain txt

        if (char.isupper()):
            result += chr((ord(char) + s - 64) % 26 + 65)
        # encypt_func lowercase characters in plain txt
        else:
            result += chr((ord(char) + s - 96) % 26 + 97)
    return result
# check the above function
txt = "CEASER CIPHER EXAMPLE"
s = 4

print("Plain txt : " + txt)
print("Shift pattern : " + str(s))
print("Cipher: " + encypt_func(txt, s))
```

Output:



```
Shift pattern : 4
Cipher: HJFXJWsHNUMJWsJCFRUQJ
PS C:\Users\Vaibh> ^C
PS C:\Users\Vaibh>
PS C:\Users\Vaibh>  c:; cd 'c:\Users\Vaibh'; & 'c:\Users\Vaibh\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\Vaibh\.vscode\extensions\ms-python.debugp
y-2024.8.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '60136' '--' 'C:\Users\Vaibh\Untitled-1.py'
Plain txt : CEASER CIPHER EXAMPLE
Shift pattern : 4
Cipher: HJFXJWsHNUMJWsJCFRUQJ
PS C:\Users\Vaibh> 
```

The above code has traversed the character at a time a time. It transferred the each character as per rule depending on the procedure of encryption and decryption of the text.

We have defined few specific set of position that generated a cipher text.

Project - 2

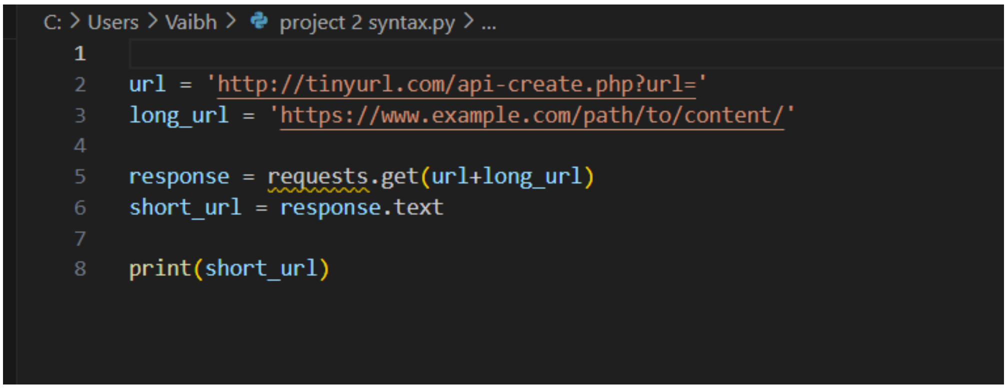
Introduction :

In the web era, concise links are crucial to distribute hyperlinks via social networking sites, text messages, and alternative communication methods. Nevertheless, lengthy URLs might pose challenges when sharing and might be truncated when sending messages. The long URLs are frequently challenging to memorize and can be highly unwieldy to enter. In order to solve the issue at hand, web address shortening platforms such as TinyURL were created to manage the duty. Python offers a convenient approach to connecting with these options. Within this post, we are going to write a Python code to engage with the TinyURL website API system.

Definition :

A link shortener is a software that receives a lengthy URL as data and produces a smaller, more convenient URL. This aids to create extended URLs simpler to exchange and recall. This abbreviated URL forwards people to the authentic prolonged URL upon clicking. Link shorteners are widely utilized on social networking sites, electronic mail communications, and any circumstance when lengthy URLs have to be exchanged conveniently. These tools to shorten extended website links into smaller and easier−to−handle links.

Syntax :

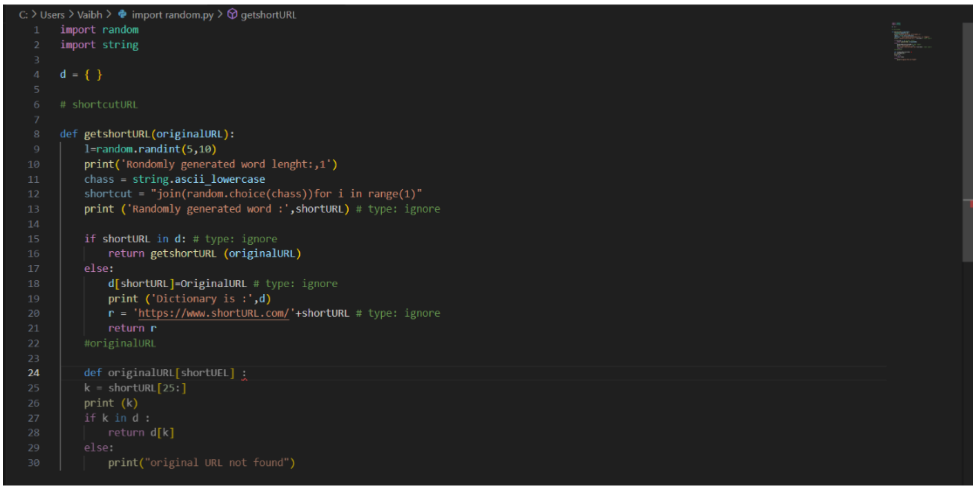


```python

url = 'http://tinyurl.com/api-create.php?url='
long_url = 'https://www.example.com/path/to/content/'

response = requests.get(url+long_url)
short_url = response.text

print(short_url)
```

This piece of code initially imports the module for making requests to perform requests over HTTP. The variable named "url" holds the base link for the TinyURL application programming interface. This "original_url" variable stores the web address that we need to reduce in length. Next, we make an HTTP request for the TinyURL API employing the requests.get() method and handing over the whole URL of the API with the prolonged URL attached.

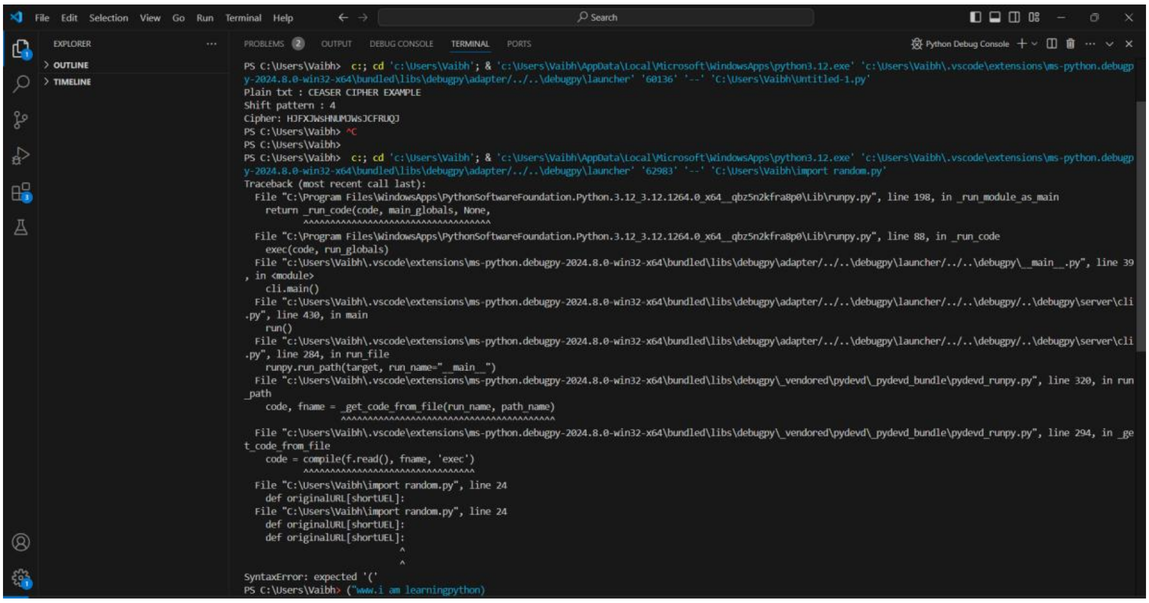
The reply from the TinyURL application programming interface is sent back as textual data using the response.text property. That is subsequently assigned to the variable named "short_url". Ultimately, the code displays the abbreviated web address.

Lets do programe 2 :



```python
import random
import string

d = { }

# shortcutURL

def getshortURL(originalURL):
    l=random.randint(5,10)
    print('Rondomly generated word lenght:,1')
    chass = string.ascii_lowercase
    shortcut = "join(random.choice(chass))for i in range(1)"
    print ('Randomly generated word :',shortURL) # type: ignore

    if shortURL in d: # type: ignore
        return getshortURL (originalURL)
    else:
        d[shortURL]=OriginalURL # type: ignore
        print ('Dictionary is :',d)
        r = 'https://www.shortURL.com/'+shortURL # type: ignore
        return r
    #originalURL

    def originalURL[shortUEL] :
    k = shortURL[25:]
    print (k)
    if k in d :
        return d[k]
    else:
        print("original URL not found")
```

Output:



```
PS C:\Users\Vaibh>  c:; cd 'c:\Users\Vaibh'; & 'c:\Users\Vaibh\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\Vaibh\.vscode\extensions\ms-python.debugp
y-2024.8.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '60136' '--' 'C:\Users\Vaibh\Untitled-1.py'
Plain txt : CEASER CIPHER EXAMPLE
Shift pattern : 4
Cipher: HJFXJWsHNUMJWsJCFRUQJ
PS C:\Users\Vaibh> ^C
PS C:\Users\Vaibh>
PS C:\Users\Vaibh>  c:; cd 'c:\Users\Vaibh'; & 'c:\Users\Vaibh\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\Vaibh\.vscode\extensions\ms-python.debugp
y-2024.8.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '62983' '--' 'C:\Users\Vaibh\import random.py'
Traceback (most recent call last):
  File "C:\Program Files\WindowsApps\PythonSoftwareFoundation.Python.3.12_3.12.1264.0_x64__qbz5n2kfra8p0\Lib\runpy.py", line 198, in _run_module_as_main
    return _run_code(code, main_globals, None,
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Program Files\WindowsApps\PythonSoftwareFoundation.Python.3.12_3.12.1264.0_x64__qbz5n2kfra8p0\Lib\runpy.py", line 88, in _run_code
    exec(code, run_globals)
  File "c:\Users\Vaibh\.vscode\extensions\ms-python.debugpy-2024.8.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher/../..\debugpy\__main__.py", line 39
, in <module>
    cli.main()
  File "c:\Users\Vaibh\.vscode\extensions\ms-python.debugpy-2024.8.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher/../..\debugpy/..\debugpy\server\cli
.py", line 430, in main
    run()
  File "c:\Users\Vaibh\.vscode\extensions\ms-python.debugpy-2024.8.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher/../..\debugpy/..\debugpy\server\cli
.py", line 284, in run_file
    runpy.run_path(target, run_name="__main__")
  File "c:\Users\Vaibh\.vscode\extensions\ms-python.debugpy-2024.8.0-win32-x64\bundled\libs\debugpy\_vendored\pydevd\_pydevd_bundle\pydevd_runpy.py", line 320, in run
_path
    code, fname = _get_code_from_file(run_name, path_name)
                  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "c:\Users\Vaibh\.vscode\extensions\ms-python.debugpy-2024.8.0-win32-x64\bundled\libs\debugpy\_vendored\pydevd\_pydevd_bundle\pydevd_runpy.py", line 294, in _ge
t_code_from_file
    code = compile(f.read(), fname, 'exec')
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\Vaibh\import random.py", line 24
    def originalURL[shortUEL]:
  File "C:\Users\Vaibh\import random.py", line 24
    def originalURL[shortUEL]:
    def originalURL[shortUEL]:
                   ^
                   ^

SyntaxError: expected '('
PS C:\Users\Vaibh> ("www.i am learningpython)
```

This lengthy URL gets passed as input to the `shorten()` function as input. The compact URL is then shown using the `print()` command. If you execute this script, the result is the abbreviated URL obtained from the link shortening API for the input `long_url`.

Every time you execute the program, the resulting condensed URL will be distinct. That is because this is based on the reply from the link shortener API. The abbreviated URL can be employed to lead users to the primary extended URL. This enables it simpler to exchange and retain.

Project -3

Introduction:

What’s a better time to simulate the spread of a disease than during a global pandemic? I don’t have much more to say — let’s jump right into programming a simple disease simulation.

In real life, there are hundreds of factors that affect how fast a contagion spreads, both from person to person and on a broader population-wide scale. I’m no epidemiologist but I’ve done my best to set up a fairly basic simulation that can mimic how a virus can infect people and spread throughout a population.

In my program, I will be using object-based programming. With this method, we could theoretically customize individual people and add in more events and factors — such as more complicated social dynamics.

Keep in mind that this is an introduction and serves as the most basic model that can be built on top of.

Variables/Explanation:

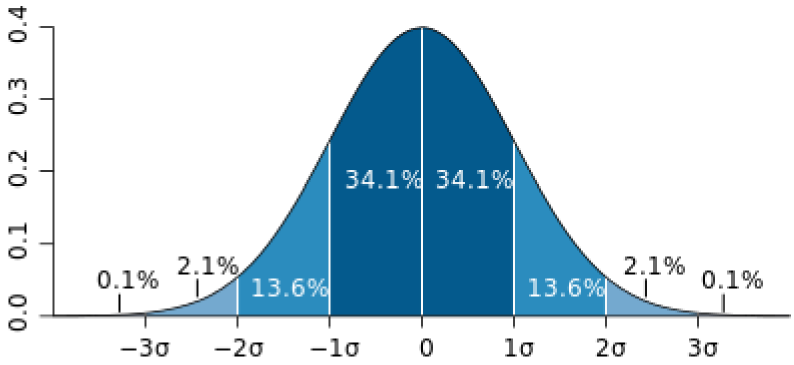
Fundamentally, our program will function around a single concept: any given person who is infected by our simulation’s disease has the potential to spread it to whoever they meet. Each person in our “peopleDictionary” will have a set **number of friends** (gaussian randomization for accuracy) and they may meet any one or more of these friends on a day to day basis.

For our starting round of simulations, we won’t implement face masks or lockdowns — we’ll just let the virus spread when people meet their friends and see if we can get that iconic pandemic “curve” which the news always talks about flattening.

So, we’ll use a Person() class and add a few characteristics. Firstly, we’ll assume that some very tiny percentage of characters simulated will already have **immunity** to our disease from the get-go, for whatever reason. I’m setting that at 1% (in reality, it’d be far lower but because our simulation runs so fast, a large portion like this makes a bit more sense). At the start of the simulation, the user will be prompted to enter this percentage.

Next, we have **contagiousness**, the all-important factor. When a person is not infected, this remains at 0. It also returns to 0 once a person ceases to be contagious and gains immunity. However, when a person is infected, this contagious value is somewhere between 0 and 100%, and it massively changes their chance of infecting a friend.

Before we implement this factor, we need to understand Gaussian Distribution. This mathematical function allows us to more accurately calculate random values between 1 and 100. Rather than the values being distributed purely randomly across the spectrum, most of them cluster around the median average point, making for a more realistic output:



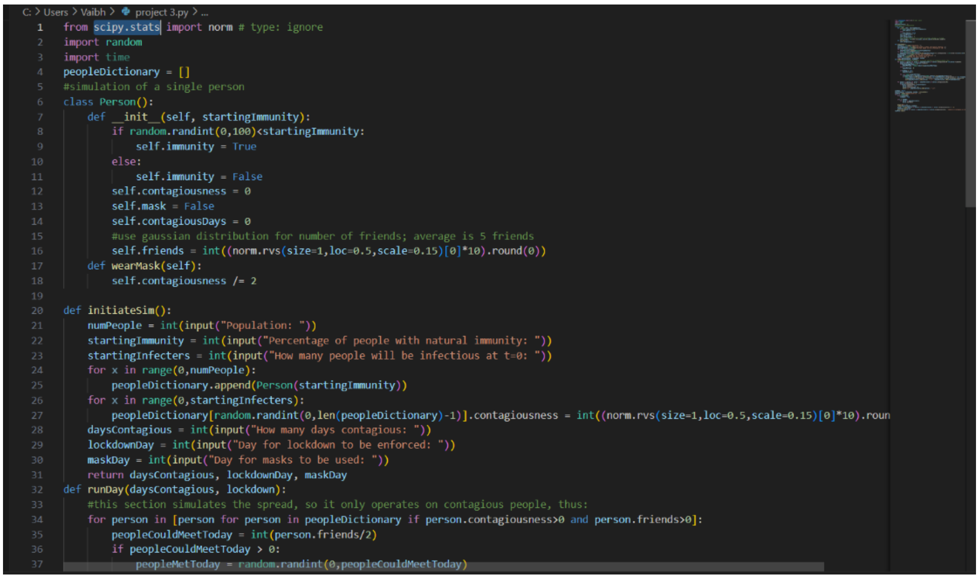
As you can see, this bell-shaped function will be a lot better for our random characteristic variables because most people will have an average level of contagiousness, rather than a purely random percentage. I'll show you how to implement this later.

We then have the variables "**mask**" and "**lockdown**" which are both boolean variables. These will be used to add a little bit of variety to our simulation after it is running.

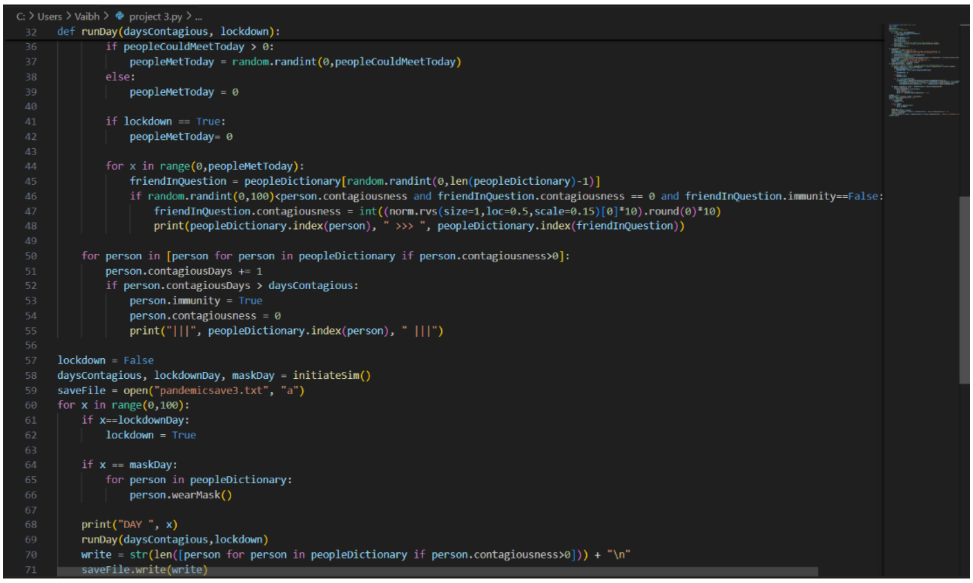
Lastly, we have the "**friends**" variable for any given person. Just like contagiousness, this is a Gaussian Distribution that ends up with most people having about 5 friends that they regularly see. In our simulation, everyone lives in a super social society where on average a person meets with 2 people face to face every day. In real life, this is probably not as realistic but we're using it because we don't want a super slow simulation. Of course, you can make any modifications to the code that you like.

There are also a couple of other variables that will be used actively in the simulation and I'll get to those as we go on!

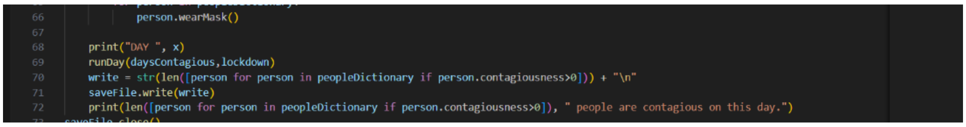
Program 3:



```python
from scipy.stats import norm # type: ignore
import random
import time
peopleDictionary = []
#simulation of a single person
class Person():
    def __init__(self, startingImmunity):
        if random.randint(0,100)<startingImmunity:
            self.immunity = True
        else:
            self.immunity = False
        self.contagiousness = 0
        self.mask = False
        self.contagiousDays = 0
        #use gaussian distribution for number of friends; average is 5 friends
        self.friends = int((norm.rvs(size=1,loc=0.5,scale=0.15)[0]*10).round(0))
    def wearMask(self):
        self.contagiousness /= 2

def initiateSim():
    numPeople = int(input("Population: "))
    startingImmunity = int(input("Percentage of people with natural immunity: "))
    startingInfecters = int(input("How many people will be infectious at t=0: "))
    for x in range(0,numPeople):
        peopleDictionary.append(Person(startingImmunity))
    for x in range(0,startingInfecters):
        peopleDictionary[random.randint(0,len(peopleDictionary)-1)].contagiousness = int((norm.rvs(size=1,loc=0.5,scale=0.15)[0]*10).roun
    daysContagious = int(input("How many days contagious: "))
    lockdownDay = int(input("Day for lockdown to be enforced: "))
    maskDay = int(input("Day for masks to be used: "))
    return daysContagious, lockdownDay, maskDay
def runDay(daysContagious, lockdown):
    #this section simulates the spread, so it only operates on contagious people, thus:
    for person in [person for person in peopleDictionary if person.contagiousness>0 and person.friends>0]:
        peopleCouldMeetToday = int(person.friends/2)
        if peopleCouldMeetToday > 0:
            peopleMetToday = random.randint(0,peopleCouldMeetToday)
```

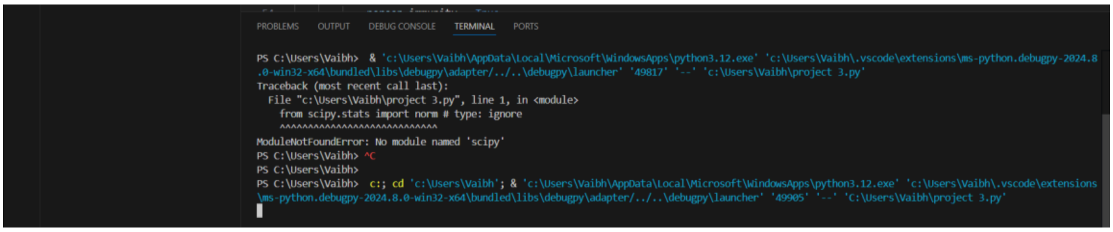


```python
def runDay(daysContagious, lockdown):
        if peopleCouldMeetToday > 0:
            peopleMetToday = random.randint(0,peopleCouldMeetToday)
        else:
            peopleMetToday = 0

        if lockdown == True:
            peopleMetToday= 0

        for x in range(0,peopleMetToday):
            friendInQuestion = peopleDictionary[random.randint(0,len(peopleDictionary)-1)]
            if random.randint(0,100)<person.contagiousness and friendInQuestion.contagiousness == 0 and friendInQuestion.immunity==False:
                friendInQuestion.contagiousness = int((norm.rvs(size=1,loc=0.5,scale=0.15)[0]*10).round(0)*10)
                print(peopleDictionary.index(person), " >>> ", peopleDictionary.index(friendInQuestion))

    for person in [person for person in peopleDictionary if person.contagiousness>0]:
        person.contagiousDays += 1
        if person.contagiousDays > daysContagious:
            person.immunity = True
            person.contagiousness = 0
            print("|||", peopleDictionary.index(person), " |||")

lockdown = False
daysContagious, lockdownDay, maskDay = initiateSim()
saveFile = open("pandemicsave3.txt", "a")
for x in range(0,100):
    if x==lockdownDay:
        lockdown = True

    if x == maskDay:
        for person in peopleDictionary:
            person.wearMask()

    print("DAY ", x)
    runDay(daysContagious,lockdown)
    write = str(len([person for person in peopleDictionary if person.contagiousness>0])) + "\n"
    saveFile.write(write)
```



```python
            person.wearMask()

    print("DAY ", x)
    runDay(daysContagious,lockdown)
    write = str(len([person for person in peopleDictionary if person.contagiousness>0])) + "\n"
    saveFile.write(write)
    print(len([person for person in peopleDictionary if person.contagiousness>0]), " people are contagious on this day.")
```

Output:



```
PS C:\Users\Vaibh>  & 'c:\Users\Vaibh\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\Vaibh\.vscode\extensions\ms-python.debugpy-2024.8
.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '49817' '--' 'c:\Users\Vaibh\project 3.py'
Traceback (most recent call last):
  File "c:\Users\Vaibh\project 3.py", line 1, in <module>
    from scipy.stats import norm # type: ignore
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^
ModuleNotFoundError: No module named 'scipy'
PS C:\Users\Vaibh> ^C
PS C:\Users\Vaibh>
PS C:\Users\Vaibh>  c:; cd 'c:\Users\Vaibh'; & 'c:\Users\Vaibh\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\Vaibh\.vscode\extensions
\ms-python.debugpy-2024.8.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '49905' '--' 'C:\Users\Vaibh\project 3.py'
```

Just to clarify, this was supposed to be a simple simulation. It is, of course, very basic with very limited parameters and functionality. However, it is incredible to see how much we can learn from a simulation that takes up barely a hundred lines of code. It really puts into perspective the impact lockdowns and face masks had.

Conclusion

Additional Resources in Python

Congratulations! You have successfully completed the beginner's course on Python. You should now be comfortable enough to ***write intermediate level programs*** in it. However, I would suggest a few benefits from this course.

Note that these suggestions are from the amount of knowledge I have. There might be a lot more options to explore. You should proceed according to your interests.

We will focus on a few important aspects only. I hope you can use it in a better way after reading this.

Why you should continue Python?

Usually, after learning a language, I find most students moving into another programming language. This is usually because one prefers to showcase their technical skills. Mostly they would prefer saying that they know 20+ languages. My advice: ***Stick to one language and excel in it.***

This can be of immense benefit. And basically in python, the future benefits can be as:

1. Python is ***easily readable and maintainable*** across a wide chain of developers. It is so because it has kicked one step higher to the programming paradigm. The code which is written ***resembles plain English*** and can be ***understood even by beginner programmers***. It also helps to ***update the code easily from time to time and invest less effort in doing so.***

2. Python supports ***both function-oriented and structure-oriented programming.***

3. It has features of ***dynamic memory management*** which can make use of computational resources efficiently.

4. It is also compatible with all popular operating systems and platforms. Hence this language can be ***universally accepted by all programmers***.

5. Python supports a ***large built-in library*** from which we can extract any feature to implement in the form of packages. Thus it enables us to ***implement a feature without writing excess code***.

Job Opportunities in Python

Programming language can be a key in your hand for your perfect dream job in the software industry. In today's market, Python has been declared as the most used and ***most demanding programming language*** of all. As the language is getting dominant in all major fields: Ranging from software development to machine learning and data analytics, Python has been declared as the ***language of the year 2018***. Hence It currently occupies ***37 percent*** of Programming language market.

Few Job opportunities in python include:

- ***Python Developer*** (Web-based or App-based)
- Product Manager (In Building a Software framework)

- ***Data Analyst.***
- Data Journalist.
- ***Educator*** at popular MOOC sites.

Due to the heavy demand for this language, there is a lot of job opportunities open ***even for students and fresh graduates***. Students can get a job as an intern with a decent salary during the study period and can widen their programming base by just learning python. Few popular sites like ***HackerRank*** and ***HackerEarth*** actively post coding competitions which use ***Python as their primary language***.

Overall, it is completely on your part how much you can learn. Being a vast language, it can sometimes be confusing on many parts but overall it is a ***huge fun*** learning this language.

Refrence

- Video studio code
- www.javapoint.com
- www.eshiksha.com
- www.Pythonorg.com
- www.GitHub.com