PASSWORD GENERATOR

By,

Ramsudhan G

Suriyanandan V

Certificate

Acknowledgement

First of all I express deep sense of gratitude to almighty God for giving me strength for the successful completion of the project.

I would like to express a deep sense of thanks & gratitude to my computer science teacher…….. for guiding me immensely through the course of project. She always evinced keen interest in my work. Her constructive advice & constant motivation has been responsible for the successful completion of this project .

I express my sincere thanks to the The Principal, for constant encouragement and the guidance provided during this project.

I also thank to my parents for their motivation and support. I must thank to my team members as well.

Last but not least; I would like to thank all those who had supported me directly and indirectly in any manner for completion of this project.

Introduction

• Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages

Guido Van Rossum conceived Python in the late 1980s. It was released in 1991 at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to the ABC language.

• Interpreted Language: Python is processed at runtime by Python Interpreter.

• Object-Oriented Language: It supports object-oriented features and techniques of programming.

• Interactive Programming Language: Users can interact with the python interpreter directly for writing programs.

• Easy language: Python is easy to learn, especially for beginners.

• Straightforward Syntax: The formation of python syntax is simple and straightforward, which also makes it popular.

• Easy to read: Python source-code is clearly defined and visible to the eyes.

• Portable: Python codes can be run on a wide variety of hardware platforms having the same interface.

• Extendable: Users can add low level-modules to Python interpreter.

• Scalable: Python provides an improved structure for supporting large programs then shell-scripts

What You Can Do with Python:

Python is used to create web and desktop applications, and some of the most popular web applications like Instagram, YouTube, Spotify all have been developed in Python. You can also develop the next big thing by using Python.

About Project

- ✓ PasswordGenerator is developed in Python programming language.
- ✓ This project is mainly used to generate lists of passwords with the given details of the victim.
- ✓ Python File “passwordgenerator.py” is of size 16 kilobytes.
- ✓ The Source code length of this project is 527 lines.
- ✓ The Generated list of passwords are stored in “<Firstname>_pass.txt” file.
- ✓ User can save the victim’s details in “details.txt” for future use.
- ✓ Both files “<Firstname>_pass.txt” and “details.txt” is created in the same directory where “passwordgenerator.py” is saved.

Source Code

#code

```
import os

def to_file(filename, unique_list_finished):
    f = open(filename, "w")
    unique_list_finished.sort()
    f.write(os.linesep.join(unique_list_finished))
    f.close()
    f = open(filename, "r")
    lines = 0
    for line in f:
        lines += 1
    f.close()
    print(
        "[+] Saving dictionary to \033[1;31m"
        + filename
        + "\033[1;m, counting \033[1;31m"
        + str(lines)
        + " words.\033[1;m"
    )
    print(
        "[+] Now load your pistolero with \033[1;31m"
```

```
        + filename
        + "\033[1;m and shoot! Good luck!"
    )

# String concats

def concats(seq, start, stop):
    for mystr in seq:
        for num in range(start, stop):
            yield mystr + str(num)

# For sorting and making combinations...

def komb(seq, start, special=""):
    for mystr in seq:
        for mystr1 in start:
            yield mystr + special + mystr1

def make_leet(x):
    """convert string to leet"""
    l=[
      ['a', '4'],
      ['i', '1'],
      ['e', '3'],
```

```
        ['t', '7'],
        ['o', '0'],
        ['s', '5'],
        ['g', '9'],
        ['z', '2']
       ]
    for letter in l:
        x = x.replace(letter[0],letter[1])
    return x


def generate(profile):
    """ Generates a wordlist from a given profile """

    chars = ['!','@','#','$','%%','&','*','^','?']
    years = [
        '2000','2001','2002','2003','2004',
        '2005','2006','2007','2008','2009',
        '2010','2011','2012','2013','2014',
        '2015','2016','2017','2018','2019',
        '2020'
        ]
    numfrom = 0
    numto = 100
```

```
profile["spechars"] = []

if profile["spechars1"] == "y":
    for spec1 in chars:
        profile["spechars"].append(spec1)
        for spec2 in chars:
            profile["spechars"].append(spec1 + spec2)
            for spec3 in chars:
                profile["spechars"].append(spec1 + spec2 + spec3)

print("\r\n[+] Now making a dictionary...")

# Now me must do some string modifications...

# Birthdays first

birthdate_yy = profile["birthdate"][-2:]
birthdate_yyy = profile["birthdate"][-3:]
birthdate_yyyy = profile["birthdate"][-4:]
birthdate_xd = profile["birthdate"][1:2]
birthdate_xm = profile["birthdate"][3:4]
birthdate_dd = profile["birthdate"][:2]
birthdate_mm = profile["birthdate"][2:4]
```

```
wifeb_yy = profile["wifeb"][-2:]

wifeb_yyy = profile["wifeb"][-3:]

wifeb_yyyy = profile["wifeb"][-4:]

wifeb_xd = profile["wifeb"][1:2]

wifeb_xm = profile["wifeb"][3:4]

wifeb_dd = profile["wifeb"][:2]

wifeb_mm = profile["wifeb"][2:4]


kidb_yy = profile["kidb"][-2:]

kidb_yyy = profile["kidb"][-3:]

kidb_yyyy = profile["kidb"][-4:]

kidb_xd = profile["kidb"][1:2]

kidb_xm = profile["kidb"][3:4]

kidb_dd = profile["kidb"][:2]

kidb_mm = profile["kidb"][2:4]


# Convert first letters to uppercase...


nameup = profile["name"].title()

surnameup = profile["surname"].title()

nickup = profile["nick"].title()

wifeup = profile["wife"].title()

wifenup = profile["wifen"].title()
```

```
kidup = profile["kid"].title()

kidnup = profile["kidn"].title()

petup = profile["pet"].title()

companyup = profile["company"].title()


wordsup = []

wordsup = list(map(str.title, profile["words"]))


word = profile["words"] + wordsup


# reverse a name


rev_name = profile["name"][::-1]

rev_nameup = nameup[::-1]

rev_nick = profile["nick"][::-1]

rev_nickup = nickup[::-1]

rev_wife = profile["wife"][::-1]

rev_wifeup = wifeup[::-1]

rev_kid = profile["kid"][::-1]

rev_kidup = kidup[::-1]


reverse = [

   rev_name,

   rev_nameup,
```

```
        rev_nick,
        rev_nickup,
        rev_wife,
        rev_wifeup,
        rev_kid,
        rev_kidup,
    ]
    rev_n = [rev_name, rev_nameup, rev_nick, rev_nickup]
    rev_w = [rev_wife, rev_wifeup]
    rev_k = [rev_kid, rev_kidup]
    # some serious work! but... who cares? :)

#<------------------------------------------------------------------------>
    # Birthdays combinations

    bds = [
        birthdate_yy,
        birthdate_yyy,
        birthdate_yyyy,
        birthdate_xd,
        birthdate_xm,
        birthdate_dd,
        birthdate_mm,
    ]
```

```
bdss = []


for bds1 in bds:
    bdss.append(bds1)
    for bds2 in bds:
        if bds.index(bds1) != bds.index(bds2):
            bdss.append(bds1 + bds2)
            for bds3 in bds:
                if (
                    bds.index(bds1) != bds.index(bds2)
                    and bds.index(bds2) != bds.index(bds3)
                    and bds.index(bds1) != bds.index(bds3)
                ):
                    bdss.append(bds1 + bds2 + bds3)


            # For a woman...
wbds = [wifeb_yy, wifeb_yyy, wifeb_yyyy, wifeb_xd, wifeb_xm, wifeb_dd, wifeb_mm]


wbdss = []


for wbds1 in wbds:
    wbdss.append(wbds1)
    for wbds2 in wbds:
```

```
        if wbds.index(wbds1) != wbds.index(wbds2):
            wbdss.append(wbds1 + wbds2)
            for wbds3 in wbds:
                if (
                    wbds.index(wbds1) != wbds.index(wbds2)
                    and wbds.index(wbds2) != wbds.index(wbds3)
                    and wbds.index(wbds1) != wbds.index(wbds3)
                ):
                    wbdss.append(wbds1 + wbds2 + wbds3)

            # A child...
kbds = [kidb_yy, kidb_yyy, kidb_yyyy, kidb_xd, kidb_xm, kidb_dd, kidb_mm]

kbdss = []

for kbds1 in kbds:
    kbdss.append(kbds1)
    for kbds2 in kbds:
        if kbds.index(kbds1) != kbds.index(kbds2):
            kbdss.append(kbds1 + kbds2)
            for kbds3 in kbds:
                if (
                    kbds.index(kbds1) != kbds.index(kbds2)
                    and kbds.index(kbds2) != kbds.index(kbds3)
```

```
                and kbds.index(kbds1) != kbds.index(kbds3)
            ):
                kbdss.append(kbds1 + kbds2 + kbds3)

#<---------------------------------------------------------------------------->
            # string combinations....

    kombinaac = [
        profile["pet"],
        petup,
        profile["company"],
        companyup
    ]

    kombina = [
        profile["name"],
        profile["surname"],
        profile["nick"],
        nameup,
        surnameup,
        nickup,
    ]

    kombinaw = [
```

```
        profile["wife"],
        profile["wifen"],
        wifeup,
        wifenup,
        profile["surname"],
        surnameup,
    ]

    kombinak = [
        profile["kid"],
        profile["kidn"],
        kidup,
        kidnup,
        profile["surname"],
        surnameup,
    ]
#<---------------------------------------------------------------------------------------->
    kombinaa = []
    for kombina1 in kombina:
        kombinaa.append(kombina1)
        for kombina2 in kombina:
            if kombina.index(kombina1) != kombina.index(kombina2) and kombina.index(
                    kombina1.title()
            )!= kombina.index(kombina2.title()):
```

```
            kombinaa.append(kombina1 + kombina2)


    kombinaaw = []
    for kombina1 in kombinaw:
        kombinaaw.append(kombina1)
        for kombina2 in kombinaw:
            if kombinaw.index(kombina1) != kombinaw.index(kombina2) and kombinaw.index(
                kombina1.title()
            ) != kombinaw.index(kombina2.title()):
                kombinaaw.append(kombina1 + kombina2)


    kombinaak = []
    for kombina1 in kombinak:
        kombinaak.append(kombina1)
        for kombina2 in kombinak:
            if kombinak.index(kombina1) != kombinak.index(kombina2) and kombinak.index(
                kombina1.title()
            ) != kombinak.index(kombina2.title()):
                kombinaak.append(kombina1 + kombina2)

#<-------------------------------------------------------------------------------------->


    kombi = {}
    kombi[1] = list(komb(kombinaa, bdss))
```

```
kombi[1] += list(komb(kombinaa, bdss, "_"))

kombi[2] = list(komb(kombinaaw, wbdss))

kombi[2] += list(komb(kombinaaw, wbdss, "_"))

kombi[3] = list(komb(kombinaak, kbdss))

kombi[3] += list(komb(kombinaak, kbdss, "_"))

kombi[4] = list(komb(kombinaa, years))

kombi[4] += list(komb(kombinaa, years, "_"))

kombi[5] = list(komb(kombinaac, years))

kombi[5] += list(komb(kombinaac, years, "_"))

kombi[6] = list(komb(kombinaaw, years))

kombi[6] += list(komb(kombinaaw, years, "_"))

kombi[7] = list(komb(kombinaak, years))

kombi[7] += list(komb(kombinaak, years, "_"))

kombi[8] = list(komb(word, bdss))

kombi[8] += list(komb(word, bdss, "_"))

kombi[9] = list(komb(word, wbdss))

kombi[9] += list(komb(word, wbdss, "_"))

kombi[10] = list(komb(word, kbdss))

kombi[10] += list(komb(word, kbdss, "_"))

kombi[11] = list(komb(word, years))

kombi[11] += list(komb(word, years, "_"))

kombi[12] = [""]

kombi[13] = [""]

kombi[14] = [""]
```

```
kombi[15] = [""]
kombi[16] = [""]
kombi[21] = [""]
if profile["randnum"] == "y":
    kombi[12] = list(concats(word, numfrom, numto))
    kombi[13] = list(concats(kombinaa, numfrom, numto))
    kombi[14] = list(concats(kombinaac, numfrom, numto))
    kombi[15] = list(concats(kombinaaw, numfrom, numto))
    kombi[16] = list(concats(kombinaak, numfrom, numto))
    kombi[21] = list(concats(reverse, numfrom, numto))
kombi[17] = list(komb(reverse, years))
kombi[17] += list(komb(reverse, years, "_"))
kombi[18] = list(komb(rev_w, wbdss))
kombi[18] += list(komb(rev_w, wbdss, "_"))
kombi[19] = list(komb(rev_k, kbdss))
kombi[19] += list(komb(rev_k, kbdss, "_"))
kombi[20] = list(komb(rev_n, bdss))
kombi[20] += list(komb(rev_n, bdss, "_"))
komb001 = [""]
komb002 = [""]
komb003 = [""]
komb004 = [""]
komb005 = [""]
komb006 = [""]
```

```
    if len(profile["spechars"]) > 0:
        komb001 = list(komb(kombinaa, profile["spechars"]))
        komb002 = list(komb(kombinaac, profile["spechars"]))
        komb003 = list(komb(kombinaaw, profile["spechars"]))
        komb004 = list(komb(kombinaak, profile["spechars"]))
        komb005 = list(komb(word, profile["spechars"]))
        komb006 = list(komb(reverse, profile["spechars"]))

#<---------------------------------------------------------------------------------->
    print("[+] Sorting list and removing duplicates...")

    komb_unique = {}
    for i in range(1, 22):
        komb_unique[i] = list(dict.fromkeys(kombi[i]).keys())

    komb_unique01 = list(dict.fromkeys(kombinaa).keys())
    komb_unique02 = list(dict.fromkeys(kombinaac).keys())
    komb_unique03 = list(dict.fromkeys(kombinaaw).keys())
    komb_unique04 = list(dict.fromkeys(kombinaak).keys())
    komb_unique05 = list(dict.fromkeys(word).keys())
    komb_unique07 = list(dict.fromkeys(komb001).keys())
    komb_unique08 = list(dict.fromkeys(komb002).keys())
    komb_unique09 = list(dict.fromkeys(komb003).keys())
    komb_unique010 = list(dict.fromkeys(komb004).keys())
```

```
komb_unique011 = list(dict.fromkeys(komb005).keys())

komb_unique012 = list(dict.fromkeys(komb006).keys())


uniqlist = (
    bdss
    + wbdss
    + kbdss
    + reverse
    + komb_unique01
    + komb_unique02
    + komb_unique03
    + komb_unique04
    + komb_unique05
)


for i in range(1, 21):
    uniqlist += komb_unique[i]


uniqlist += (
    komb_unique07
    + komb_unique08
    + komb_unique09
    + komb_unique010
    + komb_unique011
```

```
        + komb_unique012
    )
    unique_lista = list(dict.fromkeys(uniqlist).keys())
    unique_leet = []
    if profile["leetmode"] == "y":
        for x in unique_lista:
            x = make_leet(x)
            unique_leet.append(x)

    unique_list = unique_lista + unique_leet
#<---------------------------------------------------------------------------->
    unique_list_finished = []
    for i in unique_list:
        if len(i)>=4 and len(i)<=15:
            unique_list_finished.append(i)

    to_file(profile["name"]+'_pass.txt',unique_list_finished)



def details():
    print("\r\n[+] Insert the information about the victim to make a dictionary")
    print("[+] If you don't know all the info, just hit enter when asked! ;)\r\n")
```

```
profile={}

# First Name
name = input("> First Name: ").lower()
while len(name) == 0 or name.isspace():
    print("\r\n[-] You must enter a name at least!")
    name = input("> First Name: ").lower()
profile["name"]=str(name)

# Surnmar and Nickname
profile["surname"] = input("> Surname: ").lower()
profile["nick"] = input("> Nickname: ").lower()

# Birthday
birthdate = input("> Birthdate (DDMMYYYY): ")
while len(birthdate) != 0 and len(birthdate) != 8:
    print("\r\n[-] You must enter 8 digits for birthday!")
    birthdate = input("> Birthdate (DDMMYYYY): ")
profile["birthdate"] = str(birthdate)

print("\r\n")

# Partner
```

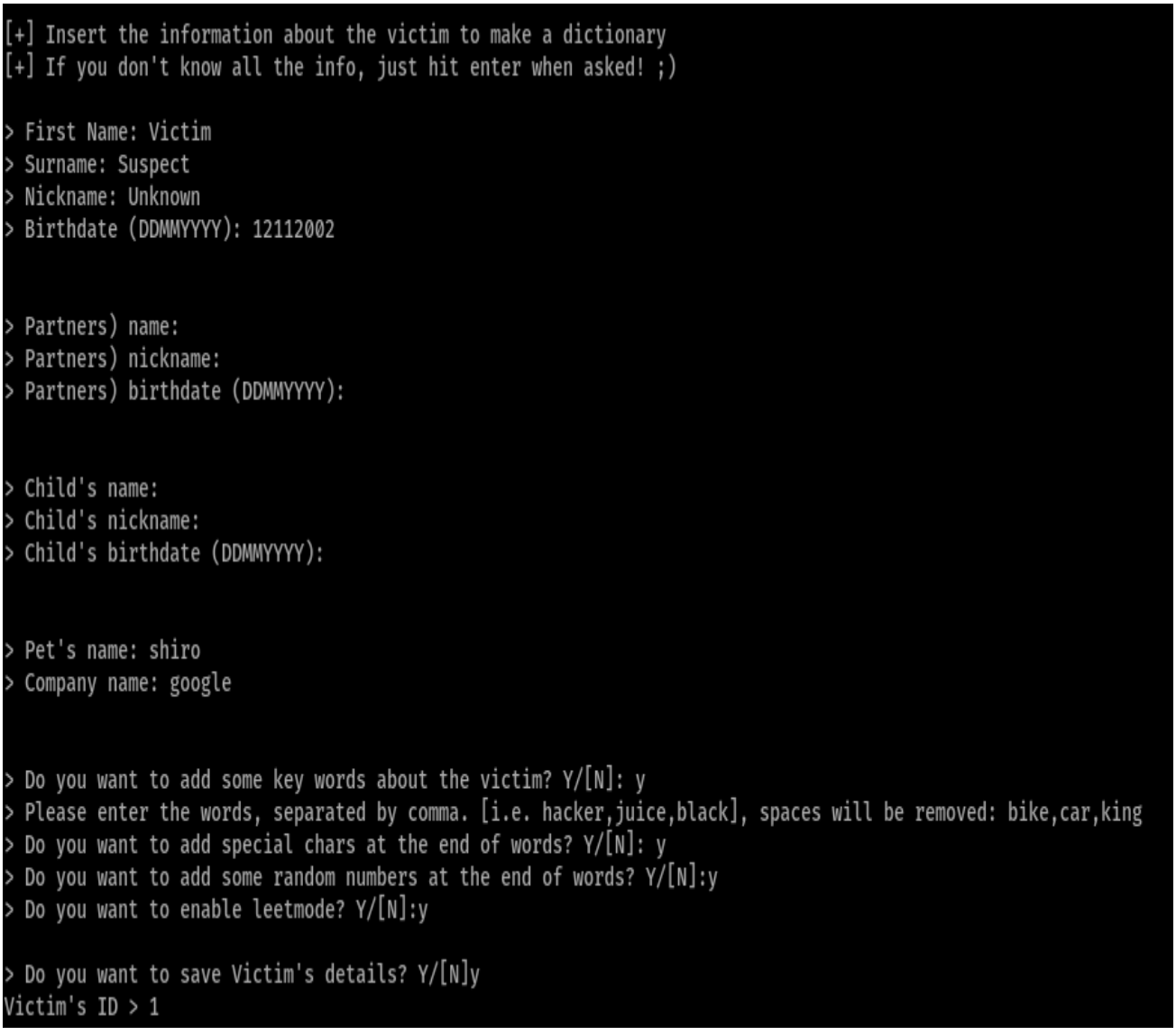
```
profile["wife"] = input("> Partners) name: ").lower()
profile["wifen"] = input("> Partners) nickname: ").lower()
wifeb = input("> Partners) birthdate (DDMMYYYY): ")
while len(wifeb) != 0 and len(wifeb) != 8:
    print("\r\n[-] You must enter 8 digits for birthday!")
    wifeb = input("> Partners birthdate (DDMMYYYY): ")
profile["wifeb"] = str(wifeb)

print("\r\n")

# Kids
profile["kid"] = input("> Child's name: ").lower()
profile["kidn"] = input("> Child's nickname: ").lower()
kidb = input("> Child's birthdate (DDMMYYYY): ")
while len(kidb) != 0 and len(kidb) != 8:
    print("\r\n[-] You must enter 8 digits for birthday!")
    kidb = input("> Child's birthdate (DDMMYYYY): ")
profile["kidb"] = str(kidb)

print("\r\n")

# Additional
profile["pet"] = input("> Pet's name: ").lower()
profile["company"] = input("> Company name: ").lower()
```

```
    print("\r\n")


    profile["words"] = [""]


    words1 = input(
        "> Do you want to add some key words about the victim? Y/[N]: "
    ).lower()


    words2 = ""
    if words1 == "y":
        words2 = input(
            "> Please enter the words, separated by comma. [i.e. hacker,juice,black], spaces will be
removed: "
        ).replace(" ", "")
    profile["words"] = words2.split(",")


    profile["spechars1"] = input(
        "> Do you want to add special chars at the end of words? Y/[N]: "
    ).lower()


    profile["randnum"] = input(
        "> Do you want to add some random numbers at the end of words? Y/[N]:"
    ).lower()
```

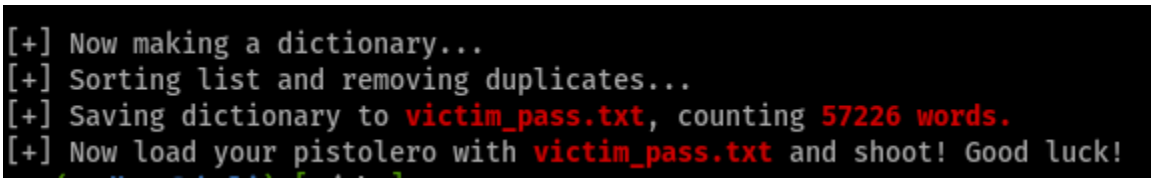
```
    profile["leetmode"] = input(
        "> Do you want to enable leetmode? Y/[N]:"
        )
    return profile

#<---------------------------------------------------------------------------->

def store():
    st=input("\r\n> Do you want to save Victim's details? Y/[N]").lower()
    if st=='y' or st.isspace() or st == 'yes':
        if os.path.exists("details.txt"):
            id=find()
            val=save(id)
            print("Victim's ID >",val)
        else:
            create=open("details.txt","w")
            create.close()
            save(0)
            print("Victim's ID >",1)

def find():
    re=open("details.txt","r")
    check=re.readlines()
    check=check[::-1]
```

```
    for i in range(len(check)):
        if check[i]=="ID\n":
            re.close()
            return int(check[i-1])
    return 0


def save(id):
    global proile
    prof=proile
    f=open("details.txt","a")
    temp=list([ i[1] for i in list(prof.items())])
    f.write("\r\n"+"ID"+"\n")
    f.write(str(id+1)+"\n\n")
    for i in temp:
        f.write(str(i)+"\n")
    f.write("\n<-------------------------------------------------------------------->\n")
    f.close()
    return id+1
#<--------------------------------------------------------------------------->
if __name__ == "__main__":
    proile=details()
    store()
    generate(proile)
#<--------------------------------------------------------------------------->
```

INPUT ScreensSnapshot:

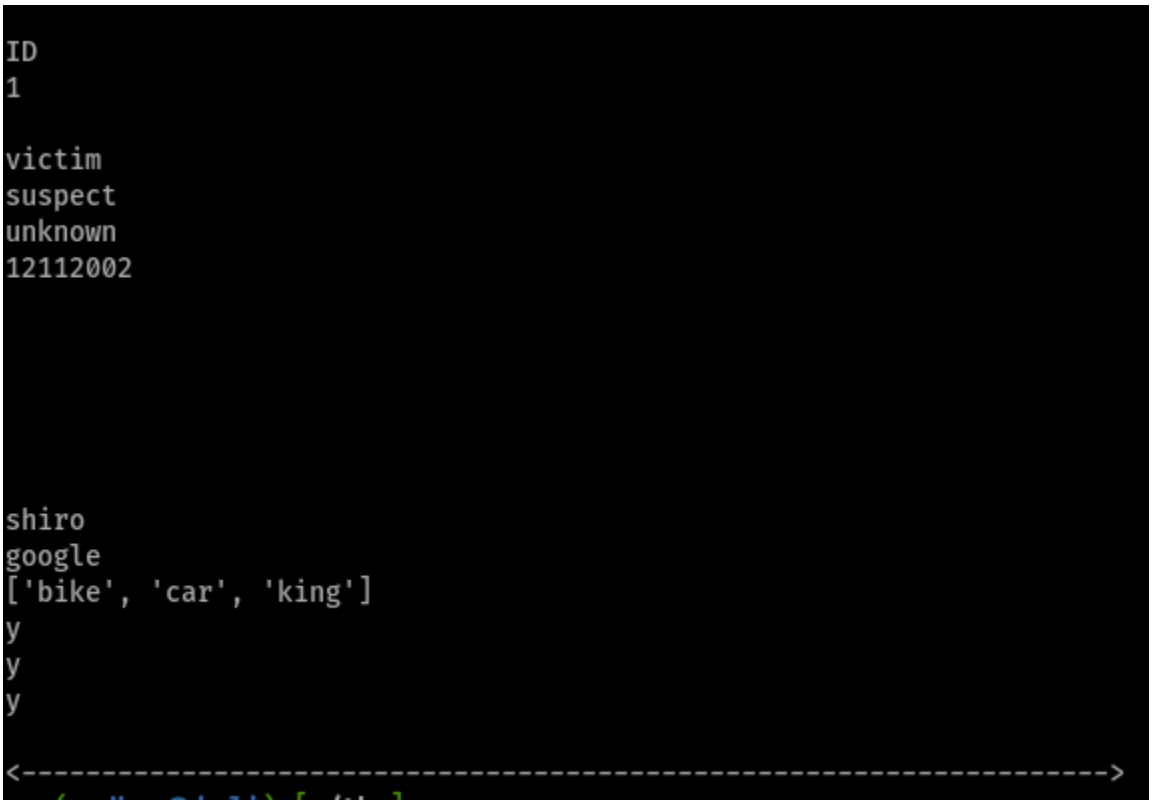


OUTPUT ScreensSnapshots:

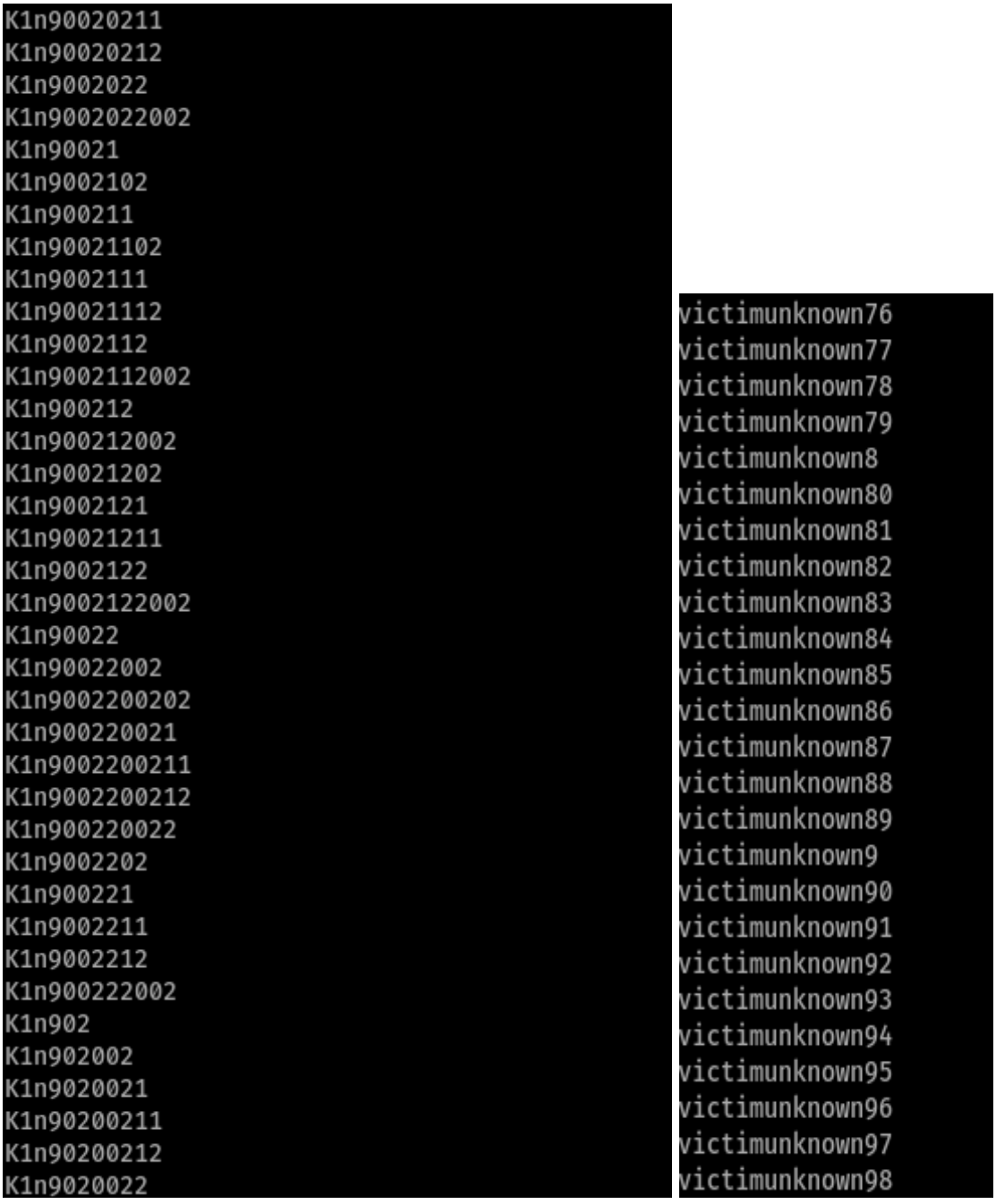
- 

- 

Sample output of "details.txt":

- 

Sample output of “victim_pass.txt”:



Conclusion

This project is User Friendly.

The Software and Hardware requirements are given below:

- Operating System : Windows 8 And Above,Linux,MacOs
- RAM : 512MB+ Hard Disk: SATA 40 GB or Above
- CD/DVD/PENDRIVE
- Python Libraries: os
- Monitor 14.1 or 15 -17 Inch Keyboard
- PYTHON IDLE 3.5 OR ABOVE
- Spyder

Python File “passwordgenerator.py” is executed successfully.

The generated password list can be used in brute force attack,to crack hash files and also to crack social media accounts like facebook, instagram,twitter,gmail,etc.

It can be used along with some tools like

- John The Ripper,
- Hydra,
- Hashcat,etc.

Bibliography

I have used the following reference to complete my project work.

- Online Classes by the teacher
- Books Referred:- Computer Sc. With Python by Preeti Arora
- Books Referred:- Computer Sc. With Python by Sumita Arora

Websites referred:

- https://www.google.co.in/
- https://stackoverflow.com/
- https://www.python.org/
- https://www.w3schools.com/python/
- https://github.com/C-YBERBOT/Investigative-Project