Chapter 1

first.py

```
print("Hello World")
```

module.py

```
import pyjokes


# print("Printing Jokes...")


# This prints a random joke
joke = pyjokes.get_joke()
print(joke)



# so thanks
# that was my program
# another line
# Yet another line
```

Chapter 1 - PS

Problem1.py

```
print(''' Twinkle, twinkle, little star,

How I wonder what you are!

Up above the world so high,

Like a diamond in the sky.


When the blazing sun is gone,

When he nothing shines upon,

Then you show your little light,

Twinkle, twinkle, all the night.


Then the trav'ller in the dark,

Thanks you for your tiny spark,

He could not see which way to go,

If you did not twinkle so.


In the dark blue sky you keep,

And often thro' my curtains peep,

For you never shut your eye,

Till the sun is in the sky.


'Tis your bright and tiny spark,

Lights the trav'ller in the dark:

Tho' I know not what you are,

Twinkle, twinkle, little star.''')
```

Problem2.py

```
# Done using REPL
```

Problem3.py

```
import pyttsx3

engine = pyttsx3.init()

engine.say("Hey I am good")

engine.runAndWait()
```

Problem4.py

```
import os


# Select the directory whose content you want to list

directory_path = '/'


# Use the os module to list the directory content

contents = os.listdir(directory_path)


# Print the contents of the directory

print(contents)
```

Chapter 2

01_variables.py

```
a = 1


b = 2


c = 7


name = "harry"


print(a + b)
```

02_datatypes.py

```
a = 1 # a is an integer


b = 5.22 # b is a floating point number


c = "Harry" # c is a string


d = False # d is a boolean variable


e = None # e is a none type variable
```

03_rules_variables.py

```
a = 23


aaa = 435
```

```
harry = 34


sameer = 45


_samerr = 34


# @sameer = 56 # Invalid due to @ symbol

# s@meer # Invalid due to @ symbol
```

04_operators.py

```
# Arithmetic Operators

a = 7

b = 4

c = a + b

print(c)


# Assignment Operators

a = 4-2 # Assign 4-2 in a

print(a)

b = 6

# b += 3 # Increment the value of b by 3 and then assign it to b

b -= 3 # Decrement the value of b by 3 and then assign it to b

print(b)


# Comparison Operators
```

```
d = 5==5

print(d)



# Logical Operators

e = True or False

# Truth table of 'or'
print("True or False is ", True or False)
print("True or True is ", True or True)
print("False or True is ", False or True)
print("False or False is ", False or False)

# Truth table of 'and'
print("True and False is ", True and False)
print("True and True is ", True and True)
print("False and True is ", False and True)
print("False and False is ", False and False)

print(not(True))
```

05_type.py

```
a = "31.2"
b = float(a) # a but the type should be float
t = type(b)
```

```
print(t)
```

06_input.py

```
a = int(input("Enter number 1: "))


b = int(input("Enter number 2: "))


print("Number a is: ", a)

print("Number b is: ", b)

print("Sum is ", a + b)
```

Chapter 2 - PS

01_problem1.py

```
a = 1


b = 5


print(a + b)
```

02_problem2.py

```
a = 37


b = 5


print("Remainder when a is divided by b is", a % b)
```

03_problem3.py

```
a = input("Enter the value of a: ")
print(type(a))
```

04_problem4.py

```
a = int(input("Enter number 1: "))
b = int(input("Enter number 2: "))


print("a is greater than b is", a>b)
```

05_problem5.py

```
a = int(input("Enter number 1: "))
b = int(input("Enter number 2: "))
```

```
print("The average of these two number is", (a+b)/2)
```

06_problem6.py

```
a = int(input("Enter your number: "))




print("The square of the number is", a**2)

print("The square of the number is", a*a)

# print("The square of the number is", a^2) # Incorrect for finding square of a number in Python
```

Chapter 3

01_intro_to_strings.py

```
name = "Harry"


nameshort = name[0:3] # start from index 0 all the way till 3 (excluding 3)
print(nameshort)
character1 = name[1]
print(character1)
```

02_negative_slicing.py

```
name = "Harry"


print(name[0:3])


print(name[-4: -1])
print(name[1:4])


print(name[:4]) # is same as print(name[0:4])
print(name[1:]) # is same as print(name[1:5])
print(name[1:5])
```

03_str_functions.py

```
name = "harry"


print(len(name))
print(name.endswith("rry"))
print(name.startswith("ha"))
```

```
print(name.capitalize())
```

04_escape_seq.py

```
a = 'Harry is a good boy\nbut not a bad \'boy\"



print(a)
```

Chapter 3 - PS

01_problem1.py

```
name = input("Enter your name: ")


print(f"Good Afternoon, {name} ")
```

02_problem2.py

```
letter = "'Dear <|Name|>,
You are selected!
<|Date|> "'


print(letter.replace("<|Name|>", "Harry").replace("<|Date|", "24 September 2050"))
```

03_problem3.py

```
name = "Harry is a good  boy and  "


print(name.find("  "))
```

04_problem4.py

```
name = "Harry is a good  boy and  "


print(name.replace("  ", " "))
print(name) # Strings are immutable which means that you cannot change them by running functions on them
```

05_problem5.py

```
letter = "Dear Harry,\n\tThis python course is nice.\nThanks!"
```

```
print(letter)
```

Chapter 4

01_list.py

```
friends = ["Apple", "Orange", 5, 345.06, False, "Aakash", "Rohan"]


print(friends[0])
friends[0] = "Grapes" # Unlike Strings lists are mutable


print(friends[0])
print(friends[1:4])
```

02_list_methods.py

```
friends = ["Apple", "Orange", 5, 345.06, False, "Aakash", "Rohan"]
print(friends)
friends.append("Harry")
print(friends)


l1 = [1, 34,62, 2, 6, 11]
# l1.sort()
# l1.reverse()
# l1.insert(2, 333333) # Insert 333333 such that its index in the list is 3
value = l1.pop(3)
print(value)
print(l1)
```

03_tuple.py

```
a = (1,45,342,3424,False, "Rohan", "Shivam")
print(a)
```

```
print(type(a))
```

04_tuple_methods.py

```
a = (1,45,342,3424,False, 45, "Rohan", "Shivam")

print(a)


no = a.count(45)

print(no)


i = a.index(3424)

print(i)


print(len(a))
```

Chapter 4 - PS

01_problem1.py

```
fruits = []


f1 = input("Enter Fruit name: ")
fruits.append(f1)
f2 = input("Enter Fruit name: ")
fruits.append(f2)
f3 = input("Enter Fruit name: ")
fruits.append(f3)
f4 = input("Enter Fruit name: ")
fruits.append(f4)
f5 = input("Enter Fruit name: ")
fruits.append(f5)
f6 = input("Enter Fruit name: ")
fruits.append(f6)
f7 = input("Enter Fruit name: ")
fruits.append(f7)


print(fruits)
```

02_problem2.py

```
marks = []


f1 = int(input("Enter Marks here: "))
marks.append(f1)
```

```
f2 = int(input("Enter Marks here: "))
marks.append(f2)
f3 = int(input("Enter Marks here: "))
marks.append(f3)
f4 = int(input("Enter Marks here: "))
marks.append(f4)
f5 = int(input("Enter Marks here: "))
marks.append(f5)
f6 = int(input("Enter Marks here: "))
marks.append(f6)


marks.sort()


print(marks)
```

03_problem3.py

```
a = (34, 234, "Harry")


a[2] = "Larry"
```

04_problem4.py

```
l = [3, 3, 5, 1]



print(sum(l))
```

05_problem5.py

```
a = (7, 0, 8, 0, 0, 9)
```

```
n = a.count(0)

print(n)
```

Chapter 5

01_dict.py

```
d = {} # Empty dictionary
marks = {
    "Harry": 100,
    "Shubham": 56,
    "Rohan": 23
}



# print(marks, type(marks))
print(marks["Harry"])
```

02_dict_methods.py

```
marks = {
    "Harry": 100,
    "Shubham": 56,
    "Rohan": 23,
    0: "Harry"
}


# print(marks.items())
# print(marks.keys())
# print(marks.values())
# marks.update({"Harry": 99, "Renuka": 100})
# print(marks)
```

```
print(marks.get("Harry2")) # Prints None

print(marks["Harry2"]) # Returns an error
```

03_sets.py

```
e = set() # Dont use s = {} as it will create an empty dictionary

s = {1, 5, 32, 54,5, 5, 5}



print(s)
```

04_set_methods.py

```
s = {1, 5, 32, 54,5, 5, 5, "Harry"}


print(s, type(s))


s.add(566)

print(s, type(s))

s.remove(1)

print(s, type(s))
```

05_set_union_intersection.py

```
s1 = {1, 45, 6, 78}

s2 = {7, 8, 1, 78}


print(s1.union(s2))

print(s1.intersection(s2))
```

Chapter 5 - PS

01_problem1.py

```
words = {
    "madad": "Help",
    "kursi": "Chair",
    "billi": "Cat"
}


word = input("Enter the word you want meaning of: ")


print(words[word])
```

02_problem2.py

```
s = set()
n = input("Enter number: ")
s.add(int(n))
n = input("Enter number: ")
s.add(int(n))
n = input("Enter number: ")
s.add(int(n))
n = input("Enter number: ")
s.add(int(n))
n = input("Enter number: ")
s.add(int(n))
n = input("Enter number: ")
s.add(int(n))
n = input("Enter number: ")
s.add(int(n))
```

```
n = input("Enter number: ")
s.add(int(n))
n = input("Enter number: ")
s.add(int(n))


print(s)
```

03_problem3.py

```
s = set()
s.add(18)
s.add("18")


print(s)
```

04_problem4.py

```
s = set()
s.add(20)
s.add(20.0)
s.add('20') # length of s after these operations?


print(len(s))
```

05_problem5.py

```
s = {}
print(type(s))
```

06_problem6.py

```
d = {}
```

```
name = input("Enter friends name: ")
lang = input("Enter Language name: ")
d.update({name: lang})

name = input("Enter friends name: ")
lang = input("Enter Language name: ")
d.update({name: lang})

name = input("Enter friends name: ")
lang = input("Enter Language name: ")
d.update({name: lang})

name = input("Enter friends name: ")
lang = input("Enter Language name: ")
d.update({name: lang})


print(d)
```

07_problem7.py

```
# The values entered later will be updated
```

08_problem8.py

```
# Nothing will happen. The values can be same
```

09_problem9.py

```
s = {8, 7, 12, "Harry", [1,2]}
```

s[4][0] = 9

Chapter 6

01_conditionals.py

```
a = int(input("Enter your age: "))


# If else statement
if(a>=18):
    print("You are above the age of consent")
    print("Good for you")

else:
    print("You are below the age of consent")



print("End of Program")
```

02_if_elif_else_ladder.py

```
a = int(input("Enter your age: "))


# If elif else ladder
if(a>=18):
    print("You are above the age of consent")
    print("Good for you")

elif(a<0):
    print("You are entering an invalid negative age")
```

```
elif(a==0):
    print("You are entering 0 which is not a valid age")

else:
    print("You are below the age of consent")


print("End of Program")
```

03_multiple_if_statements.py

```
a = int(input("Enter your age: "))

# If statement no: 1
if(a%2 == 0):
    print("a is even")
# End of If statement no: 1

# If statement no: 2
if(a>=18):
    print("You are above the age of consent")
    print("Good for you")

elif(a<0):
    print("You are entering an invalid negative age")
```

```
else:
    print("You are below the age of consent")
# End of If statement no: 2


print("End of Program")
```

Chapter 6 - PS

01_problem1.py

```
a1 = int(input("Enter number 1: "))

a2 = int(input("Enter number 2: "))

a3 = int(input("Enter number 3: "))

a4 = int(input("Enter number 4: "))


if(a1>a2 and a1>a3 and a1>a4):

    print("Greatest number is a1:", a1)


elif(a2>a1 and a2>a3 and a2>a4):

    print("Greatest number is a2:", a2)


elif(a3>a1 and a3>a2 and a3>a4):

    print("Greatest number is a3:", a3)


elif(a4>a1 and a4>a2 and a4>a3):

    print("Greatest number is a4:", a4)
```

02_problem2.py

```
marks1 = int(input("Enter Marks 1: "))

marks2 = int(input("Enter Marks 2: "))

marks3 = int(input("Enter Marks 3: "))


# Check for total percentage

total_percentage = (100*(marks1 + marks2 + marks3))/300
```

```
if(total_percentage>=40 and marks1>=33 and marks2>=33 and marks3>=33):
    print("You are passed:", total_percentage)

else:
    print("You failed, try again next year:", total_percentage)
```

03_problem3.py

```
p1 = "Make a lot of money"
p2 = "buy now"
p3 = "subscribe this"
p4 = "click this"

message = input("Enter your comment: ")

if((p1 in message) or (p2 in message )or (p3 in message) or (p4 in message)):
    print("This comment is a spam")

else:
    print("This comment is not a spam")
```

04_problem4.py

```
username = input("Enter username: ")

if(len(username)<10):
    print("Your username contains less than 10 characters")
else:
    print("Your username contains more than or equal to 10 characters")
```

05_problem5.py

```
l = ["Harry", "Rohan", "Shubham", "Divya"]


name = input("Enter your name: ")


if(name in l):
    print("Your name is in the list")
else:
    print("Your name is not in the list")
```

06_problem6.py

```
marks = int(input("Enter your marks: "))


if(marks<=100 and marks>=90):
    grade = "Ex"
elif(marks<90 and marks>=80):
    grade = "A"
elif(marks<80 and marks>=70):
    grade = "B"
elif(marks<70 and marks>=60):
    grade = "C"
elif(marks<60 and marks>=50):
    grade = "D"
elif(marks<50):
    grade = "F"


print("Your grade is:", grade)
```

07_problem7.py

```
post = input("Enter the post: ")




if("harry" in post.lower()):
    print("This post is talking about harry")


else:
    print("This post is not talking about harry")
```

Chapter 7

01_loops.py

```
print(1)

print(2)

print(3)

print(4)

print(5)


# The same task can be done like this:

for i in range(1, 6):

    print(i)
```

02_while_loops.py

```
i = 1


while(i<51):

    print(i)

    i +=1 # or i = i + 1

'''

Output:

1

2

3

4

5

'''
```

03_list_using_while.py

```
l = [1, "Harry", False, "This", "Rohan", "Shubham", "Shubhi"]


i = 0


while(i<len(l)):
    print(l[i])
    i +=1
```

04_for_loops.py

```
for i in range(4):
    print(i)
```

05_for_loop_iterate.py

```
## For Loop with Lists
l = [1, 4, 6, 234, 6, 764]
for i in l:
    print(i)

## For Loop with Tuples
t = (6, 231, 75, 122)
for i in t:
    print(i)

## For Loop with Strings
s = "Harry"
```

```
for i in s:
    print(i)
```

06_for_with_else.py

```
l= [1,7,8]


for item in l:
    print(item)


else:
    print("done") # this is printed when the loop exhausts!
```

07_break_and_continue.py

```
for i in range(100):
    if(i == 34):
        break # Exit the loop right now
    print(i)


for i in range(100):
    if(i == 34):
        continue # Skip this iteration
    print(i)
```

08_pass.py

```
for i in range(645):
    pass
```

```
i = 0
while(i<45):
    print(i)
    i += 1
```

Chapter 7 - PS

01_problem1.py

```
n = int(input("Enter a number: "))


for i in range(1, 11):
    print(f"{n} X {i} = {n * i}")
```

02_problem2.py

```
l = ["Harry", "Soham", "Sachin", "Rahul"]


for name in l:
    if(name.startswith("S")):
        print(f"Hello {name}")
```

03_problem3.py

```
n = int(input("Enter a number: "))


i = 1


while(i<11):
    print(f"{n} X {i} = {n * i}")
    i += 1
```

04_problem4.py

```
n = int(input("Enter a number: "))


for i in range(2, n):
```

```
    if(n%i) == 0:
        print("Number is not prime")
        break
else:
    print("Number is prime")
```

05_problem5.py

```
n = int(input("Enter the number: "))
i = 1
sum = 0
while(i<=n):
    sum += i
    i+=1

print(sum)
```

06_problem6.py

```
# 5! = 1 X 2 X 3 X 4 X 5

n = int(input("Enter the number: "))
product = 1
for i in range(1, n+1):
    product = product * i

print(f"The factorial of {n} is {product}")
```

07_problem7.py

```
'''
```

For n = 3

```
  *
 ***
*****
```

For n = 5

```
    *
   ***
  *****
 *******
*********
```

'''

```
n = int(input("Enter the number: "))
for i in range(1, n+1):
    print(" "* (n-i), end="")
    print("*"* (2*i-1), end="")
    print("")
```

08_problem8.py

```
n = int(input("Enter the number: "))
for i in range(1, n+1):
    print("*"* i, end="")
    print("")
```

09_problem9.py

'''

***

* * for n = 3

***

'''

```
n = int(input("Enter the number: "))
for i in range(1, n+1):
    if(i==1 or i==n):
        print("*"* n, end="")
    else:
        print("*", end="")
        print(" "* (n-2), end="")
        print("*", end="")
    print("")
```

10_problem10.py

```
n = int(input("Enter the number: "))

for i in range(1, 11):
    print(f"{n} X {11 -i} = {n*(11-i)}")
```

Chapter 8

01_functions.py

```
# a = int(input("Enter your number: "))
# b = int(input("Enter your number: "))
# c = int(input("Enter your number: "))

# average = (a + b + c)/3
# print(average)

# a = int(input("Enter your number: "))
# b = int(input("Enter your number: "))
# c = int(input("Enter your number: "))

# average = (a + b + c)/3
# print(average)

# Function Definition
def avg():
    a = int(input("Enter your number: "))
    b = int(input("Enter your number: "))
    c = int(input("Enter your number: "))

    average = (a + b + c)/3
    print(average)
```

```
avg() # Function Call
print("Thank you!")
avg()
print("Thank you!")
avg()
print("Thank you!")
avg()
avg()
```

02_quick_quiz.py

```
def goodDay():
    print("Good Day")

goodDay()
```

03_function_with_arguments.py

```
def goodDay(name, ending):
    print("Good Day, " + name)
    print(ending)
    return "ok"

a = goodDay("Harry", "Thank you")
print(a)
```

04_default_argument.py

```
def goodDay(name, ending="Thank you"):
    print(f"Good Day, {name}")
    print(ending)
```

```
goodDay("Harry", "Thanks")

goodDay("Rohan")
```

05_recursion.py

```
'''

factorial(0) = 1

factorial(1) = 1

factorial(2) = 2 X 1

factorial(3) = 3 X 2 X 1

factorial(4) = 4 X 3 X 2 X 1

factorial(5) = 5 X 4 X 3 X 2 X 1

factorial(n) = n X n-1 X.....3 X 2 X 1


factorial(n) = n * factorial(n-1)

'''


def factorial(n):
    if(n==1 or n==0):
        return 1
    return n * factorial(n-1)



n = int(input("Enter a number: "))

print(f"The factorial of this number is: {factorial(n)}")
```

Chapter 8 - PS

01_problem1.py

```
def greatest(a, b, c):
    if(a>b and a>c):
        return a
    elif(b>a and b>c):
        return b
    elif(c>b and c>a):
        return c


a = 1
b = 23
c = 3


print(greatest(a, b, c))
```

02_problem2.py

```
def f_to_c(f):
    return 5*(f-32)/9


f = int(input("Enter temperature in F: "))
c = f_to_c(f)
print(f"{round(c, 2)}°C")
```

03_problem3.py

```
print("a")
print("b")
```

```
print("c", end="")
print("d", end=""),
```

04_problem4.py

```
'''
sum(1) = 1
sum(2) = 1 + 2
sum(3) = 1 + 2 + 3
sum(4) = 1 + 2 + 3 + 4
sum(5) = 1 + 2 + 3 + 4 + 5

sum(n) = 1 + 2 + 3 + 4.... n -1 + n
sum(n) = sum(n-1) + n
'''

def sum(n):
    if(n==1):
        return 1
    return sum(n-1) + n

print(sum(4))
```

05_problem5.py

```
def pattern(n):
    if(n==0):
        return
    print("*" * n)
    pattern(n-1)
```

```
pattern(3)
```

06_problem6.py

```
def inch_to_cms(inch):
    return inch * 2.54

n = int(input("Enter value in inches: "))

print(f"The corresponding value in cms is {inch_to_cms(n)}")
```

07_problem7.py

```
def rem(l, word):
    n = []
    for item in l:
        if not(item == word):
            n.append(item.strip(word))
    return n

l = ["Harry", "Rohan", "Shubham", "an"]

print(rem(l, "an"))
```

08_problem8.py

```
def multiply(n):
```

```
    for i in range(1, 11):
        print(f"{n} X {i} = {n*i}")


multiply(5)
```

Chapter 9

01_file.py

```
'''
a = "a very long string with emails"

emails = []
3 seconds
'''

f = open("file.txt", "r")
data = f.read()
print(data)
f.close()
```

02_file_write.py

```
st = "Hey Harry you are amazing"

f = open("myfile.txt", "w")

f.write(st)

f.close()
```

03_more_file_functions.py

```
f = open("file.txt")

# lines = f.readlines()
```

```
# print(lines, type(lines))


# line1 = f.readline()
# print(line1, type(line1))


# line2 = f.readline()
# print(line2, type(line2))


# line3 = f.readline()
# print(line3, type(line3))


# line4 = f.readline()
# print(line4, type(line4))


# line5 = f.readline()
# print(line5 =="")
line = f.readline()
while(line != ""):
    print(line)
    line = f.readline()


f.close()
```

04_append.py

```
st = "Hey Harry you are amazing"


f = open("myfile.txt", "a")
```

```
f.write(st)


f.close()
```

05_with.py

```
f = open("file.txt")
print(f.read())
f.close()


# The same can be written using with statement like this:
with open("file.txt") as f:
    print(f.read())


# You dont have to explicitly close the file
```

file.txt

```
Harry is a good boy
I am a second line
This is amazing
Twinkle Twinkle little star
```

myfile.txt

```
Hey Harry you are amazingHey Harry you are amazingHey Harry you are amazingHey Harry you are amazingHey Harry you are amazingHey Harry you are amazingHey Harry you are amazingHey Harry you are amazingHey Harry you are amazingHey Harry you are amazing
```

Chapter 9 - PS

01_problem1.py

```
f = open("poem.txt")
content = f.read()
if("twinkle" in content):
    print("The word twinkle is present in the content")

else:
    print("The word twinkle is not present in the content")

f.close()
```

02_problem2.py

```
import random

def game():
    print("You are playing the game..")
    score = random.randint(1, 62)
    # Fetch the hiscore
    with open("hiscore.txt") as f:
        hiscore = f.read()
        if(hiscore!=""):
            hiscore = int(hiscore)
        else:
            hiscore = 0
```

```
    print(f"Your score: {score}")
    if(score>hiscore):
        # write this hiscore to the file
        with open("hiscore.txt", "w") as f:
            f.write(str(score))

    return score

game()
```

03_problem3.py

```
def generateTable(n):
    table = ""
    for i in range(1, 11):
        table += f"{n} X {i} = {n*i}\n"

    with open(f"tables/table_{n}.txt", "w") as f:
        f.write(table)




for i in range(2, 21):
    generateTable(i)
```

04_problem4.py

```
word = "Donkey"
```

```
with open("file.txt", "r") as f:
    content = f.read()

contentNew = content.replace(word, "######")

with open("file.txt", "w") as f:
    f.write(contentNew)
```

05_problem5.py

```
words = ["Donkey", "bad", "ganda"]

with open("file.txt", "r") as f:
    content = f.read()

for word in words:
    content = content.replace(word, "#" * len(word))

with open("file.txt", "w") as f:
    f.write(content)
```

06_problem.py

```
with open("log.txt") as f:
    content = f.read()

if("python" in content):
    print("Yes python is present")
else:
```

```
    print("No Python is not present")
```

07_problem7.py

```
with open("log.txt") as f:
    lines = f.readlines()

lineno = 1
for line in lines:
    if("python" in line):
        print(f"Yes python is present. Line no: {lineno}")
        break
    lineno += 1

else:
    print("No Python is not present")
```

08_problem8.py

```
with open("this.txt") as f:
    content = f.read()

with open("this_copy.txt", "w") as f:
    f.write(content)
```

09_problem9.py

```
with open("this.txt") as f:
    content1 = f.read()
```

```
with open("this_copy.txt") as f:
    content2 = f.read()

if(content1 == content2):
    print("Yes these files are identical")

else:
    print("No these files are not identical")
```

10_problem10.py

```
with open("this_copy.txt", "w") as f:
    f.write("")
```

11_problem.py

```
with open("old.txt") as f:
    content = f.read()

with open("renamed_by_python.txt", "w") as f:
    f.write(content)
```

file.txt

```
###### is a nice ###### but not a ### ###### #####

bhai aap bahut hi ##### ho. ###### is good
```

hiscore.txt

```
45
```

log.txt

Lorem ipsum dolor sit amet consectetur adipisicing elit. Voluptatibus illo corrupti minima cupiditate? Molestias officiis doloribus architecto similique natus nesciunt, hic aspernatur fugit vitae! Placeat tempore suscipit deserunt, consequatur nobis reiciendis dolor dolorem sunt corporis minus labore! Animi quaerat optio nemo, similique, perspiciatis tenetur in nostrum unde eum porro officiis. Eum aperiam debitis sed cum tempore ipsam ex, fugit, voluptas beatae necessitatibus placeat deleniti porro tenetur non? Maxime voluptates, nostrum cupiditate reprehenderit error quod in ullam id tenetur perspiciatis beatae ex, sit dolores ducimus sint quos, qui blanditiis? Optio enim soluta ducimus quasi odio, ipsam voluptas mollitia debitis deserunt sapiente et harum suscipit eius. Excepturi, fugiat amet? Nulla

python repellat magni libero, voluptatem, earum maxime animi sunt architecto harum atque asperiores provident dolores laborum itaque accusamus at. Aut voluptates numquam amet odit labore, similique eius minus quas unde asperiores cupiditate aliquam fugit rerum eligendi. Qui officia commodi maxime soluta quisquam exercitationem laboriosam sit amet, nisi officiis fugiat aut, obcaecati reiciendis culpa beatae dolore. Reiciendis numquam sunt ratione molestiae. Ea quas, tempora esse, pariatur voluptas adipisci tempore dicta quos harum non autem cumque exercitationem distinctio odio ex ipsa veritatis pytsdfhon dolores nam provident voluptatum officiis debitis quo. Dolores vel perspiciatis debitis ad error est voluptatem voluptatibus, quod fugiat optio, voluptate ea quos perferendis cupiditate doloremque architecto praesentium beatae libero officiis tempore cum temporibus illum totam tenetur! Dolores cum deserunt delectus earum nostrum inventore dignissimos error harum aspernatur consequatur numquam itaque architecto, rerum quaerat natus quae saepe? Ratione nam et explicabo ex quis architecto, dolores voluptate consequatur perspiciatis sapiente voluptatem fuga vitae ut blanditiis numquam quibusdam error velit id laudantium reiciendis. At, voluptatibus voluptates alias exercitationem ipsam voluptate itaque aut molestias similique repudiandae deleniti nam omnis voluptas possimus. Maxime iste minima quod maiores facilis ipsam, vero eveniet aperiam assumenda blanditiis esse nostrum fugiat veniam vel ullam obcaecati dicta. Non error nisi necessitatibus facilis, blanditiis illo quisquam doloremque quis

enim recusandae tenetur corporis molestias repudiandae iste, perferendis asperiores ea eligendi, reprehenderit autem. Ipsum optio dignissimos cum qui, ut quia praesentium vitae incidunt similique aut beatae est? Ab sit unde debitis expedita minus reprehenderit laudantium laboriosam, alias eveniet beatae consequuntur cupiditate in quibusdam similique dolores tempora. Sunt ex officiis amet rem voluptas nobis quidem, harum deserunt, laborum cum vel aut dignissimos reiciendis architecto consequuntur assumenda odio et! Porro nam aliquam officia optio, sint quis minus, quam nesciunt alias eum mollitia ipsum saepe dolor similique. Sunt et repellat est illo. At commodi deserunt natus eum consequuntur? Enim at voluptates, excepturi est nulla praesentium laboriosam temporibus rerum aut vitae hic quam deserunt commodi facilis officia soluta sed, inventore, dolorum eius. Magni, eaque expedita.

old.txt

I am old

poem.txt

twinkle twinkle little star

Kaise ho mere bhai log yaar

renamed_by_python.txt

I am old

this.txt

hey mai this hooon

muje bahut maze aa rahe hain

mai maze me aa rha hu

this_copy.txt

Chapter 10

01_class.py

```
class Employee:
    language = "Py" # This is a class attribute
    salary = 1200000



harry = Employee()
harry.name = "Harry" # This is an instance attribute
print(harry.name, harry.language, harry.salary)

rohan = Employee()
rohan.name = "Rohan Roro Robinson"
print(rohan.name, rohan.salary, rohan.language)

# Here name is instance attribute and salary and language are class attributes as they directly belong to the class
```

02_instance_vs_class_attr.py

```
class Employee:
    language = "Python" # This is a class attribute
    salary = 1200000



harry = Employee()
harry.language = "JavaScript" # This is an instance attribute
```

```
print(harry.language, harry.salary)
```

03_self.py

```
class Employee:
    language = "Python" # This is a class attribute
    salary = 1200000

    def getInfo(self):
        print(f"The language is {self.language}. The salary is {self.salary}")

    @staticmethod
    def greet():
        print("Good morning")


harry = Employee()
# harry.language = "JavaScript" # This is an instance attribute
harry.greet()
harry.getInfo()
# Employee.getInfo(harry)
```

04_constructor.py

```
class Employee:
    language = "Python" # This is a class attribute
    salary = 1200000
```

```
    def __init__(self, name, salary, language): # dunder method which is automatically called
        self.name = name
        self.salary = salary
        self.language = language
        print("I am creating an object")


    def getInfo(self):
        print(f"The language is {self.language}. The salary is {self.salary}")

    @staticmethod
    def greet():
        print("Good morning")


harry = Employee("Harry", 1300000, "JavaScript")
# harry.name = "Harry"
print(harry.name, harry.salary, harry.language)

rohan = Employee()
```

Chapter 10 - PS

01_problem1.py

```
class Programmer:
    company = "Microsoft"
    def __init__(self, name, salary, pin):
        self.name = name
        self.salary = salary
        self.pin = pin



p = Programmer("Harry", 1200000, 245001)
print(p.name, p.salary, p.pin, p.company)
r = Programmer("Rohan", 1200000, 245001)
print(r.name, r.salary, r.pin, r.company)
```

02_problem2.py

```
class Calculator:
    def __init__(self, n):
        self.n = n

    def square(self):
        print(f"The square is {self.n*self.n}")

    def cube(self):
        print(f"The cube is {self.n*self.n*self.n}")
```

```
    def squareroot(self):
        print(f"The squareroot is {self.n**1/2}")


a = Calculator(4)
a.square()
a.cube()
a.squareroot()
```

03_problem3.py

```
class Demo:
    a = 4


o = Demo()
print(o.a) # Prints the class attribute because instance attribute is not present
o.a = 0 # Instance attribute is set
print(o.a) # Prints the instance attribute because instance attribute is present
print(Demo.a) # Prints the class attribute
```

04_problem4.py

```
class Calculator:
    def __init__(self, n):
        self.n = n


    def square(self):
        print(f"The square is {self.n*self.n}")


    def cube(self):
        print(f"The cube is {self.n*self.n*self.n}")
```

```
    def squareroot(self):
        print(f"The squareroot is {self.n**1/2}")

    @staticmethod
    def hello():
        print("Hello there!")

a = Calculator(4)
a.hello()
a.square()
a.cube()
a.squareroot()
```

05_problem5.py

```
from random import randint

class Train:

    def __init__(self, trainNo):
        self.trainNo = trainNo

    def book(self, fro, to):
        print(f"Ticket is booked in train no: {self.trainNo} from {fro} to {to}")

    def getStatus(self):
        print(f"Train no: {self.trainNo} is running on time")
```

```
    def getFare(self, fro, to):
        print(f"Ticket fare in train no: {self.trainNo} from {fro} to {to} is {randint(222, 5555)}")



t = Train(12399)
t.book("Rampur", "Delhi")
t.getStatus()
t.getFare("Rampur", "Delhi")
```

06_problem6.py

```
from random import randint


class Train:

    def __init__(slf, trainNo):
        slf.trainNo = trainNo

    def book(harry, fro, to):
        print(f"Ticket is booked in train no: {harry.trainNo} from {fro} to {to}")

    def getStatus(self):
        print(f"Train no: {self.trainNo} is running on time")

    def getFare(self, fro, to):
        print(f"Ticket fare in train no: {self.trainNo} from {fro} to {to} is {randint(222, 5555)}")
```

```
t = Train(12399)

t.book("Rampur", "Delhi")

t.getStatus()

t.getFare("Rampur", "Delhi")
```

Chapter 11

01_inheritance.py

```
class Employee:
    company = "ITC"
    def show(self):
        print(f"The name of the Employee is {self.name} and the salary is {self.salary}")



# class Programmer:
#     company = "ITC Infotech"
#     def show(self):
#         print(f"The name is {self.name} and the salary is {self.salary}")

#     def showLanguage(self):
#         print(f"The name is {self.name} and he is good with {self.language} language")



class Programmer(Employee):
    company = "ITC Infotech"
    def showLanguage(self):
        print(f"The name is {self.name} and he is good with {self.language} language")



a = Employee()
b = Programmer()
```

```
print(a.company, b.company)
```

02_multiple_inheritance.py

```
class Employee:
    company = "ITC"
    name = "Default name"
    def show(self):
        print(f"The name of the Employee is {self.name} and the company is {self.company}")


class Coder:
    language = "Python"
    def printLanguages(self):
        print(f"Out of all the languages here is your language: {self.language}")




class Programmer(Employee, Coder):
    company = "ITC Infotech"
    def showLanguage(self):
        print(f"The name is {self.company} and he is good with {self.language} language")



a = Employee()
b = Programmer()

b.show()
b.printLanguages()
```

```
b.showLanguage()
```

03_multilevel_inheritance.py

```
class Employee:
    a = 1


class Programmer(Employee):
    b = 2


class Manager(Programmer):
    c = 3


o = Employee()
print(o.a) # Prints the a attribute
# print(o.b) # Shows an error as there is no b attribute in Employee class


o = Programmer()
print(o.a, o.b)



o = Manager()
print(o.a, o.b, o.c)
```

04_super.py

```
class Employee:
    def __init__(self):
        print("Constructor of Employee")
    a = 1
```

```
class Programmer(Employee):
    def __init__(self):
        print("Constructor of Programmer")
    b = 2


class Manager(Programmer):
    def __init__(self):
        super().__init__()
        print("Constructor of Manager")
    c = 3


# o = Employee()
# print(o.a) # Prints the a attribute


# o = Programmer()
# print(o.a, o.b)



o = Manager()
print(o.a, o.b, o.c)
```

05_class_methods.py

```
class Employee:
    a = 1


    @classmethod
```

```
    def show(cls):
        print(f"The class attribute of a is {cls.a}")


e = Employee()
e.a = 45


e.show()
```

06_property_decorators.py

```
class Employee:
    a = 1


    @classmethod
    def show(cls):
        print(f"The class attribute of a is {cls.a}")


    @property
    def name(self):
        return f"{self.fname} {self.lname}"


    @name.setter
    def name (self,value):
        self.fname = value.split(" ")[0]
        self.lname = value.split(" ")[1]


e = Employee()
e.a = 45
```

```
e.name = "Harry Khan"

print(e.fname, e.lname)


e.show()
```

07_operator_overloading.py

```
class Number:
    def __init__(self, n):
        self.n = n


    def __add__(self, num):
        return self.n + num.n


n = Number(1)

m = Number(2)


print(n + m)
```

Chapter 11 - PS

01_problem1.py

```
class TwoDVector:
    def __init__(self, i, j):
        self.i = i
        self.j = j

    def show(self):
        print(f"The vector is {self.i}i + {self.j}j ")



class ThreeDVector(TwoDVector):
    def __init__(self, i, j, k):
        super().__init__(i, j)
        self.k = k

    def show(self):
        print(f"The vector is {self.i}i + {self.j}j + {self.k}k")


a = TwoDVector(1, 2)
a.show()
b = ThreeDVector(5, 2, 3)
b.show()
```

02_problem2.py

```
class Animals:
```

```
    pass


class Pets(Animals):
    pass


class Dog(Pets):

    @staticmethod
    def bark():
        print("Bow Bow!")



d = Dog()

d.bark()
```

03_problem3.py

```
class Employee:
    salary = 234
    increment = 20

    @property
    def salaryAfterIncrement(self):
        return (self.salary + self.salary * (self.increment/100))

    @salaryAfterIncrement.setter
    def salaryAfterIncrement(self, salary):
```

```
        self.increment =  ((salary/self.salary) -1)*100




e = Employee()
# print(e.salaryAfterIncrement)
e.salaryAfterIncrement = 280.8
print(e.increment)
```

04_problem4.py

```
class Complex:
    def __init__(self, r, i):
        self.r = r
        self.i = i

    def __add__(self, c2):
        return Complex(self.r + c2.r, self.i + c2.i)

    def __mul__(self, c2):
        real_part = self.r * c2.r - self.i * c2.i
        imag_part = self.r * c2.i + self.i * c2.r
        return Complex(real_part, imag_part)

    def __str__(self):
        return f"{self.r} + {self.i}i"
```

```
c1 = Complex(1, 2)
c2 = Complex(3, 4)
print(c1 + c2)
print(c1*c2)
```

05_problem5.py

```
class Vector:
    def __init__(self, x, y, z):
        self.x = x
        self.y = y
        self.z = z

    def __add__(self, other):
        result = Vector(self.x + other.x, self.y + other.y, self.z + other.z)
        return result

    def __mul__(self, other):
        result = self.x * other.x + self.y * other.y + self.z * other.z
        return result

    def __str__(self):
        return f"Vector({self.x}i+ {self.y}j + {self.z}k)"

# Test the implementation
v1 = Vector(1, 2, 3)
```

```
v2 = Vector(4, 5, 6)
v3 = Vector(7, 8, 9)  # Same dimension vector

print(v1 + v2)  # Output: Vector(5, 7, 9)
print(v1 * v2)  # Output: 32

print(v1 + v3)  # Output: Vector(8, 10, 12)
print(v1 * v3)  # Output: 50
```

06_problem6.py

```
class Vector:
    def __init__(self, x, y, z):
        self.x = x
        self.y = y
        self.z = z

    def __add__(self, other):
        result = Vector(self.x + other.x, self.y + other.y, self.z + other.z)
        return result

    def __mul__(self, other):
        result = self.x * other.x + self.y * other.y + self.z * other.z
        return result

    def __str__(self):
        return f"{self.x}i + {self.y}j + {self.z}k"
```

```
# Test the implementation
v1 = Vector(1, 2, 3)
v2 = Vector(4, 5, 6)
v3 = Vector(7, 8, 9)  # Same dimension vector

print(v1 + v2)  # Output: Vector(5, 7, 9)
print(v1 * v2)  # Output: 32

print(v1 + v3)  # Output: Vector(8, 10, 12)
print(v1 * v3)  # Output: 50
```

07_problem7.py

```
class Vector:
    def __init__(self, l):
        self.l = l




    def __len__(self):
        return len(self.l)

# Test the implementation
v1 = Vector([1, 2, 3, 4])
print(len(v1))
```

Chapter 12

01_walrus.py

```
# Using walrus operator
if (n := len([1, 2, 3, 4, 5])) > 3:
    print(f"List is too long ({n} elements, expected <= 3)") # Output: List is too long (5 elements, expected <= 3)
```

02_types.py

```
from typing import List, Union, Tuple


n : int = 5


name: str = "Harry"



def  sum(a: int, b: int) -> int:
    return a+b
```

03_match_case.py

```
def http_status(status):
    match status:
        case 200:
            return "OK"
        case 404:
            return "Not Found"
        case 500:
```

```
            return "Internal Server Error"
        case _:
            return "Unknown status"



print(http_status(5007))
```

04_exception.py

```
try:
    a = int(input("Hey, Enter a number: "))
    print(a)

except ValueError as v:
    print("Heyyyy!")
    print(v)

except Exception as e:
    print(e)
print(500)
```

05_raising_exceptions.py

```
a = int(input("Enter a number: "))
b = int(input("Enter second number: "))

if(b == 0):
    raise ZeroDivisionError("Hey our program is not meant to divide numbers by zero")
else:
```

```
    print(f"The division a/b is {a/b}")
```

06_try_else.py

```
try:
    a = int(input("Hey, Enter a number: "))
    print(a)



except Exception as e:
    print(e)



else:
    print("I am inside else")
```

07_try_finally.py

```
def main():
    try:
        a = int(input("Hey, Enter a number: "))
        print(a)
        return



    except Exception as e:
        print(e)
        return
```

```
    finally:
        print("Hey I am inside of finally")



main()
```

08_main.py

```
from module import myFunc



```

09_global.py

```
a = 89


def fun():
    # global a
    a = 3
    print(a)



fun()
print(a)
```

10_enumerate.py

```
l = [3, 513, 53, 535]


# index = 0
# for item in l:
```

```
#     print(f"The item number at index {index} is {item}")
#     index += 1

# This can be simplified using enumerate function

for index, item in enumerate(l):
    print(f"The item number at index {index} is {item}")
```

11_list_comprehensions.py

```
myList = [1, 2, 9, 5, 3, 5]

# squaredList = []
# for item in myList:
#     squaredList.append(item*item)

squaredList = [i*i for i in myList]

print(squaredList)
```

module.py

```
def myFunc():
    print("Hello world!")



if __name__ == "__main__":
    # If this code is directly executed by running the file its present in
    print("We are directly running this code")
```

```
myFunc()

print(__name__)
```

Chapter 12 - PS

01_problem1.py

```
try:
    with open("1.txt", "r") as f:
        print(f.read())
except Exception as e:
    print(e)

try:
    with open("2.txt", "r") as f:
        print(f.read())
except Exception as e:
    print(e)

try:
    with open("3.txt", "r") as f:
        print(f.read())
except Exception as e:
    print(e)



print("Thank You!")
```

02_problem2.py

```
l = [1, 2, 3, 4, 5, 6 ,7 , 8]
```

```
for i, item in enumerate(l):
    if i == 2 or i == 4 or i == 6:
        print(item)
```

03_problem3.py

```
n = int(input("Enter a number: "))


table = [n*i for i in range(1, 11)]
print(table)
```

04_problem4.py

```
try:
    a = int(input("Enter a: "))
    b = int(input("Enter b: "))
    print(a/b)
except ZeroDivisionError as v:
    print("Infinite")
```

05problem5.py

```
n = int(input("Enter a number: "))


table = [n*i for i in range(1, 11)]
with open("tables.txt", "a") as f:
    f.write(f"Table of {n}: {str(table)} \n")
```

tables.txt

```
Table of 5: [5, 10, 15, 20, 25, 30, 35, 40, 45, 50]
Table of 2: [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
```

Table of 22: [22, 44, 66, 88, 110, 132, 154, 176, 198, 220]

Table of 3323423: [3323423, 6646846, 9970269, 13293692, 16617115, 19940538, 23263961, 26587384, 29910807, 33234230]

Chapter 13

01_venv.py

02_lambda.py

```
# def square(n):
#     return n*n

square = lambda x: x*x

print(square(5))
```

03_join.py

```
a = ["Harry", "Rohan", "Shubham"]

final = "::".join(a)
print(final)
```

04_format.py

```
a = "{1} is a good {0}".format("harry", "boy")

print(a)
```

05_map_filter_reduce.py

```
from functools import reduce
# Map Example
l = [1, 2, 3, 4, 5]
```

```
square = lambda x: x*x


sqList = map(square, l)
print(list(sqList))


# Filter Example
def even(n):
    if (n%2 == 0):
        return True
    return False


onlyEven= filter(even, l)
print(list(onlyEven))


# Reduce Example
def sum(a, b):
    return a + b


mul = lambda x,y:x*y


print(reduce(sum, l))
print(reduce(mul, l))
```

requirements.txt

```
n u m p y = = 1 . 2 6 . 4


 p a n d a s = = 2 . 2 . 2
```

python-dateutil==2.9.0.post0

pytz==2024.1

six==1.16.0

tzdata==2024.1

Chapter 13 - PS

01_problem1.py

02_problem2.py

```
name = input("Enter name: ")
marks = int(input("Enter marks: "))
phone = int(input("Phone number: "))

s = "The name of the student is {}, his marks are {} and phone number is {}".format(name, marks, phone)

print(s)
```

03_problem3.py

```
table = [str(7*i) for i in range(1, 11)]

s = "\n".join(table)
print(s)
```

04_problem4.py

```
def divisible5(n):
    if(n%5 == 0):
        return True
    return False

a  = [1,2,34234,53,6234235,64343, 65,754,45,55]
```

```
f = list(filter(divisible5, a))
print(f)
```

05_problem5.py

```
from functools import reduce
l  = [111, 2, 65, 5553, 635, 65, 74, 45,55]



def greater(a, b):
    if (a>b):
        return a
    return b


print(reduce(greater, l))
```

06_problem6.py

```
# pip freeze > requirements.txt
# virtualenv harryenv
```

07_problem7.py

```
from flask import Flask


app = Flask(__name__)


@app.route("/")
def hello_world():
    return "<p>Hello, World!</p>"
```

app.run()

requirements.txt

```
beautifulsoup4==4.12.3

blinker==1.8.2

certifi==2024.2.2

charset-normalizer==3.3.2

click==8.1.7

colorama==0.4.6

comtypes==1.4.2

distlib==0.3.8

filelock==3.14.0

Flask==3.0.3

idna==3.7

itsdangerous==2.2.0
```

Jinja2==3.1.4

MarkupSafe==2.1.5

numpy==1.26.4

pandas==1.5.2

platformdirs==4.2.2

pydub==0.25.1

pygame==2.5.2

pyjokes==0.6.0

pypiwin32==223

python-dateutil==2.9.0.post0

pyttsx3==2.90

pytz==2024.1

pywin32==306

requests==2.32.3

six==1.16.0

soupsieve==2.5

tzdata==2024.1

urllib3==2.2.1

virtualenv==20.26.2

Werkzeug==3.0.3

wikipedia==1.4.0

env1

env2

harryenv

Mega Project 1 - Jarvis

client.py

```
from openai import OpenAI


# pip install openai
# if you saved the key under a different environment variable name, you can do something like:
client = OpenAI(
  api_key="<Your Key Here>",
)


completion = client.chat.completions.create(
  model="gpt-3.5-turbo",
  messages=[
    {"role": "system", "content": "You are a virtual assistant named jarvis skilled in general tasks like Alexa and Google Cloud"},
    {"role": "user", "content": "what is coding"}
  ]
)


print(completion.choices[0].message.content)
```

main.py

```
import speech_recognition as sr
import webbrowser
import pyttsx3
import musicLibrary
```

```
import requests
from openai import OpenAI
from gtts import gTTS
import pygame
import os

# pip install pocketsphinx

recognizer = sr.Recognizer()
engine = pyttsx3.init()
newsapi = "<Your Key Here>"

def speak_old(text):
    engine.say(text)
    engine.runAndWait()

def speak(text):
    tts = gTTS(text)
    tts.save('temp.mp3')

    # Initialize Pygame mixer
    pygame.mixer.init()

    # Load the MP3 file
    pygame.mixer.music.load('temp.mp3')

    # Play the MP3 file
```

```
    pygame.mixer.music.play()

    # Keep the program running until the music stops playing
    while pygame.mixer.music.get_busy():
        pygame.time.Clock().tick(10)

    pygame.mixer.music.unload()
    os.remove("temp.mp3")

def aiProcess(command):
    client = OpenAI(api_key="<Your Key Here>",
    )

    completion = client.chat.completions.create(
    model="gpt-3.5-turbo",
    messages=[
        {"role": "system", "content": "You are a virtual assistant named jarvis skilled in general tasks like Alexa and Google Cloud. Give short responses please"},
        {"role": "user", "content": command}
    ]
    )

    return completion.choices[0].message.content

def processCommand(c):
    if "open google" in c.lower():
        webbrowser.open("https://google.com")
```

```
elif "open facebook" in c.lower():
    webbrowser.open("https://facebook.com")
elif "open youtube" in c.lower():
    webbrowser.open("https://youtube.com")
elif "open linkedin" in c.lower():
    webbrowser.open("https://linkedin.com")
elif c.lower().startswith("play"):
    song = c.lower().split(" ")[1]
    link = musicLibrary.music[song]
    webbrowser.open(link)

elif "news" in c.lower():
    r = requests.get(f"https://newsapi.org/v2/top-headlines?country=in&apiKey={newsapi}")
    if r.status_code == 200:
        # Parse the JSON response
        data = r.json()

        # Extract the articles
        articles = data.get('articles', [])

        # Print the headlines
        for article in articles:
            speak(article['title'])

else:
    # Let OpenAI handle the request
    output = aiProcess(c)
```

```
        speak(output)




if __name__ == "__main__":
    speak("Initializing Jarvis....")
    while True:
        # Listen for the wake word "Jarvis"
        # obtain audio from the microphone
        r = sr.Recognizer()


        print("recognizing...")
        try:
            with sr.Microphone() as source:
                print("Listening...")
                audio = r.listen(source, timeout=2, phrase_time_limit=1)
            word = r.recognize_google(audio)
            if(word.lower() == "jarvis"):
                speak("Ya")
                # Listen for command
                with sr.Microphone() as source:
                    print("Jarvis Active...")
                    audio = r.listen(source)
                    command = r.recognize_google(audio)
```

```
            processCommand(command)


    except Exception as e:
        print("Error; {0}".format(e))
```

musicLibrary.py

```
music = {
    "stealth": "https://www.youtube.com/watch?v=U47Tr9BB_wE",
    "march": "https://www.youtube.com/watch?v=Xqeq4b5u_Xw",
    "skyfall": "https://www.youtube.com/watch?v=DeumyOzKqgI&pp=ygUHc2t5ZmFsbA%3D%3D",
                                                                    "wolf":
"https://www.youtube.com/watch?v=ThCH0U6aJpU&list=PLnrGi_-oOR6wm0Vi-1OsiLiV5ePSPs9oF
&index=21"
}
```

Mega Project 2 - AI AutoReply Bot

01_get_cursor.py

```
import pyautogui


while True:
    a = pyautogui.position()
    print(a)
    # 1639, 1412
    # 1003 237 to 2187 1258
    # 1026 244
    # to 1131 1321
```

02_openai.py

```
from openai import OpenAI


# pip install openai
# if you saved the key under a different environment variable name, you can do something like:
client = OpenAI(
  api_key="<Your Key Here>",
)


command = '''
[20:30, 12/6/2024] Naruto: jo sunke coding ho sake?
[20:30, 12/6/2024] Rohan Das: https://www.youtube.com/watch?v=DzmG-4-OASQ
[20:30, 12/6/2024] Rohan Das: ye
[20:30, 12/6/2024] Rohan Das: https://www.youtube.com/watch?v=DzmG-4-OASQ
```

[20:31, 12/6/2024] Naruto: This is hindi

[20:31, 12/6/2024] Naruto: send me some english songs

[20:31, 12/6/2024] Naruto: but wait

[20:31, 12/6/2024] Naruto: this song is amazing

[20:31, 12/6/2024] Naruto: so I will stick to it

[20:31, 12/6/2024] Naruto: send me some english song also

[20:31, 12/6/2024] Rohan Das: hold on

[20:31, 12/6/2024] Naruto: I know what you are about to send

[20:32, 12/6/2024] Naruto: ??

[20:32, 12/6/2024] Rohan Das: https://www.youtube.com/watch?v=ar-3chBG4NU

ye hindi English mix hai but best hai

[20:33, 12/6/2024] Naruto: okok

[20:33, 12/6/2024] Rohan Das: Haan

'''

```
completion = client.chat.completions.create(
  model="gpt-3.5-turbo",
  messages=[
    {"role": "system", "content": "You are a person named harry who speaks hindi as well as english. He is from India and is a coder. You analyze chat history and respond like Harry"},
    {"role": "user", "content": command}
  ]
)


print(completion.choices[0].message.content)
```

03_bot.py

```
import pyautogui
```

```
import time
import pyperclip
from openai import OpenAI




client = OpenAI(
  api_key="<Your Key Here>",
)

def is_last_message_from_sender(chat_log, sender_name="Rohan Das"):
    # Split the chat log into individual messages
    messages = chat_log.strip().split("/2024] ")[-1]
    if sender_name in messages:
        return True
    return False




    # Step 1: Click on the chrome icon at coordinates (1639, 1412)
pyautogui.click(1639, 1412)

time.sleep(1)  # Wait for 1 second to ensure the click is registered
while True:
    time.sleep(5)
    # Step 2: Drag the mouse from (1003, 237) to (2187, 1258) to select the text
```

```
    pyautogui.moveTo(972,202)
    pyautogui.dragTo(2213, 1278, duration=2.0, button='left')  # Drag for 1 second

    # Step 3: Copy the selected text to the clipboard
    pyautogui.hotkey('ctrl', 'c')
    time.sleep(2)  # Wait for 1 second to ensure the copy command is completed
    pyautogui.click(1994, 281)

    # Step 4: Retrieve the text from the clipboard and store it in a variable
    chat_history = pyperclip.paste()

    # Print the copied text to verify
    print(chat_history)
    print(is_last_message_from_sender(chat_history))
    if is_last_message_from_sender(chat_history):
        completion = client.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=[
            {"role": "system", "content": "You are a person named Naruto who speaks hindi as well as english. You are from India and you are a coder. You analyze chat history and roast people in a funny way. Output should be the next chat response (text message only)"},
            {"role": "system", "content": "Do not start like this [21:02, 12/6/2024] Rohan Das: "},
            {"role": "user", "content": chat_history}
        ]
        )

        response = completion.choices[0].message.content
```

```
pyperclip.copy(response)


# Step 5: Click at coordinates (1808, 1328)
pyautogui.click(1808, 1328)
time.sleep(1)  # Wait for 1 second to ensure the click is registered

# Step 6: Paste the text
pyautogui.hotkey('ctrl', 'v')
time.sleep(1)  # Wait for 1 second to ensure the paste command is completed

# Step 7: Press Enter
pyautogui.press('enter')
```

Project 1

main-shortened.py

```
import random
'''
1 for snake
-1 for water
0 for gun
'''
computer = random.choice([-1, 0, 1])
youstr = input("Enter your choice: ")
youDict = {"s": 1, "w": -1, "g": 0}
reverseDict = {1: "Snake", -1: "Water", 0: "Gun"}

you = youDict[youstr]

# By now we have 2 numbers (variables), you and computer

print(f"You chose {reverseDict[you]}\nComputer chose {reverseDict[computer]}")

if(computer == you):
    print("Its a draw")

else:
    '''
     if(computer ==-1 and you == 1): (computer - you) = -2
        print("You win!")
```

```
    elif(computer ==-1 and you == 0): (computer - you) = -1
        print("You Lose!")

    elif(computer == 1 and you == -1): (computer - you) = 2
        print("You lose!")

    elif(computer ==1 and you == 0): (computer - you) = 1
        print("You Win!")

    elif(computer ==0 and you == -1): (computer - you) = 1
        print("You Win!")

    elif(computer == 0 and you == 1): computer - you) = -1
        print("You Lose!")

        The below logic is written on the basis of the value of computer - you

    '''
    if((computer - you) == -1 or  (computer - you) == 2 ):
        print("You lose!")
    else:
        print("You win!")
```

main.py

```
import random
'''
```

```
1 for snake

-1 for water

0 for gun

'''

computer = random.choice([-1, 0, 1])

youstr = input("Enter your choice: ")

youDict = {"s": 1, "w": -1, "g": 0}

reverseDict = {1: "Snake", -1: "Water", 0: "Gun"}


you = youDict[youstr]


# By now we have 2 numbers (variables), you and computer


print(f"You chose {reverseDict[you]}\nComputer chose {reverseDict[computer]}")


if(computer == you):

    print("Its a draw")


else:

    if(computer ==-1 and you == 1):

        print("You win!")


    elif(computer ==-1 and you == 0):

        print("You Lose!")


    elif(computer == 1 and you == -1):

        print("You lose!")
```

```
elif(computer ==1 and you == 0):
    print("You Win!")

elif(computer ==0 and you == -1):
    print("You Win!")

elif(computer == 0 and you == 1):
    print("You Lose!")

else:
    print("Something went wrong!")
```

Project 2

main.py

```
import random

n = random.randint(1, 100)

a = -1

guesses = 1

while(a != n):

    a = int(input("Guess the number: "))

    if(a >n):

        print("Lower number please")

        guesses +=1

    elif(a<n):

        print("Higher number Please")

        guesses +=1


print(f"You have guessed the number {n} correctly in {guesses} attempts")
```

tables

table_2.txt

2 X 1 = 2

2 X 2 = 4

2 X 3 = 6

2 X 4 = 8

2 X 5 = 10

2 X 6 = 12

2 X 7 = 14

2 X 8 = 16

2 X 9 = 18

2 X 10 = 20

table_3.txt

3 X 1 = 3

3 X 2 = 6

3 X 3 = 9

3 X 4 = 12

3 X 5 = 15

3 X 6 = 18

3 X 7 = 21

3 X 8 = 24

3 X 9 = 27

3 X 10 = 30

table_4.txt

4 X 1 = 4

4 X 2 = 8

4 X 3 = 12

4 X 4 = 16

4 X 5 = 20

4 X 6 = 24

4 X 7 = 28

4 X 8 = 32

4 X 9 = 36

4 X 10 = 40

table_5.txt

5 X 1 = 5

5 X 2 = 10

5 X 3 = 15

5 X 4 = 20

5 X 5 = 25

5 X 6 = 30

5 X 7 = 35

5 X 8 = 40

5 X 9 = 45

5 X 10 = 50

table_6.txt

6 X 1 = 6

6 X 2 = 12

6 X 3 = 18

6 X 4 = 24

6 X 5 = 30

6 X 6 = 36

6 X 7 = 42

6 X 8 = 48

6 X 9 = 54

6 X 10 = 60

table_7.txt

7 X 1 = 7

7 X 2 = 14

7 X 3 = 21

7 X 4 = 28

7 X 5 = 35

7 X 6 = 42

7 X 7 = 49

7 X 8 = 56

7 X 9 = 63

7 X 10 = 70

table_8.txt

8 X 1 = 8

8 X 2 = 16

8 X 3 = 24

8 X 4 = 32

8 X 5 = 40

8 X 6 = 48

8 X 7 = 56

8 X 8 = 64

8 X 9 = 72

8 X 10 = 80

table_9.txt

9 X 1 = 9

9 X 2 = 18

9 X 3 = 27

9 X 4 = 36

9 X 5 = 45

9 X 6 = 54

9 X 7 = 63

9 X 8 = 72

9 X 9 = 81

9 X 10 = 90

table_10.txt

10 X 1 = 10

10 X 2 = 20

10 X 3 = 30

10 X 4 = 40

10 X 5 = 50

10 X 6 = 60

10 X 7 = 70

10 X 8 = 80

10 X 9 = 90

10 X 10 = 100

table_11.txt

11 X 1 = 11

11 X 2 = 22

11 X 3 = 33

11 X 4 = 44

11 X 5 = 55

11 X 6 = 66

11 X 7 = 77

11 X 8 = 88

11 X 9 = 99

11 X 10 = 110

table_12.txt

12 X 1 = 12

12 X 2 = 24

12 X 3 = 36

12 X 4 = 48

12 X 5 = 60

12 X 6 = 72

12 X 7 = 84

12 X 8 = 96

12 X 9 = 108

12 X 10 = 120

table_13.txt

13 X 1 = 13

13 X 2 = 26

13 X 3 = 39

13 X 4 = 52

13 X 5 = 65

13 X 6 = 78

13 X 7 = 91

13 X 8 = 104

13 X 9 = 117

13 X 10 = 130

table_14.txt

14 X 1 = 14

14 X 2 = 28

14 X 3 = 42

14 X 4 = 56

14 X 5 = 70

14 X 6 = 84

14 X 7 = 98

14 X 8 = 112

14 X 9 = 126

14 X 10 = 140

table_15.txt

15 X 1 = 15

15 X 2 = 30

15 X 3 = 45

15 X 4 = 60

15 X 5 = 75

15 X 6 = 90

15 X 7 = 105

15 X 8 = 120

15 X 9 = 135

15 X 10 = 150

table_16.txt

16 X 1 = 16

16 X 2 = 32

16 X 3 = 48

16 X 4 = 64

16 X 5 = 80

16 X 6 = 96

16 X 7 = 112

16 X 8 = 128

16 X 9 = 144

16 X 10 = 160

table_17.txt

17 X 1 = 17

17 X 2 = 34

17 X 3 = 51

17 X 4 = 68

17 X 5 = 85

17 X 6 = 102

17 X 7 = 119

17 X 8 = 136

17 X 9 = 153

17 X 10 = 170

table_18.txt

18 X 1 = 18

18 X 2 = 36

18 X 3 = 54

18 X 4 = 72

18 X 5 = 90

18 X 6 = 108

18 X 7 = 126

18 X 8 = 144

18 X 9 = 162

18 X 10 = 180

table_19.txt

19 X 1 = 19

19 X 2 = 38

19 X 3 = 57

19 X 4 = 76

19 X 5 = 95

19 X 6 = 114

19 X 7 = 133

19 X 8 = 152

19 X 9 = 171

19 X 10 = 190

table_20.txt

20 X 1 = 20

20 X 2 = 40

20 X 3 = 60

20 X 4 = 80

20 X 5 = 100

20 X 6 = 120

20 X 7 = 140

20 X 8 = 160

20 X 9 = 180

20 X 10 = 200

The-Ultimate-Python-Course-main