Check Phone Number

```
--------------------
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Thu Mar  9 04:19:57 2023

@author: Prabodh C P
"""
import re


def isphonenumber(numStr):
    if len(numStr) != 12:
        return False
    for i in range(len(numStr)):
        if i==3 or i==7:
            if numStr[i] != "-":
                return False
        else:
            if numStr[i].isdigit() == False:
                return False
    return True


def chkphonenumber(numStr):
    ph_no_pattern = re.compile(r'^\d{3}-\d{3}-\d{4}$')
    if ph_no_pattern.match(numStr):
        return True
    else:
        return False

ph_num = input("Enter a phone number : ")
print("Without using Regular Expression")
if isphonenumber(ph_num):
    print("Valid phone number")
else:
    print("Invalid phone number")

print("Using Regular Expression")
if chkphonenumber(ph_num):
    print("Valid phone number")
else:
    print("Invalid phone number")
```

```
-----------------------------------
Search Phone Number & Email
-----------------------------------
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Thu Mar  9 04:40:10 2023

@author: Prabodh C P
"""

import re

# Define the regular expression for phone numbers
phone_regex = re.compile(r'\+\d{12}')
email_regex = re.compile(r'[A-Za-z0-9._]+@[A-Za-z0-9]+\.[A-Z|a-z]{2,}')
# Open the file for reading
with open('example.txt', 'r') as f:
    # Loop through each line in the file
    for line in f:
        # Search for phone numbers in the line
        matches = phone_regex.findall(line)
        # Print any matches found
        for match in matches:
            print(match)

        matches = email_regex.findall(line)
        # Print any matches found
        for match in matches:
            print(match)
```

File Operations

```
----------------
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Thu Mar  9 05:26:33 2023

@author: Prabodh C P
"""

import os.path
import sys

fname = input("Enter the filename : ")

if not os.path.isfile(fname):
    print("File", fname, "doesn't exists")
    sys.exit(0)

infile = open(fname, "r")

lineList = infile.readlines()

for i in range(20):
    print(i+1, ":", lineList[i])

word = input("Enter a word : ")
cnt = 0
for line in lineList:
    cnt += line.count(word)

print("The word", word, "appears", cnt, "times in the file")
```

Zip operation on a folder

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Fri Dec 23 16:14:28 2022

@author: Prabodh C P
"""

import os
import sys
import pathlib
import zipfile

dirName = input("Enter Directory name that you want to backup : ")

if not os.path.isdir(dirName):
    print("Directory", dirName, "doesn't exists")
    sys.exit(0)

curDirectory = pathlib.Path(dirName)

with zipfile.ZipFile("myZip.zip", mode="w") as archive:
    for file_path in curDirectory.rglob("*"):
        archive.write(file_path, arcname=file_path.relative_to(curDirectory))

if os.path.isfile("myZip.zip"):
    print("Archive", "myZip.zip", "created successfully")
else:
    print("Error in creating zip archive")
```

Inheritance

```
import math
class Shape:
    def __init__(self):
        self.area = 0
        self.name = ""

    def showArea(self):
        print("The area of the", self.name, "is", self.area, "units")

class Circle(Shape):
    def __init__(self,radius):
        self.area = 0
        self.name = "Circle"
        self.radius = radius

    def calcArea(self):
        self.area = math.pi * self.radius * self.radius

class Rectangle(Shape):
    def __init__(self,length,breadth):
        self.area = 0
        self.name = "Rectangle"
        self.length = length
        self.breadth = breadth

    def calcArea(self):
        self.area = self.length * self.breadth

class Triangle(Shape):
    def __init__(self,base,height):
        self.area = 0
        self.name = "Triangle"
        self.base = base
        self.height = height

    def calcArea(self):
        self.area = self.base * self.height / 2



c1 = Circle(5)
c1.calcArea()
c1.showArea()
r1 = Rectangle(5, 4)
r1.calcArea()
r1.showArea()
t1 = Triangle(3, 4)
t1.calcArea()
t1.showArea()
```

Employee Details

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Thu Mar  9 12:09:50 2023

@author: Prabodh C P
"""

class Employee:
    def __init__(self):
        self.name = ""
        self.empId = ""
        self.dept = ""
        self.salary = 0

    def getEmpDetails(self):
        self.name = input("Enter Employee name : ")
        self.empId = input("Enter Employee ID : ")
        self.dept = input("Enter Employee Dept : ")
        self.salary = int(input("Enter Employee Salary : "))

    def showEmpDetails(self):
        print("Employee Details")
        print("Name : ", self.name)
        print("ID : ", self.empId)
        print("Dept : ", self.dept)
        print("Salary : ", self.salary)

    def updtSalary(self):
        self.salary = int(input("Enter new Salary : "))
        print("Updated Salary", self.salary)


e1 = Employee()
e1.getEmpDetails()
e1.showEmpDetails()
e1.updtSalary()
```

Polymorphism and Inheritance

```
class PaliStr:
    def __init__(self):
        self.isPali = False

    def chkPalindrome(self, myStr):
        if myStr == myStr[::-1]:
            self.isPali = True
        else:
            self.isPali = False

        return self.isPali

class PaliInt(PaliStr):
    def __init__(self):
        self.isPali = False

    def chkPalindrome(self, val):
        temp = val
        rev = 0
        while temp != 0:
            dig = temp % 10
            rev = (rev*10) + dig
            temp = temp //10

        if val == rev:
            self.isPali = True
        else:
            self.isPali = False

        return self.isPali

st = input("Enter a string : ")

stObj = PaliStr()
if stObj.chkPalindrome(st):
    print("Given string is a Palindrome")
else:
    print("Given string is not a Palindrome")

val = int(input("Enter a integer : "))

intObj = PaliInt()
if intObj.chkPalindrome(val):
    print("Given integer is a Palindrome")
else:
    print("Given integer is not a Palindrome")
```
