Day Objectives

- Day Objectives
  - Regular Expressions
    - Constructing Regular Expressions for various use cases
    - Regular Expressions Module and related in Python
    - Improving the Contacts application with name and phone number validation using regular expressions
  - File Handling
    - Text Files
    - Upgrading the Contacts Application to store contact information in a text file

Regular Expressions

- Pattern Matching
- Symbolic Notation of a pattern
  - Pattern: Format which repeats
  - Pattern(RE)- Represents the se of all values that matches the pattern
- [0-9]----> Any digit
- [a-z]----> Any lower case character or alphabet
- [8642]---->All single digit multiples of 2
- [2486]---->Order may be in Any form
- ^[0-9]{1}$--->Only single digit number 1
- ^[0-9]{2}$--->Only two digit numbers 34
- ^[a-z]{2}$---> Only two alphabets gh
- ^[1-9][0-9]*[05]$ ----> All Multiples of 5

```
- Using the or ^[1-9][0-9]*[05]$|^([5])$
```

- ^[0-9]{10}$ -----> All 10 digit numbers

- [w][o][r][d] or (word) ----> Searching for a 'word'
- ^[6-9]{1}[0-9]{9}$ ------> Valid phone number in India

```
- ^[6-9][0-9]{9}$|^[0][6-9][0-9]{9}|^[+][9][1][6-9][0-9]{9}$
```

- Email Validation(username@domain extension)
  - username
    - Length of suername : [6,15] ^[0-9a-z][0-9a-z_.]{4,13}[0-9a-z]$
    - No special characters other than _.
    - Should not begin and end with _.
    - Characters Set : all digits and lower case alphabet
  - domain
    - Length of domain : [3,18]
    - No Special Characters
    - Character Set: all digits and lower case alphabet
  - extension

- Length of extension : [2,4]
- No special characters
- Character Set : alphabet

EmailExtension

- **^[0-9a-z][0-9a-z_.]{4,13}[@][0-9a-z]{3,18}[.][a-z]{2,4}|[.][a-z]{2,4}$**
- ^[a]...[z]$ -----> Any string of length 5 that starts with 'a' and ends with 'z'
  - ^[a].*[z]$ -----> Any string of length of any that starts with 'a' and ends with 'z'

In [ ]:
```
1
```

Validating the Phone number

In [11]:
```
# Function to Validate a phone nuber

import re

def phoneNumberValidator(n):
    pattern='^[6-9][0-9]{9}$|^[0][6-9][0-9]{9}|^[+][9][1][6-9][0-9]{9}$'
    if re.match(pattern,str(n)):
        return True
    return False
n=input()
phoneNumberValidator(n)


```

```
dfref
```

Out[11]:
```
True
```

In [12]:
```
def emailValidator(email):
    pattern='^[0-9a-z][0-9a-z_.]{4,13}[@][0-9a-z]{3,18}[.][a-z]{2,4}|[.][a-z
    if re.match(pattern,email):
        return True
    return False
email= input()
emailValidator("syamaladevi241@gmail.com")

```

Out[12]:
```
True
```

In [35]:
```
contacts={"syamu":[8331063380,'name@domain.ext'],"name1":[8331063380,'name1@
def addContact(name,phone,email):
    # Verify the contact doesnot already exist
    if name in contacts:
        print(name,"already exists.")
    else:
        if not phoneNumberValidator(phone):
            print("Invalid Phone number")
        if not emailValidator(email):
            print("Invalid Email address")
            return
        newcontact =[]
        newcontact.append(phone)
        newcontact.append(email)
        contacts[name]=newcontact
        print(name,"added Succesfully")
    return
name=input()
phone=int(input())
email=input()
#print(contacts)
addContact(name,phone,email)
```

```
friends
34567890
syamala23@gmail.com
Invalid Phone number
friends added Succesfully
```

In [16]:
```
def searchContacts(name):
    if name in contacts:
        print(name)
        print("Phone:",contacts[name][0])
        print("Email:",contacts[name][1])
    else:
        print("%s does not exist" % name)
    return
name=input()
searchContacts(name)
```

```
syamu
syamu
Phone: 8331063380
Email: name@domain.ext
```

In [37]:
```
def importContacts(newConctacts):
    contacts.update(newContacts)
    print(len(newContacts.keys()),"contacts  added successfully")
    return
name2=input()
name3=input()
phone2=input()
phone3=input()
newContacts={"syamu":[8331063380,'name@domain.ext'],"name1":[8331063380,'nam
importContacts(newContacts)
```

```
ojiof
mfogjior
3456789
3456789
2 contacts  added successfully
```

In [39]:
```
newContacts
```

Out[39]:
```
{'syamu': [8331063380, 'name@domain.ext'],
 'name1': [8331063380, 'name1@domain.ext']}
```

In [40]:
```
def listAllContacts():
    for contact,info in contacts.items():
        print(contact,"\n","Phone:",info[0],"\n","Email:",info[1])
    return
listAllContacts()
```

```
syamu
 Phone: 8331063380
 Email: name@domain.ext
name1
 Phone: 8331063380
 Email: name1@domain.ext
friends
 Phone: 34567890
 Email: syamala23@gmail.com
```

In [ ]:
```
def edit(name,phone,email):
```

In [ ]:
```

```

In [1]:
```
contacts={"syamu":[8331063380,'name@domain.ext'],"name1":[8331063380,'name1@
def addContact(name,phone):
    # Verify the contact doesnot already exist
    if name not in contacts and phoneNumberValidator(phone):
        contacts[name]=phone
        print("Contact %s Added" % name)
    if name in contacts:
        print("contact %s already exists" % name)
    #elif not phoneNumberValidator(phone):
        #print("Invalid phone numberr")
    return
name=input()
phone=int(input())
#print(contacts)
addContact(name,phone)
```

```
syamu
8331063380
contact syamu already exists
```

In [ ]:
```

```

In [95]:
```
def searchContacts(name):
    if name in contacts:
        print(name,"exists",contacts[name])
    else:
        print("%s does not exist" % name)
    return
name=input()
searchContacts(name)
```

```
syamu
syamu exists [8331063380, 'name@domain.ext']
```

In [22]:
```
def importContacts(newConctacts):
    contacts.update(newContacts)
    print(len(newContacts.keys()),"contacts  added successfully")
    return
name2=input()
name3=input()
phone2=input()
phone3=input()
newContacts={name2:phone2,name3:phone3}
importContacts(newContacts)
```

```
dfdjf
fdfjio
8331063380
9075347882
2 contacts  added successfully
```

In [ ]:
```
def removeContacts(name):
    if name in contacts:
        contacts.pop(name)
        print("%s removed"% name)
    else:
        print("%s not does not exists "% name)
    return
name=input()
removeContacts(name)
```

In [23]:
```
newContacts
```

Out[23]:
```
{'dfdjf': '8331063380', 'fdfjio': '9075347882'}
```

In [ ]:
```

```

Ali and help his innocent friends

In [13]:
```
s=input()
if(len(s)>9):
    print("invalid")
else:
    if (s[2]!="A" ) and (s[2]!= "E") and (s[2]!="i" )and (s[2]!="O") and (s[
        if((int(s[0])+int(s[1]))%2==0 and (int(s[3])+int(s[4]))%2==0 and (in
            print("valid")
        else:
            print("invalid")
    else:
        print("valid")
```

```
12X345-78
invalid
```

In [15]:
```
s=input()
if(len(s)>9):
    print("invalid")
else:
    if ((s[2]!="A")  and (s[2]!= "E") and (s[2]!="I" )and (s[2]!="O") and (s
        if((int(s[0])+int(s[1]))%2==0 and (int(s[3])+int(s[4]))%2==0 and (in
            print("valid")
        else:
            print("invalid")
    else:
        print("valid")
```

```
13Y357-22
valid
```

Ali and Helping innocent people

In [16]:
```
# Determine yes or no Ali and Helping innocent people
s=input()
if(len(s)>9):
    print("invalid")
else:
    if ((s[2]!="A")  and (s[2]!= "Y") and (s[2]!="I" )and (s[2]!="O") and (s
        print("valid")
    else:
        print("invalid")

```

```
13Y35-768
invalid
```

Ladderophilia

In [96]:
```
# Generate the Pattern of the following
n=int(input())
for i in range(n):
    for j in range(2):
        print("*",end="    ")
        print("*")
    print("*****")
for k in range(2):
    print("*",end="    ")
    print("*")
```

```
4
*   *
*   *
*****
*   *
*   *
*****
*   *
*   *
*****
*   *
*   *
*****
*   *
*   *
```

In [ ]:
```

```

File Handling in Python

File - Document containing information residing on the permanent storage

Types- Text,PDF,CSV etc

File I/O Channelling I/O -data to files for Reading and writing

Read a file - Input from file

Write to a file - Output to a file

Read a file - open (filename,mode)

Read/Write file -open(filename,mode)

Reading A File

In [52]:
```
# Function to read a file

def readFile(filename):
    f=open(filename,'r')
    filedata=f.read()
    f.close()
    return filedata
filename='Data Files\data.txt'
readFile(filename).split('\n')

```

Out[52]:
```
['Line1', 'Line2', 'Line3', '']
```

Reading a File in another way

In [48]:
```
# Function to read a file

def readFile(filename):
    f=open(filename,'r')
    filedata=f.read()
    f.close()
    return filedata
filename='Data Files\data.txt'
filedata=readFile(filename)
#readFile(filename).split('\n')
for line in readFile(filename).split('\n'):
    print(line)

```

```
Line1
Line2
Line3
```

Reading the data in files in anoter way

In [57]:

```
# Define file data

# Function to read a file

def readFile(filename):
    f=open(filename,'r')
    filedata=f.read()
    f.close()
    return filedata
filename='Data Files\data.txt'
filedata=readFile(filename)


def printFileDataLines(filename):
    f=open(filename,'r')
    for line in f:
        print(line,end='')
    return
printFileDataLines(filename)

print(readFile(filename))
```

```
Line1
Line2
Line3
Line1
Line2
Line3
```

In [61]:
```
# Define file data

# Function to read a file

def readFile(filename):
    f=open(filename,'r')
    filedata=f.read()
    f.close()
    return filedata
filename='Data Files\data.txt'
filedata=readFile(filename)


def printFileDataLines(filename):
    with open(filename,'r') as f:
        for line in f:
            print(line,end='')
    return
printFileDataLines(filename)

#print(readFile(filename))
```

```
Line1
Line2
Line3
Line1
Line2
Line3
```

Write data into a file

In [91]:
```
# Function to write data into a file

def writeINTOFile(filename,filedata):
    with open(filename,'w') as f:
        f.write(filedata)
    return
filename='Data Files/data.txt'

writeINTOFile(filename,"new data")

```

In [ ]:
```

```

In [90]:
```
def appendDataToFile(filename,filedata):
    with open(filename,'a') as f:
        f.writelines(filedata)
    return
filedata=["Line2,Line3"]

appendDataToFile(filename,filedata)

```

In [ ]:
```
1
```

In [94]:
```
1 def appendDataToFile(filename,filedata):
2     with open(filename,'a') as f:
3         f.write(filedata)
4     return
5 filedata="\nLine2\nLine7"
6
7 appendDataToFile(filename,filedata)
```

In [93]:
```
1 def appendDataToFile(filename,filedata):
2     with open(filename,'a') as f:
3         f.writelines("\n"+line)
4     return
5 filedata='Data Files/data.txt'
6 appendDataToFile(filename,filedata)
7
```

In [ ]:
```
1
```