|  |  |
| --- | --- |
| **Ex No: 01** | **prime numbers** |
| **Date:** |

* **PRIME NUMBERS**
* **PROGRAM\_CODING:**

n=int(input("enter the upper limit:"))

print("prime numbers are")

for num in range(0,n+1):

#prime num are greater than 1

if num>1:

for i in range(2,num):

if(num%i)==0:

break

else:

print(num)

* PROGRAM OUtPUT

Enter the upper limit: 20

Prime numbers are

2

3

5

7

11

13

17

19

|  |  |
| --- | --- |
| **Ex No: 02** | **exponentiation of a number** |
| **Date:** |

* **EXPONENTIATION OF A NUMBER**
* **PROGRAM\_CODING:**

def power(base,exp):

if(exp==1):

return(base)

if(exp!=1):

return(base\*power(base,exp-1))

base=int(input("enter base:"))

exp=int(input("enter exponential value:"))

print("result:",power(base,exp))

* PROGRAM OutPUT

Enter base: 7

Enter exponential value: 2

Result:49

|  |  |
| --- | --- |
| **Ex No: 03** | **maximum in the given list of numbers** |
| **Date:** |

* **MAXIMUM in the given LIST OF NUMBERS**
* **PROGRAM\_CODING:**

l=[]

n=int(input("enter the upper limit:"))

for i in range(0,n):

a=int(input("enter the numbers:"))

l.append(a)

maxno=l[0]

for i in range(0,len(l)):

if l[i]>maxno:

maxno=l[i]

print("the maximum number is %d"%maxno)

* PROGRAM OUtPUT

Enter the upper limit 3

Enter the numbers 6

Enter the numbers 9

Enter the numbers 90

The maximum number is 90

|  |  |
| --- | --- |
| **Ex No: 04** | **Binary Search** |
| **Date:** |

* **BINARY SEARCH**
* **PROGRAM\_CODING:**

def binary\_search(list\_arr):

print(list\_arr)

#binary search algo...

search\_input=int(input("Enter the number to search:"))

start=0;

end=len(list\_arr)-1

mid=0

while (start<=end):

mid=(end+start)//2

if(list\_arr[mid]<search\_input):

start=mid+1

elif(list\_arr[mid]>search\_input):

end=mid-1

else:

return mid;

return -1

obj\_res=binary\_search([1,2,3,4,5,6,7,8,9,10])

print("The Element in the index at :'"+str(obj\_res)+"'" if (obj\_res!=-1)else "Element not present");

* PROGRAM OUtPUT

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

Enter the number to search:5

The Element in the index at :'4'

|  |  |
| --- | --- |
| **Ex No: 05** | **Linear Search** |
| **Date:** |

* **LINEAR SEARCH**
* **PROGRAM\_CODING:**

items=[5,7,10,12,15]

print("list of items is",items)

x=int(input("enter item to search:"))

i=flag=0

while i<len(items):

if items[i]==x:

flag=1

break

i=i+1

if flag==1:

print("items found at position:",i+1)

else:

print("items not found")

* PROGRAM OUPUT

(list of items is: [5, 7, 10, 12, 15] )

enter item to search: 7

(item found at position:, 2)

|  |  |
| --- | --- |
| **Ex No: 06** | **selection sort** |
| **Date:** |

* **SELECTION SORT**
* **PROGRAM\_CODING:**

def selectionSort(alist):

for i in range(len(alist)-1,0,-1):

pos=0

for location in range(1,i+1):

if alist[location]>alist[pos]:

pos=location

temp=alist[i]

alist[i]=alist[pos]

alist[pos]=temp

alist=[54,26,93,17,77,31,44,55,20]

selectionSort(alist)

print(alist)

* PROGRAM OUPUT

[17, 20, 26, 31, 44, 54, 55, 77, 93]

|  |  |
| --- | --- |
| **Ex No: 07** | **Radio button using UI with tkinter** |
| **Date:** |

* **Radio button using UI with tkinter**
* **PROGRAM\_CODING:**

from Tkinter import\*

class Tkinter\_button:

def \_\_init\_\_(self,root):

self.master=root

self.master.title("Tkinter Radiobutton");

self.master.geometry("100x100")

#labell declaration

self.label1=Label(self.master,text="This is a label");

self.label1.pack()

self.radio\_val=IntVar()

#radiobutton deceleration

self.R1=Radiobutton(self.master,text="Python",value=1,variable=self.radio\_val,command=lambda:get\_val() )

self.R1.pack()

self.R2=Radiobutton(self.master,text="Java",value=2,variable=self.radio\_val,command=lambda:get\_val() )

self.R2.pack()

#lable12 declaration

self.label2=Label(self.master);

self.label2.pack()

#logic

def get\_val():

if(self.radio\_val.get()==1):

self.label2.config(text="You selected: Python")

if(self.radio\_val.get()==2):

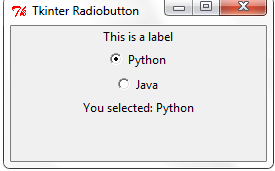
self.label2.config(text="You selected: Java")

self.master.mainloop();

if \_\_name\_\_=="\_\_main\_\_":

obj=Tkinter\_button(Tk());

* PROGRAM OUPUT



|  |  |
| --- | --- |
| **Ex No: 08** | **Application window using tkinter** |
| **Date:** |

* **Application window using tkinter**
* **PROGRAM\_CODING:**

from Tkinter import \*

class Tk\_TxtField:

def \_\_init\_\_(self,root):

self.master=root

self.master.title("Tkinter TextField:")

self.master.geometry("200x150")

#frontend

self.e1=StringVar()

self.e2=StringVar()

self.e3=StringVar()

self.e4=StringVar()

self.label1=Label(self.master, text="Num 1").grid(row=0,column=0)

self.entry1=Entry(self.master, textvariable=self.e1).grid(row=0,column=1)

self.label2=Label(self.master, text="Num 2").grid(row=1,column=0)

self.entry2=Entry(self.master, textvariable=self.e2).grid(row=1,column=1)

self.label3=Label(self.master, text="Operation").grid(row=2,column=0)

self.entry3=Entry(self.master, textvariable=self.e3).grid(row=2,column=1)

self.label4=Label(self.master, text="Result").grid(row=3,column=0)

self.entry4=Entry(self.master, textvariable=self.e4).grid(row=3,column=1)

self.b1=Button(self.master, text="=",padx=30,bd=4,command=lambda:self.operation()).place(x=20,y=100)

self.b2=Button(self.master, text="clear",padx=20,bd=4,command=lambda:self.clear()).place(x=100,y=100)

self.master.mainloop();

#logic area

def operation(self):

t1=self.e1.get()

t2=self.e2.get()

op=self.e3.get()

if(t1=="" and t2=="" and op==""):

print("please fill yhe textfield")

else:

self.e4.set(str(eval(""+str(t1)+""+str(op)+""+str(t2)+"")))

def clear(self):

self.e1.set("")

self.e2.set("")

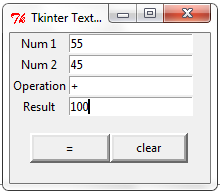
self.e3.set("")

self.e4.set("")

if \_\_name\_\_=="\_\_main\_\_":

obj=Tk\_TxtField(Tk());

* PROGRAM OUPUT



|  |  |
| --- | --- |
| **Ex No: 09** | **Gene SEQUENCE MINIMISING Using python** |
| **Date:** |

* **Gene SEQUENCE MINIMISING USING python**
* **PROGRAM\_CODING:**

def gene\_sequence\_mining(dna,base):

total=0

for i in dna:

if(base==i):

total+=1

return total

dna="ATGCGGCCTAT"

base="A"

print("The DNA:"+dna)

obj=gene\_sequence\_mining(dna,base)

print("The char '"+str(base)+"' has appeared'"+str(obj)+"'")

* PROGRAM OUPUT

The DNA:ATGCGGCCTAT

The char 'A' has appeared'2'

|  |  |
| --- | --- |
| **Ex No: 10** | **BIO- COMpUTING USING python** |
| **Date:** |

* **BIO COMpUTING USING python**
* **PROGRAM\_CODING:**

def base\_freq(dna):

acount=0

ccount=0

gcount=0

tcount=0

for i in dna:

if(i == 'A'):

acount+=1

elif(i == 'C'):

ccount+=1

elif(i == 'T'):

tcount+=1

elif(i == 'G'):

gcount+=1

print("Base frequencies of the sequence '%s':" %dna)

print("A: %0.2f, C: %0.2f, T: %0.2f, G: %0.2f" %(acount/float(len(dna)), ccount/float(len(dna)), tcount/float(len(dna)), gcount/float(len(dna))))

dna = 'ACCAGAGT'

base\_freq(dna)

* PROGRAM OUPUT

Base frequencies of the sequence 'ACCAGAGT':

A: 0.38, C: 0.25, T: 0.12, G: 0.25