

# Python Programming

## Lab Programs – 2019-20

### **Please Note:**

- This document contains all Python SEE lab programs along with sample output.
- This manual is created by referring the programs of document “SEE PYTHON SOLUTIONS by\_back\_bencher\_3”.
- The lab programs in this document are not to be considered as final.
- The will give the proper output and these are only for reference.
- The document is not an official copy. Creator of this document is not responsible if you don't get the proper output for the programs.
- Please execute these and test by yourself. If any mistakes are found those will be modified and new copy will be uploaded to google drive under “V Semester Study Materials”.

### **Prepared by:**

Shawn Linton Miranda  
4NM17CS164

# Index

## PART-A

S.No	Program Title	Page No
1	Monthly pay for employee	1
2	Net salary of employee	1-2
3	First n fibonacci numbers	2
4	Top 3 scorers and average marks	2-3
5	Find uppercase letters	3
6	Count of vowels	3
7	Removal of punctuations	4
8	Merging uppercase letters of strings	4
9	Binary search	4-5
10	Index of smallest element	5
11	Counting lines, words and characters	5-6
12	Processing student marks	6

## PART - B

S.No	Program Title	Page No
1	Rectangle objects	7
2	Bank account transactions	8-9
3	Counting marks of students using answer key	9-10
4	Simple calculator using Tkinter	10-11
5	Country details	11-12
6	Future value of investment	13
7	Search student details in database	13-14
8	Employee database	15-16

## PART - A

### 1. Monthly pay for employee

The finance department of a company wants to calculate the monthly pay of one of its employee. Monthly pay should be calculated as mentioned in the formula below and display all the employee details.

**Monthly Pay= No. of hours worked in a week \* Pay rate per hour \* No .of weeks in a Month.**

Write a function in Program with function to implement the problem..

#### Program:

```
def monthlyPay(week_hours, hourly_pay, weeks):  
    monthly_pay=week_hours*hourly_pay*weeks  
    return monthly_pay  
  
name=input("Enter employee name : ")  
id=input("Enter employee ID : ")  
week_hours=int(input("Enter weekly working hours : "))  
hourly_pay=int(input("Enter pay rate per hour : "))  
weeks=int(input("Enter total number of weeks : "))  
  
print("\nEmployee Details")  
print("Employee Name : ", name)  
print("Employee ID : ", id);  
print("Monthly Pay : ", monthlyPay(week_hours, hourly_pay, weeks))
```

#### Output:

```
Enter employee name : Rajesh  
Enter employee ID : 1234  
Enter weekly working hours : 25  
Enter pay rate per hour : 500  
Enter total number of weeks : 4  
  
Employee Details  
Employee Name : Rajesh  
Employee ID : 1234  
Monthly Pay : 50000
```

### 2. Net salary of employee

The finance department wants to calculate the monthly net pay of one of its employee by finding the income tax to be paid and the net salary after the income tax deduction. The employee should pay the income tax based on the following table: Display basic salary, allowances, gross pay, income tax and net pay.

#### Program:

```
name=input("Enter employee name : ")  
id=input("Enter employee ID : ")  
basic=float(input("Enter basic salary : "))  
allowances=float(input("Enter allowances : "))  
gross_sal=basic+allowances  
if gross_sal>20000:  
    income_tax=gross_sal*0.3  
elif gross_sal>10000:  
    income_tax=gross_sal*0.2  
elif gross_sal>5000:  
    income_tax=gross_sal*0.1
```

Gross Salary	Tax Percentage
Below 5,000	Nil
5,001 to 10,000	10%
10,001 to 20,000	20%
More than 20,000	30%

else:

income\_tax=0

net\_sal=gross\_sal-income\_tax

print("\nEmployee Details")

print("Employee Name : ",name)

print("Employee ID : ",id)

print("Basic Salary : ",basic)

print("Allowances : ",allowances)

print("Gross Salary : ",gross\_sal)

print("Income Tax : ",income\_tax)

print("Net Salary : ",net\_sal)

### Output:

Enter employee name : *Arun*

Enter employee ID : *111*

Enter basic salary : *50000*

Enter allowances : *1000*

Employee Details

Employee Name : *Arun*

Employee ID : *111*

Basic Salary : *50000.0*

Allowances : *1000.0*

Gross Salary : *51000.0*

Income Tax : *15300.0*

Net Salary : *35700.0*

### 3. First n fibonacci numbers

Write a Python program to generate first 'n' Fibonacci numbers.

#### Program:

fib1=0

fib2=1

n=int(input("Enter the number of fibonacci numbers to be printed : "))

if n>0:

print("First ",n," fibonacci numbers are : ")

print(fib1, end=" ") #end line with space instead of \n

print(fib2, end=" ")

for i in range(2,n):

fib3=fib1+fib2

print(fib3,end=" ")

fib1=fib2

fib2=fib3

else:

print("Invalid choice!")

#### Output:

Enter the number of fibonacci numbers to be printed : *10*

First 10 fibonacci numbers are :

0 1 1 2 3 5 8 13 21 34

### 4. Top 3 scorers and average marks

Given below is the list of marks scored by students. Find top three scorers for the course and also display the average marks scored by all students. Implement the solution using Python Programming.

#### Program:

marks={'John':86.5, 'Jack':91.2, 'Jill':84.5, 'Harry':72.1,'Joe':80.5}

sorted\_keys=sorted(marks, key=marks.get, reverse=True)

print("Top 3 scorers : ")

for i in sorted\_keys[:3]:

print(i, " : ",marks[i])

sum=0

for i in marks:

sum+=marks[i]

Student Name	Marks
John	86.5
Jack	91.2
Jill	84.5
Harry	72.1
Joe	80.5

```

avg=sum/5
print("\nTotal marks of all students : %.2f" %(sum))
print("Average of all marks = %.2f" %(avg))

```

### Output:

```

Top 3 scorers :
Jack : 91.2
John : 86.5
Jill : 84.5

Total marks of all students : 414.80
Average of all marks = 82.96

```

## 5. Find uppercase letters

Write a program to count the number of capital letters and display the position of each capital letter in a user entered string via keyboard.

### Program:

```

str=input("Enter a string : ");
letters=[]
position=[]
for i in range(len(str)):
    if(str[i].isupper()):
        letters.append(str[i])
        position.append(i)
print("Number of capitals letters : ",len(letters))
print("Letter --- Position")
for i in range(len(letters)):
    print(" ",letters[i],"\\t\\t\\t",position[i])

```

### Output:

```

Enter a string : Python LAB
Number of capitals letters : 5
Letter --- Position
P          0
N          5
L          7
A          8
B          9

```

## 6. Count of vowels

Write a program to count the number of each vowel in a given string.

### Program:

```

vowels={'a':0,'e':0,'i':0,'o':0,'u':0}
count=0
str=input("Enter a string : ")
for i in str.lower():
    if i in 'aeiou':
        vowels[i]=vowels[i]+1
        count=count+1

print("Total number of vowels : ",count)
print("Vowel occurrences : ")
for vow in vowels:
    if(vowels[vow]>0):
        print(" ",vow," --> ",vowels[vow]," times")

```

### Output:

```

Enter a string : This program counts number of vowels
Total number of vowels : 10
Vowel occurrences :
a --> 1 times
e --> 2 times
i --> 1 times
o --> 4 times
u --> 2 times

```

## 7. Removal of punctuations

Write a program to remove all punctuations like “’!()-[]{};:’’,\,<>./,?,@,#,\$,%^&\* \_~” from the string.

### Program:

```
str=input("Enter a string : ")
pmarks="\!()-[]{};:’’,\,<>./,?,@,#,$,%^&* _~" #some characters(“ ’ /) preceded with slash to escape their meaning
res=""
for i in str:
    if i not in pmarks:
        res+=i
print("Original string : ",str)
print("String after modification : ",res)
```

### Output:

```
Enter a string : Hi_I'm Learning!. (Python)
Original string : Hi_I'm Learning!. (Python)
String after modification : HiIm LearningPython
```

## 8. Merging uppercase letters of strings

Consider two strings, String1 and String2 and display the merged string as output. The merged string should be the capital letters from both the strings in the order they appear.

### Program:

```
str1=input("Enter string-1 : ")
str2=input("Enter string-2 : ")
res=""
for char in str1:
    if(char.isupper()):
        res+=char
for char in str2:
    if(char.isupper()):
        res+=char
print("Resultant String is ",res)
```

### Output:

```
Enter string-1 : AbCdE
Enter string-2 : fgHIj
Resultant String is ACEHI
```

## 9. Binary search

Write a binary search function which searches an item in a sorted list. The function should return the index of element to be searched in the list.

### Program:

```
def binarySearch(elements,key):
    low=0
    high=len(elements)-1
    while low<=high:
        mid=int((low+high)/2)
        if elements[mid]==key:
            return mid
        elif key<elements[mid]:
            high=mid-1
        else:
            low=mid+1
    return -1

list=[]
n=int(input("Enter the number of element : "))
for i in range(n):
    list.append(int(input("Element-%d : "%(i+1))))
key=int(input("Enter the element to be searched : "))
list.sort()
```

```

print("Sorted List : ",list);
pos=binarySearch(list,key)
if pos != -1 :
    print("Key found at position ",pos+1)
else:
    rint("Key not found!")

```

### Output:

```

Enter the number of element : 5
Element-1 : 7
Element-2 : 45
Element-3 : 09
Element-4 : 23
Element-5 : 1
Enter the element to be searched : 9
Sorted List : [1, 7, 9, 23, 45]
Key found at position 3

```

```

Enter the number of element : 5
Element-1 : 3
Element-2 : 2
Element-3 : 4
Element-4 : 2
Element-5 : 5
Enter the element to be searched : 6
Sorted List : [2, 2, 3, 4, 5]
Key not found!

```

## 10. Index of smallest element

Write a function that returns the index of the smallest element in a list of integers. If the number of such elements is greater than 1, return the smallest index. Use the header: `def indexOfSmallestElement(lst):`

Write a test program that prompts the user to enter a list of numbers, invokes this function to return the index of the smallest element and displays the index.

### Program:

```

def indexOfSmallestElement(lst):
    index=lst.index(min(lst))
    return index+1

n=int(input("Enter the number of elemets : "))
lst=[]
for i in range(n):
    lst.append(int(input("Element-%d : " %(i+1))))
print("Index of smallest element is :", indexOfSmallestElement(lst))

```

### Output:

```

Enter the number of elemets : 5
Element-1 : 34
Element-2 : 76
Element-3 : 86
Element-4 : 34
Element-5 : 2
Index of smallest element is : 5

```

## 11. Counting lines, words and characters

Write a program that will count the number of characters, words, and lines in a file. Words are separated by a white-space character. Your program should prompt the user to enter a filename.

### Program:

```

filename=input("Enter the file name : ")
try:
    file=open(filename,"r")
    lines=words=chars=0
    for i in file.readlines():
        lines+=1 #Increment line count
        for w in i.split(" "):
            if len(w)>0 and w!='\n': #To ignore empty words and \n
                words+=1
                chars+=len(w)

```

```

print("Number of lines : ",lines)
print("Number of words : ",words)
print("Number of characters : ",chars)
file.close()
except FileNotFoundError:
    print("Invalid filename!")

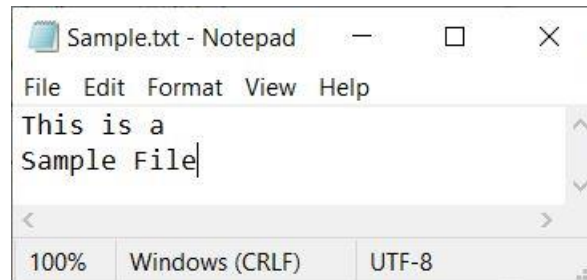
```

### Output:

```

Enter the file name : D:/Sample.txt
Number of lines : 2
Number of words : 5
Number of characters : 18

```



## 12. Processing student marks

Suppose that a text file contains marks for 6 courses for a student in a line. Each course marks is separated by space as delimiter. File contains marks for 'n' number of students in separate lines. Write a program that reads the marks from the file for each student and displays the total and average. Your program should prompt the user to enter a filename.

### Program:

```

filename=input("Enter the file name : ")
try:
    file=open(filename,"r")
    count=0
    print("Student | Total | Average")
    for line in file.readlines():
        count+=1 #count students
        score=0
        line=line.split(' ')
        for mark in line:
            score+=int(mark)
        print("\t%d \t %d \t %.2f" %(count,score, score/len(line)))
    file.close()
except FileNotFoundError:
    print("Invalid filename!")

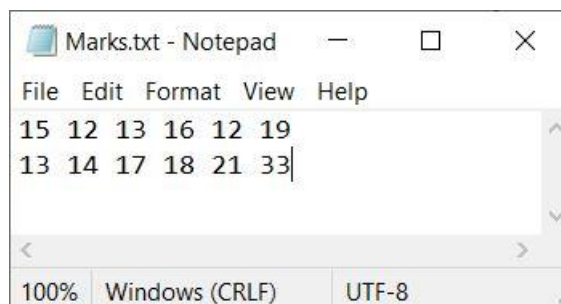
```

### Output:

```

Enter the file name : D:/Marks.txt
Student | Total | Average
1      87   14.50
2     116   19.33

```





## PART - B

### 1. Rectangle objects

Design a class named *Rectangle* to represent a rectangle. class contains: Two data fields named *width* and *height*. A constructor that creates a rectangle with the specified width and height and a method *getArea()*. Write a program that creates 'n' number of *Rectangle* objects. Read values of width and height from the keyboard for each *Rectangle*. Display the width and height of the rectangle having maximum and minimum area.

#### Program:

```
class Rectangle():
    def __init__(self, width, height):
        self.width = width
        self.height = height

    def getArea(self):
        return self.width * self.height

n = int(input("Enter the number of rectangles : "))
rectangles = []
for i in range(n):
    print("Rectangle-%d:" % (i + 1))
    width = int(input("Width : ")) #You can specify as float for reading decimal values
    height = int(input("Height : "))
    obj = Rectangle(width, height)
    rectangles.append(obj)

sorted_rects=sorted(rectangles,key=lambda x: x.getArea()) #Sort base on area

min_area = sorted_rects[0]
max_area = sorted_rects[n - 1]

print("\nMinimum area Rectangles :")
for i in rectangles:
    if (i.getArea() == min_area.getArea()):
        print("Height=", i.height, "\tWidth=", i.width, "\tArea=", i.getArea())

print("\nMaximum area Rectangles :")
for i in rectangles:
    if (i.getArea() == max_area.getArea()):
        print("Height=", i.height, "Width=", i.width, "Area=", i.getArea())
```

#### Output:

```
Enter the number of rectangles : 4
Rectangle-1:
Width : 2
Height : 2
Rectangle-2:
Width : 1
Height : 4
Rectangle-3:
Width : 2
Height : 8
```

```
Rectangle-4:
Width : 3
Height : 3

Minimum area Rectangles :
Height= 2   Width= 2   Area= 4
Height= 4   Width= 1   Area= 4

Maximum area Rectangles :
Height= 8 Width= 2 Area= 16
```

## 2. Bank account transactions

Design a class named Account that contains: A private int data field named accountno for the account. A private float data field named balance for the account. A constructor that creates an account with the specified accountno and Initial balance (default 100). A method named withdraws that withdraws a specified amount from the account. A minimum balance of 100 should be maintained for each account. A method named deposit that deposits a specified amount to the specific account.

Write a program that maintains 'n' number of Account objects with unique account no and supports following operations. a) Create account b) deposit c) withdraw d) Display account no with highest balance

### Program:

```
class Account():
    def __init__(self, accno):
        self.__accno=accno
        self.__bal=100
    def withdraw(self, amt):
        if(amt > (self.__bal-100)):
            print("Insufficient Balance.")
        else:
            self.__bal -= amt
            print(amt, "debited successfully.")
    def deposit(self, amt):
        self.__bal += amt
        print(amt, "credited successfully.")
    def getBalance(self):
        return self.__bal
    def getAccount(self):
        return self.__accno

def exists(account, accno): #this method is not a part of class
    flag=0
    for acc in account:
        if accno == acc.getAccount():
            flag=1
            break
    if flag==0:
        return None
    else:
        return acc

account=[]
while True:
    print("\n1.Create account  2.Withdraw  3.Deposit  4.Display Max Balance  5.Exit")
    choice=int(input("Enter your choice:"))
    if choice==1:
        accno=int(input("Enter the Acc No : "))
        if exists(account, accno) != None:
            print("Account already exists.")
        else:
            obj=Account(accno)
            account.append(obj)
            print("Account created successfully.")
    elif choice==2:
        accno=int(input("Enter the Acc No : "))
        amt=int(input("Enter the amount : "))
        acc=exists(account, accno)
        if acc==None:
            print("Account not found.");
```

```

else:
    acc.withdraw(amt)

elif choice==3:
    accno=int(input("Enter the Acc No : "))
    amt=int(input("Enter the amount : "))
    acc = exists(account, accno)
    if acc == None:
        print("Account not found.");
    else:
        acc.deposit(amt)

elif choice==4:
    accnt=account.sort(key=lambda x:x.getBalance(),reverse=True)
    print("Account with maximum balance:",account[0].getAccount(),"and balance is:",account[0].getBalance())
elif choice==5:
    break
else:
    print("Invalid choice!")

```

### Output:

```

1.Create account  2.Withdraw  3.Deposit  4.Display Max Balance  5.Exit
Enter your choice:1
Enter the Acc No : 123
Account created successfully.

1.Create account  2.Withdraw  3.Deposit  4.Display Max Balance  5.Exit
Enter your choice:1
Enter the Acc No : 123
Account already exists.

1.Create account  2.Withdraw  3.Deposit  4.Display Max Balance  5.Exit
Enter your choice:1
Enter the Acc No : 123
Account already exists.

1.Create account  2.Withdraw  3.Deposit  4.Display Max Balance  5.Exit
Enter your choice:3
Enter the Acc No : 123
Enter the amount : 5000
5000 credited successfully.

1.Create account  2.Withdraw  3.Deposit  4.Display Max Balance  5.Exit
Enter your choice:2
Enter the Acc No : 4000
Enter the amount : 4000
Account not found.

1.Create account  2.Withdraw  3.Deposit  4.Display Max Balance  5.Exit
Enter your choice:2
Enter the Acc No : 4000
Enter the amount : 4000
Account not found.

1.Create account  2.Withdraw  3.Deposit  4.Display Max Balance  5.Exit
Enter your choice:2
Enter the Acc No : 123
Enter the amount : 4000
4000 debited successfully.

1.Create account  2.Withdraw  3.Deposit  4.Display Max Balance  5.Exit
Enter your choice:4
Account with maximum balance is: 123 and balance is: 1100

1.Create account  2.Withdraw  3.Deposit  4.Display Max Balance  5.Exit
Enter your choice:5

```

### 3. Counting marks of students using answer key

Develop python program to perform the below mentioned operations.

a) display total marks scored by each student    b) Display top scorer and the top score.

Scenario: There are 8 students and answers to 10 multiple choice questions of each student is stored in a file called marks.txt. Each answer is delimited by space. Each line provides a student's answers to the questions, as shown below. The answer key is stored in a file named keys.txt with following format.

marks.txt - Notepad

```

File Edit Format View Help
Student0 A B A C C D E E A D
Student1 D B D C C D A E A D
Student2 E D D A C B E E A D
Student3 C B A E D C E E A D
Student4 A B D C C D E E A D
Student5 B B E C C D E E A D
Student6 B B A C C D E E A D
Student7 E B E C C D E E A D

```

keys.txt - Notepad

```

File Edit Format View Help
Key B B D C C D A E A D

```

**Program:**

```

try:
    answer=open("keys.txt","r").readline().split()
    file=open("marks.txt","r")
    max=0
    for marks in file.readlines():
        marks = marks.split()
        name = marks[0]
        count = 0
        for i in range(1, len(marks)):
            if marks[i] == answer[i]:
                count += 1
        print(name, " Marks : ", count)
        if count>max:
            max=count
            max_student=name

    print("\nStudent with maximum marks :")
    print(max_student," Marks : ",max)
except FileNotFoundError:
    print("File not found.");

```

**Output:**

```

Student0    Marks : 7
Student1    Marks : 10
Student2    Marks : 5
Student3    Marks : 4
Student4    Marks : 8
Student5    Marks : 7
Student6    Marks : 7
Student7    Marks : 7

```

```

Student with maximum marks :
Student1    Marks : 10

```

**4. Simple calculator using Tkinter**

*Develop python Tkinter program to perform arithmetic operations addition, subtraction, multiplication and division of two numbers. The numbers should be read using two different text fields and there should be four separate buttons for performing each operation. Result should be displayed in a separate message dialog box.*

**Program:**

```

from tkinter import *
from tkinter import messagebox
def addition():
    num1=float(t1.get())
    num2=float(t2.get())
    messagebox.showinfo("Result","The sum is %.2f" %(num1+num2))

def subtraction():
    num1=float(t1.get())
    num2=float(t2.get())
    messagebox.showinfo("Result","The difference is %.2f" %(num1-num2))

def multiplication():
    num1=float(t1.get())
    num2=float(t2.get())
    messagebox.showinfo("Result","The product is %.2f" %(num1*num2))

def division():
    num1=float(t1.get())
    num2=float(t2.get())
    messagebox.showinfo("Result","The quotient is %.2f" %(num1/num2))

window=Tk()
l1=Label(window,text="Enter the number 1")
l2=Label(window,text="Enter the number 2")
t1=Entry(window,width=10)
t2=Entry(window,width=10)
b1=Button(window,text="Add",command=addition)

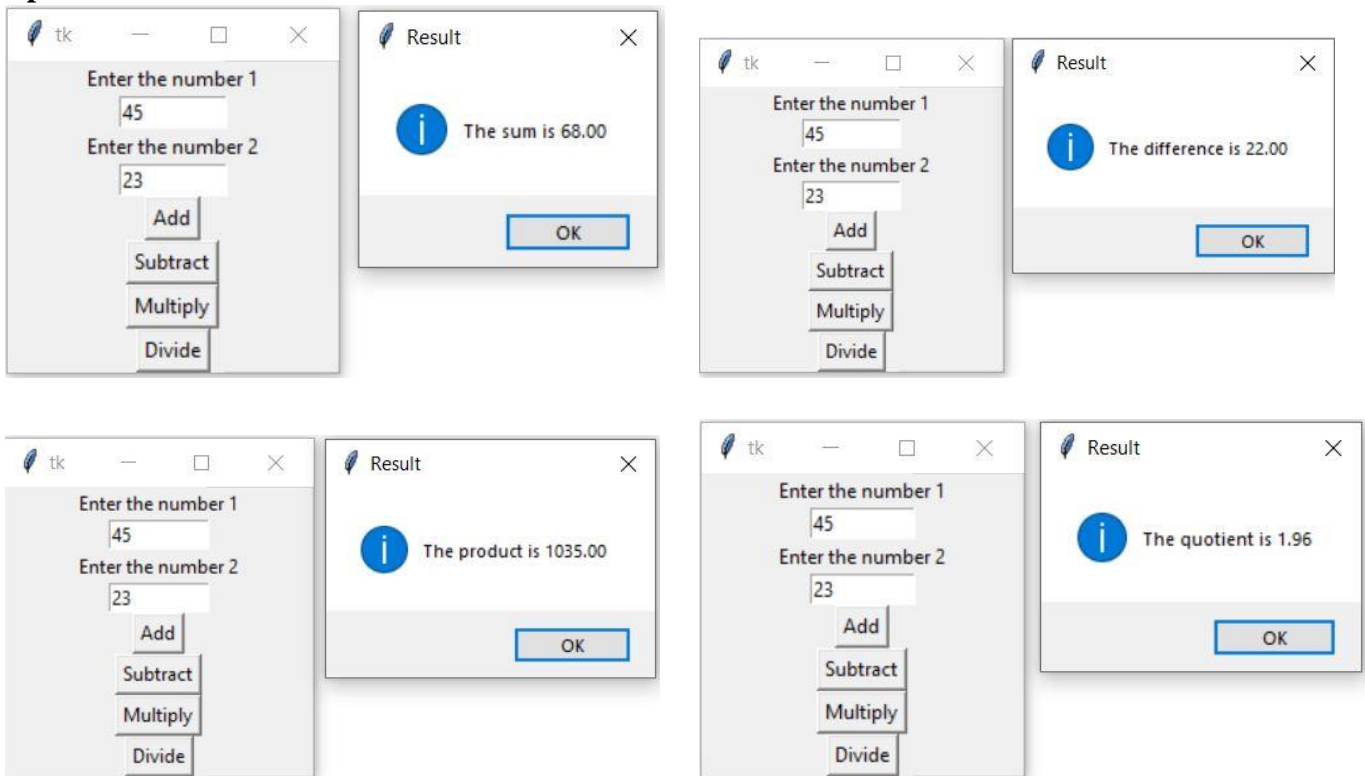
```

```

b2=Button(window,text="Subtract",command=subtraction)
b3=Button(window,text="Multiply",command=multiplication)
b4=Button(window,text="Divide",command=division)
l1.pack()
t1.pack()
l2.pack()
t2.pack()
b1.pack()
b2.pack()
b3.pack()
b4.pack()
window.mainloop()

```

### Output:



## 5. Country details

Write a program to create a list to maintain country names, respective capital and its population. The program should support following operations

- To enter country name, capital and its population.
- To accept the name of a country as an input and print the corresponding capital name and population as output. Otherwise, the program should print an appropriate message if the country is not found in the list.
- To display the country name with highest population.

### Program:

```

class Country:
    def __init__(self,name,capital,population):
        self.name=name
        self.capital=capital
        self.population=population

    def printDetails(self):
        print("Country:", self.name, " Capital:", self.capital, " Population:", self.population)

```

```

country=[]
while True:
    print("\n1.Add country 2.Display details 3.Display highest population 4.Exit");
    ch=int(input("Enter your choice : "))
    if ch==1:
        name=input("Enter country name : ")
        cap=input("Enter capital : ")
        pop=int(input("Enter the population : "))
        country.append(Country(name, cap, pop))
    elif ch==2:
        name=input("Enter country name : ")
        found=False
        for c in country:
            if c.name==name:
                c.printDetails()
                found=True
                break
        if found==False:
            print("Country not found!")
    elif ch==3:
        if len(country)==0:
            print("List doesnt contain any country details.")
            continue
        high=0
        for c in country: #to obtain highest population
            if c.population>high:
                high=c.population
        for c in country: #To get all countries which have highest population
            if c.population==high:
                c.printDetails()
    elif ch==4:
        break
    else:
        print("Invalid choice!")

```

### Output:

```

1.Add country 2.Display details 3.Display highest population 4.Exit
Enter your choice : 1
Enter country name : India
Enter capital : Delhi
Enter the population : 1250000000

```

```

1.Add country 2.Display details 3.Display highest population 4.Exit
Enter your choice : 1
Enter country name : China
Enter capital : Beijing
Enter the population : 1500000000

```

```

1.Add country 2.Display details 3.Display highest population 4.Exit
Enter your choice : 2
Enter country name : America
Country not found!

```

```

1.Add country 2.Display details 3.Display highest population 4.Exit
Enter your choice : 3
Country: China Capital: Beijing Population: 1500000000

```

```

1.Add country 2.Display details 3.Display highest population 4.Exit
Enter your choice : 4

```

## 6. Future value of investment

Write a program that calculates the future value of an investment at a given interest rate for a specified number of years. Formula : **futureValue=investmentAmount\*(1+monthlyInterestRate)years\*12**

Use textfields for users to enter the investment amount, years, and interest rate. Display the future amount in a label field when the user clicks the Calculate button.

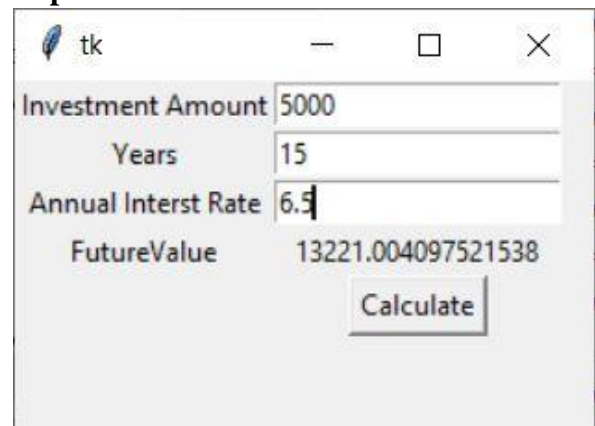
### Program:

```
from tkinter import *

def calc():
    i=eval(t1.get())
    y=eval(t2.get())
    r=eval(t3.get())/1200
    d=i*(r+1)**(y*12)
    t4.configure(text=str(d))

window=Tk()
window.geometry('250x150')
lbl1=Label(window,text="Investment Amount")
lbl1.grid(column=0,row=0)
lbl2=Label(window,text="Years")
lbl2.grid(column=0,row=1)
lbl3=Label(window,text="Annual Interest Rate")
lbl3.grid(column=0,row=2)
lbl4=Label(window,text="FutureValue")
lbl4.grid(column=0,row=3)
t1=Entry(window,width=20)
t1.grid(column=1,row=0)
t2=Entry(window,width=20)
t2.grid(column=1,row=1)
t3=Entry(window,width=20)
t3.grid(column=1,row=2)
t4=Label(window)
t4.grid(column=1,row=3)
b=Button(window,text="Calculate",command=calc)
b.grid(column=1,row=4)
window.mainloop()
```

### Output:



Investment Amount	5000
Years	15
Annual Interest Rate	6.5
FutureValue	13221.004097521538

Calculate

## 7. Search student details in database

Using Tkinter interface to display student details such as name, age, branch in text fields by performing search operation for the user entered usn via textfield. Information is displayed upon click on the search button. Students information is stored in a database by considering the table

Student(USN:String,Name:String,Age:Int,Branch:String).Display the success and failure message.

### Program:

```
from tkinter import *
from tkinter import messagebox
import pymysql

def search():
    db = pymysql.connect ("172.16.2.3","student","student","College") #Here college is a Database name
    cursor = db.cursor()
    sql="select * from student where usn='%s'"%(v1.get())
    try:
        cursor.execute(sql)
```



```

data=cursor.fetchall()
if len(data)==0:
    messagebox.showinfo("Error","USN does not exists.")
else:
    for row in data:
        name=row[1]
        age=row[2]
        branch=row[3]
        v2.set(name)
        v3.set(age)
        v4.set(branch)
        messagebox.showinfo("Success","USN found")
except:
    messagebox.showerror("Error","Unsuccessful")
db.close()

```

```

window=Tk()
v1=StringVar()
v2=StringVar()
v3=StringVar()
v4=StringVar()
window.geometry("250x190")
l1=Label(window,text="Enter the usn")
b=Button(window,text="Search",command=search)
l2=Label(window,text="Name")
l3=Label(window,text="Age")
l4=Label(window,text="Branch")
t1=Entry(window,width=20,textvariable=v1)
t2=Entry(window,width=20,textvariable=v2)
t3=Entry(window,width=10,textvariable=v3)
t4=Entry(window,width=10,textvariable=v4)
l1.grid(column=0,row=0)
l2.grid(column=0,row=2)
l3.grid(column=0,row=3)
l4.grid(column=0,row=4)
t1.grid(column=1,row=0)
t2.grid(column=1,row=2)
t3.grid(column=1,row=3)
t4.grid(column=1,row=4)
b.grid(column=1,row=1)
window.mainloop()

```

### Database:

```

mysql> create database College;
Query OK, 1 row affected (0.00 sec)

mysql> use College;
Database changed
mysql> create table student (
  -> usn varchar(10),
  -> name varchar(30),
  -> age int,
  -> branch varchar(10)
  -> );
Query OK, 0 rows affected (0.04 sec)

mysql> insert into student values('1234','Varun',20,'CS');
Query OK, 1 row affected (0.00 sec)

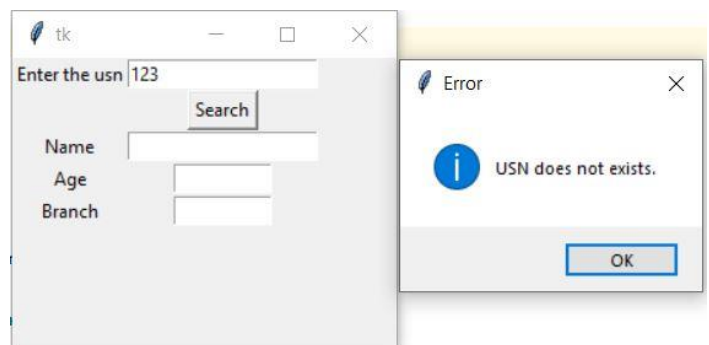
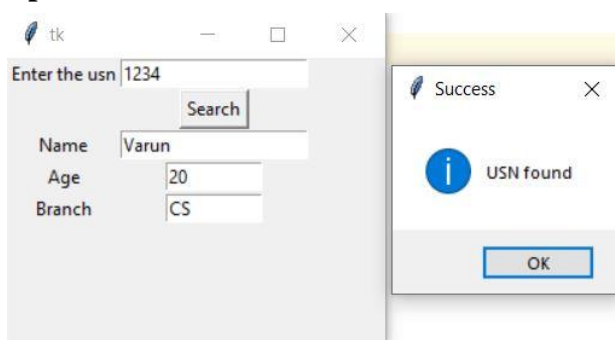
mysql> insert into student values('1235','Reshma',21,'CS
  -> ');
Query OK, 1 row affected (0.00 sec)

mysql> insert into student values('1236','Arjun',19,'CS');
Query OK, 1 row affected (0.00 sec)

mysql> select * from student;
+-----+-----+-----+-----+
| usn  | name  | age  | branch |
+-----+-----+-----+-----+
| 1234 | Varun | 20   | CS     |
| 1235 | Reshma | 21   | CS     |
|      |      |      |      |
| 1236 | Arjun | 19   | CS     |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)

```

### Output:





## 8. Employee database

Develop a python program to perform the following operations on a database by considering the table Employee (SSN: Int, Fname: String, LName: String, Age: Int, Place: String, Salary: Int).

Display the success and failure message.

a. Insert employee details      b. Delete a employee record      c. Update the employee details.

### Program:

```
import pymysql

def Insert():
    db = pymysql.connect ("172.16.2.3","student","student"," EmployeeDB ") #Here EmployeeDB is DB name
    cursor = db.cursor()
    ssn=int(input("Enter employee ssn : "));
    fname=input("Enter first name : ");
    lname=input("Enter last name : ");
    age=int(input("Enter the age : "));
    place=input("Enter the place : ");
    salary=int(input("Enter the salary : "));
    sql = "INSERT INTO Employee VALUES(%d,'%s','%s',%d,'%s',%d)" %(ssn,fname,lname,age,place,salary)
    try:
        cursor.execute(sql)
        db.commit()
        print("Successfully inserted.")
    except:
        print("Unsuccessful!")
        db.close()

def Delete():
    db = pymysql.connect ("172.16.2.3","student","student"," EmployeeDB ") #Here EmployeeDB is DB name
    cursor = db.cursor()
    ssn=int(input("Enter the SSN : "))
    sql = "DELETE FROM Employee WHERE ssn=%d" %(ssn)
    try:
        cursor.execute(sql)
        db.commit()
        print("Deleted Successfully.")
    except:
        print("Unsuccessful!")
        db.close()

def Update():
    db = pymysql.connect ("172.16.2.3","student","student"," EmployeeDB ") #Here EmployeeDB is DB name
    cursor = db.cursor()
    ssn=int(input("Enter the ssn : "));
    fname=input("Enter first name : ");
    lname=input("Enter last name : ");
    age=int(input("Enter the age : "));
    place=input("Enter the place : ");
    salary=int(input("Enter the salary : "));
    try:
        cursor.execute("update EMPLOYEE set fname='%s',lname='%s',age=%d,place='%s',salary=%d where
                                                                ssn=%d"%(fname,lname,age,place,salary,ssn))
        db.commit()
        print("Successfully updated")
```

```

except:
    print("Unsuccessful!")
    db.close()

while True:
    print("\n1.Insert\t2.Delete \t3.Update\t4.Exit")
    n=int(input("Enter the choice : "))
    if n==1:
        Insert()
    elif n==2:
        Delete()
    elif n==3:
        Update()
    elif n==4:
        break
    else:
        print("Invalid choice")

```

## Output:

```

1.Insert    2.Delete    3.Update    4.Exit
Enter the choice : 1
Enter employee ssn : 123
Enter first name : Sukhesh
Enter last name : Kumar
Enter the age : 45
Enter the place : Nitte
Enter the salary : 50000
Successfully inserted.

```

```

mysql> select * from employee;
+-----+-----+-----+-----+-----+-----+
| ssn | fname | lname | age | place | salary |
+-----+-----+-----+-----+-----+-----+
| 123 | Sukhesh | Kumar | 45 | Nitte | 50000 |
+-----+-----+-----+-----+-----+-----+

```

```

1.Insert    2.Delete    3.Update    4.Exit
Enter the choice : 3
Enter the ssn : 123
Enter first name : Sukhesh
Enter last name : Shetty
Enter the age : 46
Enter the place : Nitte
Enter the salary : 55000
Successfully updated

```

```

mysql> select * from employee;
+-----+-----+-----+-----+-----+-----+
| ssn | fname | lname | age | place | salary |
+-----+-----+-----+-----+-----+-----+
| 123 | Sukhesh | Shetty | 46 | Nitte | 55000 |
+-----+-----+-----+-----+-----+-----+

```

```

1.Insert    2.Delete    3.Update    4.Exit
Enter the choice : 2
Enter the SSN : 123
Deleted Successfully.

1.Insert    2.Delete    3.Update    4.Exit
Enter the choice : 4

```

```

mysql> select * from employee;
Empty set (0.00 sec)

```

