Python programming

LAB MANUAL

**Birla Vishwakarma Mahavidyalaya**

Rahul Mourya - 150073107007

|  |
| --- |
| **Practical 1: Python program to print Hello world!** |
| **Code:**  print "Hello, World" |
| Output: |

|  |
| --- |
| **Practical 2: Python program to demonstrate use of tuple, list and dictionary.** |
| **Code:**  #Practical:2 Program to demonstrate list, tuple and dictionary  list = [10, "20", 30, 40];  tuple = {"Rahul", 20, 1997};  dict = {'name': "Rahul", 'age': 20, 'DOY': 1997};  print("List:", list);  print("Tuple:", tuple);  print("Dictionary", dict['name']); |
| Output: |

|  |
| --- |
| **Practical 3: Python program to find Square root** |
| **Code:**  #Practical 3: Find square root  import math  a = input("\nEnter a number")  print(math.sqrt(a)) |
| Output: |

|  |
| --- |
| **Practical 4: Program to calculate the area of triangle** |
| **Code:**  #Practical 4: Program to find area of triangle  #Formulae AreaOfTriangle: 1/2 b \* h;  import math  breadth = input("Enter breadth")  height = input("Enter height")  result = 0.5 \* breadth \* height  print(result) |
| Output: |
|  |

|  |
| --- |
| **Practical 5: Program to solve quadratic equation** |
| **Code:**  #Practical 4: Program to find area of triangle  #Formulae AreaOfTriangle: 1/2 b \* h;  import math  breadth = input("Enter breadth")  height = input("Enter height")  result = 0.5 \* breadth \* height  print(result) |
| Output: |
|  |

|  |
| --- |
| **Practical 6: Program to swap two variables** |
| **Code:**  #Practical 6: Prgram to swap 2 variables  x = input("X:- ")  y = input("Y:- ")  choice = input("1. With temp variable \n2. Without temp variable")  if choice == 1:  #With temporary variable  x = x + y  y = x - y  x = x - y  else:  #without temporary variable  x = x \* y  y = x / y  x = x / y  print("X: ", x,"Y:", y) |
| Output: |
|  |

|  |
| --- |
| **Practical 7: Python program to generate random number** |
| **Code:**  #Practical 7: Program to generate random number  import os  import random  choice = input("1. Range \n2. Without Range\n")  if choice == 1:  lower\_range = input("Lower range: ")  upper\_range = input("Upper range: ")  step = input("Step: ")  print("Random Value: ", random.randrange(lower\_range, upper\_range, step))  else:  print("Random value: ", random.random()); |
| Output: |
|  |

|  |
| --- |
| **Practical 8: Python program to convert Celsius to Fahrenheit** |
| **Code:**  #Practical 8: Convert celsius to fahrenheit  #Celsius = Fahrenheit - 32 / 1.8  #Fahrenheit = celsius / 0.55 + 32  def convertCelcius():  x = input("Fahrenheit: ")  print("In celsius: ", (x - 32) / 1.8)  def convertFahrenheit():  x = input("Celsius: ")  print("In Fahrenheit", x / 0.55 + 32)  def main():  choice = input("1. Fahrenheit to Celsius \n2. Celsius to Fahrenheit\n")  if choice == 1:  convertCelcius()  else:  convertFahrenheit()  if \_\_name\_\_ == '\_\_main\_\_':  main() |
| Output: |
|  |

|  |
| --- |
| **Practical 9: Python program to check if a number is positive, negative or zero** |
| **Code:**  #Practical 9: Program to check number os positive, negative or zero  value = input("Enter a value: ")  value = int (value)  if value < 0:  print("Number is Negative\n")  elif value > 0:  print("Number is Positive\n")  else:  print("Number is zero\n") |
| Output: |
|  |

|  |
| --- |
| **Practical 10: Python program to check number is odd or even** |
| **Code:**  #Practical 10: Check number is odd or even  value = input('Enter value: ')  if value % 2 == 0:  print("Even \n")  else:  print("Odd \n") |
| Output: |
|  |

|  |
| --- |
| **Practical 11: Python program to check leap year** |
| **Code:**  #Practical 11: Find leap year  '''  function isLeapYear (year):  if ((year modulo 4 is 0) and (year modulo 100 is not 0))  or (year modulo 400 is 0)  then true  else false  '''  year = input("Enter a year: ")  if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):  print("Leap Year \n")  else:  print("Not leap year \n") |
| Output: |
|  |

|  |
| --- |
| **Practical 12: Python program to find largest among three number** |
| **Code:**  #Practical 12: Largest among three numbers  value = [-1, 0, 1]  value[0] = input("Enter three numbers: ")  value[1] = input()  value[2] = input()  value[0] = int(value[0])  value[1] = int(value[1])  value[2] = int(value[2])  max = 0  for i in range(0, 3):  if(int(max) < value[i]):  max = value[i]  print("Max value is: ", max) |
| Output: |
|  |

|  |
| --- |
| **Practical 13: Python program generate prime numbers between 0 to 1000** |
| **Code:**  #Practical 13: Generate prime numbers 1 to 1000  def is\_prime(n):  status = True  if n < 2:  status = False  else:  for i in range(2, n):  if n % i == 0:  status = False  return status  value = input("\nNumber Upto? ")  for n in range(1, value):  if is\_prime(n):  if n == value:  print n  else:  print n, ",", |
| **Output**: |
|  |

|  |
| --- |
| **Practical 14: Python program generate prime number in interval** |
| **Code:**  #Practical 13: Generate prime numbers in interval  def is\_prime(n):  status = True  if n < 2:  status = False  else:  for i in range(2, n):  if n % i == 0:  status = False  return status  value = input("\nNumber Upto? ")  for n in range(1, value):  if is\_prime(n):  if n == value:  print n  else:  print n, ",", |
| **Output**: |
|  |

|  |
| --- |
| **Practical 15: Python program find factorial of number** |
| **Code:**  #Practical 15: Find factorial of number  def getFactorial(n):  if n == 1 or n == 0:  return 1  else:  return (n \* getFactorial(n - 1))  value = input("Enter a number: ")  print(getFactorial(value)) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 16: Python program to display multiplication table** |
| **Code:**  #Practical 16: Multiplication table  choice = input("1. Range table \n2. Single table: ")  if choice == 1:  value = input('Upto? ')  for j in range(1, value + 1):  for i in range(1, 11):  print(j \* i)  print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\t")  else:  value = input("Show table of: ")  for i in range(1, 11):  print(value \* i) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 17: Python program to print Fibonacci series** |
| **Code:**  #Practical 16: Multiplication table  choice = input("1. Range table \n2. Single table: ")  if choice == 1:  value = input('Upto? ')  for j in range(1, value + 1):  for i in range(1, 11):  print(j \* i)  print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\t")  else:  value = input("Show table of: ")  for i in range(1, 11):  print(value \* i) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 18: Python program to print Armstrong number** |
| **Code:**  #Practical 18: Amstrong number in an interval  value = int(input("upto?: "))  for i in range(1, value):  \_temp = i  sum = 0  temp = i  while temp > 0:  digit = temp % 10  sum += digit \*\* 3  temp //= 10  if \_temp == sum:  print(\_temp) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 19: Python program to find sum of natural numbers** |
| **Code:**  #Practical 19: Sum of natural numbers  value = int(input("upto? "))  sum = 0  for i in range(1, value):  sum = value \* (value + 1) / 2  print(sum) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 20: Python program to convert decimal to binary** |
| **Code:**  #Practical 20: Decimal to binary conversion  value = int(input("Decimal value: "))  base\_2 = bin(value)  base\_8 = oct(value)  base\_16 = hex(value)  print("Binary: ", base\_2)  print("Octal: ", base\_8)  print("Hexadecimal:", base\_16) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 21: Python program to find factors of number** |
| **Code:**  #Practical 22: Factors of number  value = int(input("Number? "))  factor = []  for i in range(1, value):  if(value % i) == 0:  factor.append(i)  print(factor) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 23: Python program to make simple calculator** |
| **Code:**  #Practial 23: Simple calculator  def compute(x, \_a, \_b):  return {  1: \_a + \_b,  2: \_a \* \_b,  3: \_a / \_b,  4: \_a - \_b  #5: \_a \*\* \_b,  #6: \_a % \_b  }.get(x, 1)  choice = int(  input("1. Addition \n2.Multiplication \n3.Division \n4.Subtraction: "))  operand\_1 = int(input("Operand 1 value: "))  operand\_2 = int(input("Operand 2 value: "))  print(compute(choice, operand\_1, operand\_2)) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 24: Python program to display calendar** |
| **Code:**  #Practical 24: Calendar in python  import calendar  choice = int(input("1.Year \n2.Month: "))  if choice == 1:  year = input("Year: ")  for i in range(1, 12):  print(calendar.month(year, i))  if choice == 2:  year = input("Year: ")  month = input("Month: ")  print(calendar.month(year, month)) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 25: Python program to implement Fibonacci series using recursion** |
| **Code:**  #Practicals 25: Fibonacci series using recursion  def fibo\_recur(n):  if n <= 1:  return n  else:  return(fibo\_recur(n-1) + fibo\_recur(n-2))  nterms = 10  nterms = int(input("How many terms? "))  if nterms <= 0:  print("Plese enter a positive integer")  else:  print("Fibonacci sequence:")  for i in range(nterms):  print(fibo\_recur(i)) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 26: Python program to implement sum of natural number using recursion.** |
| **Code:**  #Practical 26: Sum of natural numbers using recursion  def sumOfNatural(x):  if x == 1:  return 1  else:  return (x + sumOfNatural(x-1))  value = int(input("Upto? "))  print(sumOfNatural(value)) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 27: Python program to implement factorial number using recursion.** |
| **Code:**  #Practicals 25: Fibonacci series using recursion  def factorial(x):  if x == 0 or x == 1:  return 1  else:  return (x \* factorial(x-1))  value = int(input("Upto? "))  print(factorial(value)) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 28: Python program to add two matrices** |
| **Code:**  #Practicals 25: Fibonacci series using recursion  def factorial(x):  if x == 0 or x == 1:  return 1  else:  return (x \* factorial(x-1))  value = int(input("Upto? "))  print(factorial(value)) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 29: Python program to add two matrices** |
| **Code:**  #Practicals 25: Fibonacci series using recursion  def factorial(x):  if x == 0 or x == 1:  return 1  else:  return (x \* factorial(x-1))  value = int(input("Upto? "))  print(factorial(value)) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 30: Python program to add two matrices** |
| **Code:**  #Practicals 25: Fibonacci series using recursion  def factorial(x):  if x == 0 or x == 1:  return 1  else:  return (x \* factorial(x-1))  value = int(input("Upto? "))  print(factorial(value)) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 31: Python program to display different patterns** |
| **Code:**  #Practical 31: Patterns in python  def triangle(\_depth, \_type):  k = 2 \* \_depth - 2  for i in range(0, \_depth):  for j in range(0, k):  print(end=" ")  k = k - 1  for j in range(0, i+1):  print(\_type, end=" ")  print("\r")  def rightTriangle(\_depth, \_type):  for i in range(0, \_depth):  for j in range(0, i+1):  print(\_type, end=" ")  print("\r")  def leftTriangle(\_depth, \_type):  k = 2 \* \_depth - 2  for i in range(0, \_depth):  for j in range(0, k):  print(end=" ")  k = k - 2  for j in range(0, i+1):  print(\_type, end=" ")  print("\r")  \_choice = int(input(  "1. Triangle \n2.Right-angle triangle - right \n3.Right-angle triangle - left\n"))  \_depth = int(input("Enter depth: "))  \_type = input("pattern character: ")  if \_choice == 1:  triangle(\_depth, \_type)  elif \_choice == 2:  rightTriangle(\_depth, \_type)  else:  leftTriangle(\_depth, \_type) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 32: Python program to check weather string is palindrome or not** |
| **Code:**  #Practical 32: Check palindrome of string  value = input("Enter string: ")  temp = value[::-1]  if value == temp:  print("Palindrome 1")  else:  print("Not Palindrome -1") |
| **Output**: |
|  |

|  |
| --- |
| **Practical 33: Python program to sort word in alphabetic order** |
| **Code:**  #Practical 33: String sorting in alphabetic order  string = input("Enter string: ")  print(sorted(string)) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 34: Python program to illustrate different set operations** |
| **Code:**  #Practical 33: String sorting in alphabetic order  string = input("Enter string: ")  print(sorted(string)) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 35: Python program to count the numbers of each vowel** |
| **Code:**  #Practical 35: Count numbers of vowels  #https://stackoverflow.com/questions/19967001/count-vowels-in-string-python  sentence = input("Sentence: ")  counts = { i : 0 for i in 'aeiouAEIOU'}  for char in sentence:  if char in counts:  counts[char] += 1    for k,v in counts.items():  print(k, v) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 36: Python program to find hah of file** |
| **Code:**  #Practical 36: Find hash of file  import sys  import hashlib  BUF\_SIZE = 65536  md5 = hashlib.md5()  sha1 = hashlib.sha1()  with open(sys.argv[1], 'rb') as f:  while True:  data = f.read(BUF\_SIZE)  if not data:  break  md5.update(data)  sha1.update(data)  print("MD5: {0}".format(md5.hexdigest()))  print("SHA1: {0}".format(sha1.hexdigest())) |
| **Output**: |
|  |

|  |
| --- |
| **Practical 37: Python program to demonstrate file handling operations** |
| **Code:**  #Practical 37: File handling  #To open a text file, use:  fh = open("hello.txt", "r")  #To read a text file, use:  fh = open("hello.txt","r")  print fh.read()  #To read one line at a time, use:  fh = open("hello.txt", "r")  print fh.readline()  #To read a list of lines use:  fh = open("hello.txt.", "r")  print fh.readlines()  #To write to a file, use:  fh = open("hello.txt","w")  fh.write("Hello World")  fh.close()  #To write to a file, use:  fh = open("hello.txt", "w")  lines\_of\_text = ["a line of text", "another line of text", "a third line"]  fh.writelines(lines\_of\_text)  fh.close()  #To append to file, use:  fh = open("Hello.txt", "a")  fh.write("Hello World again")  fh.close  #To close a file, use  fh = open("hello.txt", "r")  print fh.read()  fh.close() |
| **Output**: |
|  |

|  |
| --- |
| **Practical 38: Python program to demonstrate exception handling** |
| **Code:**  #Pratical 38: Program to handle exceptions  value\_1 = int(input("1st value: "))  value\_2 = int(input("2nd value: "))  try:  value\_1 //= value\_2  except Exception:  value\_1 = 'NULL'  print("Can't divide with zero")  else:  print("no errors: printing ")  finally:  print(value\_1, "\n") |
| **Output**: |
|  |