RAMAKRISHNA MISSION RESIDENTIAL COLLEGE

NAME: Rishav Nath Pati

ROLL: CSUG/194/18

SUB: Python(LNB)

QUESTION:
1. Add two numbers.

CODE:
```
a = int(input("Enter 1st number: "))
b = int(input("Enter 2nd number: "))
c = a+b
print("Sum is: ", c)
```

OUTPUT:
```
Enter 1st number: 3
Enter 2nd number: 4
Sum is:  7
```

QUESTION:
2. Find out the area and perimeter of a rectangle.

CODE:
```
length = int(input("Enter length: "))
breadth = int(input("Enter breadth: "))

area = length*breadth
perimeter = 2*(length+breadth)

print("Area: ", area)
print("Perimeter: ", perimeter)
```

OUTPUT:
```
Enter length: 4
Enter breadth: 6
Area:  24
Perimeter:  20
```

QUESTION:
3. Input three decimal numbers and find their sum and average.

CODE:
```
n1 = float(input("Enter 1st decimal number: "))
n2 = float(input("Enter 2nd decimal number: "))
n3 = float(input("Enter 3rd decimal number: "))

sum = n1+n2+n3
```

```
average = sum/3

print("Sum: ", sum)
print("Average: ", average)
```

OUTPUT:
Enter 1st decimal number: 2
Enter 2nd decimal number: 3
Enter 3rd decimal number: 4
Sum: 9.0
Average: 3.0

QUESTION:
4. Input two numbers and swap them.

CODE:
```
a = int(input("Enter 1st number: "))
b = int(input("Enter 2nd number: "))

t = a
a = b
b = t

print("1st number is now: ", a)
print("2nd number is now: ", b)
```

OUTPUT:
Enter 1st number: 4
Enter 2nd number: 6
1st number is now: 6
2nd number is now: 4

QUESTION:
5. Input temperature in Celsius and convert it to Fahrenheit.

CODE:
```
celsius = float(input("Enter temperature in celsius: "))
fahrenheit = (celsius * 9/5) + 32
print('%.2f Celsius is: %0.2f Fahrenheit' %(celsius, fahrenheit))
```
OUTPUT:
Enter temperature in celsius: 100
100.00 Celsius is: 212.00 Fahrenheit

QUESTION:
6. Input a number and find its absolute value.

CODE:
```
n = float(input("Enter any number: "))
abs = 0
if (n < 0):
    abs = (-1) * n

print("Absolute value of %.2f is %.2f" % (n, abs))
```

OUTPUT:
```
Enter any number: -32
Absolute value of -32.00 is 32.00
```

QUESTION:
7. Input a number and check whether it is odd or even and display accordingly.

CODE:
```
n = float(input("Enter any number: "))
if(n % 2 == 0):
    print("Number is even")
else:
    print("Number is odd")
```

OUTPUT:
```
Enter any number: 45
Number is odd
```

QUESTION:
8. Find the largest and smallest among three numbers supplied by user.

CODE:
```
number1 = int(input('Enter 1st number : '))
number2 = int(input('Enter 2nd number : '))
number3 = int(input('Enter 3rd number : '))


def largest(num1, num2, num3):
    if (num1 > num2) and (num1 > num3):
        largest_num = num1
```

```
    elif (num2 > num1) and (num2 > num3):
        largest_num = num2
    else:
        largest_num = num3
    print("The largest of the 3 numbers is : ", largest_num)


def smallest(num1, num2, num3):
    if (num1 < num2) and (num1 < num3):
        smallest_num = num1
    elif (num2 < num1) and (num2 < num3):
        smallest_num = num2
    else:
        smallest_num = num3
    print("The smallest of the 3 numbers is : ", smallest_num)


largest(number1, number2, number3)
smallest(number1, number2, number3)
```

OUTPUT:

```
Enter 1st number : 23
Enter 2nd number : 45
Enter 3rd number : 12
The largest of the 3 numbers is :  45
The smallest of the 3 numbers is :  12
```

QUESTION:

9. Check whether an input year is a leap year or not.

CODE:

```
year = int(input("Enter a year: "))
if (year % 4) == 0:
    if (year % 100) == 0:
        if (year % 400) == 0:
            print("{0} is a leap year".format(year))
        else:
            print("{0} is not a leap year".format(year))
    else:
        print("{0} is a leap year".format(year))
else:
    print("{0} is not a leap year".format(year))
```

OUTPUT:
Enter a year: 2000
2000 is a leap year

QUESTION:
10. Compute the telephone bill for Mr. X as per the call rates given below:
Rental = 250
1st 100 calls @Rs. 0.2
Next 100 calls @ Rs. 0.3
Remaining calls @ Rs. 0.5

CODE:
```
calls = int(input("Enter number of calls: "))
bill = 0

if calls <= 100:
    bill = calls*0.2

elif calls > 100 and calls <= 200:
    bill = 100*0.2+(calls-100)*0.3

else:
    bill = 100*0.2+100*0.3+(calls-200)*0.5

bill += 250

print("Total bill amount is", bill)
```

OUTPUT:
Enter number of calls: 169
Total bill amount is 290.7

QUESTION:
11. Solve a given quadratic equation. [Without imaginary roots].

CODE:
```
import cmath

a = int(input("Enter co-eff with deg 2:"))
b = int(input("Enter co-eff with deg 1:"))
c = int(input("Enter co-eff with deg 0:"))
```

```
# calculate the discriminant
d = (b**2) - (4*a*c)

# find two solutions
s1 = (-b-cmath.sqrt(d))/(2*a)
s2 = (-b+cmath.sqrt(d))/(2*a)

print('The solution are {0} and {1}'.format(s1, s2))
```

OUTPUT:
Enter co-eff with deg 2:1
Enter co-eff with deg 1:-2
Enter co-eff with deg 0:1
The solution are (1+0j) and (1+0j)

QUESTION:
12. Print the following patterns up to n no. of lines:
(a)

```
 *
 * *
 * * *
 * * * *
 * * * * *
```

CODE:

```
rows = int(input("Enter number of rows: "))

for num in range(rows+1):

    for i in range(num):

        print("*", end=" ")  # print number

    # line after each row to display pattern correctly

    print(" ")
```

OUTPUT:
Enter number of rows: 5

*

```
* *
* * *
* * * *
* * * * *
```

QUESTION:
(b)

```
    1
  1 2
 1 2 3
1 2 3 4
1 2 3 4 5
```

CODE:

```
rows = int(input("Enter number of rows: "))

for row in range(1, rows+1):
    num = 1
    for j in range(rows, 0, -1):
        if j > row:
            print(" ", end=" ")
        else:
            print(num, end=" ")
            num += 1
    print("")
```

OUTPUT:

```
Enter number of rows: 5
    1
  1 2
 1 2 3
1 2 3 4
1 2 3 4 5
```

QUESTION:
(c)

```
    *
  * * *
* * * * *
```

CODE:

```
a = int(input("Enter number of rows: "))
n = 2*a-1
for i in range(1, a+1):
    for j in range(1, a-i+1):
        print(" ", end="")
    for j in range(1, 2*i):
        print("*", end="")
    print()
```

OUTPUT:

```
Enter number of rows: 5
    *
   ***
  *****
 *******
*********
```

QUESTION:

(d)

```
     *
    * * *
   * * * * *
  * * * * * * *
   * * * * *
    * * *
     *
```

CODE:

```
a = int(input("Enter number of rows: "))

n = 2*a-1
for i in range(1, a+1):
    for j in range(1, a-i+1):
        print(" ", end="")
    for j in range(1, 2*i):
        print("*", end="")
    print()
for i in range(a-1, 0, -1):
    for j in range(1, a-i+1):
        print(" ", end="")
    for j in range(1, 2*i):
```

```
        print("*", end="")
    print()
```

OUTPUT:

```
Enter number of rows: 5
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

QUESTION:

(e)

```
 * * * * * * *
  * * * * *
   * * *
    *
   * * *
  * * * * *
 * * * * * * *
```

CODE:

```
a = int(input("Enter number of rows: "))
n = 2*a-1
for i in range(a, 0, -1):
    for j in range(1, a-i+1):
        print(" ", end="")
    for j in range(1, 2*i):
        print("*", end="")
    print()
for i in range(2, a+1):
    for j in range(1, a-i+1):
        print(" ", end="")
    for j in range(1, 2*i):
        print("*", end="")
    print()
```

OUTPUT:

```
Enter number of rows: 5
*********
 *******
  *****
   ***
    *
   ***
  *****
 *******
*********
```

QUESTION:
13. Input two numbers and find their hcf and lcm

CODE:

```
a = int(input("Enter 1st number: "))
b = int(input("Enter 2nd number: "))

hcf = 0
lcm = 0

if (a > b):
    x = a
else:
    x = b
for i in range(1, x):
    if (a % i == 0 and b % i == 0):
        hcf = i
print("HCF is: ", hcf)

lcm = (a*b)//hcf
print("lcm of both is: ", lcm)
```

OUTPUT:

```
Enter 1st number: 34
Enter 2nd number: 36
HCF is:  2
lcm of both is:  612
```

QUESTION:
14. Input a number n and find:
(a) Fibonacci series up to n

(b) the nth Fibonacci number
(c) First n Fibonacci numbers

CODE:

```
n = int(input("Enter nth number: "))
num = 0
f = [0, 1]
for i in range(2, n+1):
    num = (f[i-1] + f[i-2])
    if(num <= n):
        f.append(num)
    else:
        break

print("a) Fibonacci series upto n: ", f)

f = [0, 1]
for i in range(2, n+1):
    f.append(f[i-1] + f[i-2])

print("b) nth Fibonacci number: ", f[n-1])
print("c) First n Fibonacci number: ", f)
```

OUTPUT:

```
Enter nth number: 8
a) Fibonacci series upto n:  [0, 1, 1, 2, 3, 5, 8]
b) nth Fibonacci number:  13
c) First n Fibonacci number:  [0, 1, 1, 2, 3, 5, 8, 13, 21]
```

QUESTION:
15. Input a number and find the sum of its digits using while/do-while loop.

CODE:

```
n = int(input("Enter a number: "))

sum = 0
while(n != 0):
    r = n % 10
    n = int(n/10)
    sum += r

print("Sum of digits: ", sum)
```

OUTPUT:
Enter a number: 123456789
Sum of digits: 45

QUESTION:
16. Input a number and reverse it using while/do-while loop.

CODE:
```
n = int(input("Enter a number: "))

rev = 0
while(n != 0):
    r = n % 10
    n = int(n/10)
    rev = rev*10 + r

print("Reverse: ", rev)
```

OUTPUT:
Enter a number: 1234
Reverse: 4321

QUESTION:
17. Input a number and check if it is a prime number or not.

CODE:
```
n = int(input("Enter a number: "))
c = 0
for i in range(1, n+1):
    if(n % i == 0):
        c += 1

if(c > 2):
    print("Composite number")
else:
    print("Prime number")
```

OUTPUT:
Enter a number: 43
Prime number

QUESTION:

18. According to the Goldbach conjecture, every even number greater than two is the sum of two prime numbers.
Input an even number and decompose it into two primes.

CODE:

```
primes = []


def sieveSundaram(n):
    k = (n - 2) // 2
    integers_list = [True] * (k + 1)
    for i in range(1, k + 1):
        j = i
        while i + j + 2 * i * j <= k:
            integers_list[i + j + 2 * i * j] = False
            j += 1
    if n > 2:
        primes.append(2)
    for i in range(1, k + 1):
        if integers_list[i]:
            primes.append(2*i + 1)
    findPrimes(n)


def findPrimes(n):
    if (n <= 2 or n % 2 != 0):
        print("Invalid Input")
        return

    i = 0
    while (primes[i] <= n // 2):
        diff = n - primes[i]
        if diff in primes:
            print(primes[i], "+", diff, "=", n)
            return
        i += 1


n = int(input("Enter a even number to check: "))
sieveSundaram(n)
```

OUTPUT:

Enter a even number to check: 34
3 + 31 = 34

QUESTION:
19. Input a number and check whether it is an automorphic number or not using while/do-while loop.

CODE:

```
n = int(input("Enter a number to check for Automorphic: "))


def isAutomorphic(n):
    sq = n * n
    while (n > 0):
        if (n % 10 != sq % 10):
            return False

        n //= 10
        sq //= 10

    return True


if isAutomorphic(n):
    print("Automorphic")
else:
    print("Not Automorphic")
```

OUTPUT:
Enter a number to check for Automorphic: 25
Automorphic

QUESTION:
20. Input a number and check whether it is an Armstrong number or not using while/do-while loop.

CODE:

```
num = int(input("Enter a number to check for Armstrong: "))
order = len(str(num))

sum = 0
```

```
temp = num
while temp > 0:
    digit = temp % 10
    sum += digit ** order
    temp //= 10

if num == sum:
    print(num, "is an Armstrong number")
else:
    print(num, "is not an Armstrong number")
```

OUTPUT:
Enter a number to check for Armstrong: 153
153 is an Armstrong number

QUESTION:
21. Implement simple arithmetic calculator using user defined functions for each operation (addition, subtraction,
multiplication, division, modulus, exponent). You may use a dictionary to print the menu.

CODE:

```
class Calculator:
    def add(x, y):
        return x + y

    def subtract(x, y):
        return x - y

    def multiply(x, y):
        return x * y

    def divide(x, y):
        return x / y

    def input_num():
        num1 = float(input("Enter first number: "))
        num2 = float(input("Enter second number: "))
        return num1, num2

    while True:
        print("\n\nSelect operation.")
```

```
    print("1. Add")
    print("2. Subtract")
    print("3. Multiply")
    print("4. Divide")
    print("0. Exit")
    choice = input("Enter choice(1/2/3/4/0): ")
    if choice in ('1', '2', '3', '4', '0'):
        if choice == '0':
            break

        elif choice == '1':
            num1, num2 = input_num()
            print(num1, "+", num2, "=", add(num1, num2))

        elif choice == '2':
            num1, num2 = input_num()
            print(num1, "-", num2, "=", subtract(num1, num2))

        elif choice == '3':
            num1, num2 = input_num()
            print(num1, "*", num2, "=", multiply(num1, num2))

        elif choice == '4':
            num1, num2 = input_num()
            if(num2==0):
                print("divide by zero error")
            else:
                print(num1, "/", num2, "=", divide(num1, num2))

    else:
        print("Invalid Input")
```

OUTPUT:

```
Select operation.
1. Add
2. Subtract
3. Multiply
4. Divide
0. Exit
Enter choice(1/2/3/4/0): 3
```

Enter first number: 23
Enter second number: -8
23.0 * -8.0 = -184.0

Select operation.
1. Add
2. Subtract
3. Multiply
4. Divide
0. Exit
Enter choice(1/2/3/4/0): 4
Enter first number: 34
Enter second number: 0
divide by zero error

Select operation.
1. Add
2. Subtract
3. Multiply
4. Divide
0. Exit
Enter choice(1/2/3/4/0): 0

QUESTION:
22. Input a number n and find its factorial using a user defined function long int fact(int)

CODE:
```
n = int(input("Enter a number to find factorial: "))

def fact(n):
    if n == 0:
        return 1
    return n*fact(n-1)

print("Factorial is: ",fact(n))
```

OUTPUT:
Enter a number to find factorial: 6
Factorial is: 720

QUESTION:
23. Input a number and check if it a Krishnamurthy number.

CODE:

```
n = int(input("Enter a number to check Krishnamurthy: "))


def factorial(n):
    fact = 1
    while (n != 0):
        fact = fact * n
        n = n - 1
    return fact


def isKrishnamurthy(n):
    sum = 0
    temp = n
    while (temp != 0):
        rem = temp % 10
        sum = sum + factorial(rem)
        temp = temp // 10

    return (sum == n)


if (isKrishnamurthy(n)):
    print("YES")
else:
    print("NO")
```

OUTPUT:
Enter a number to check Krishnamurthy: 145
YES

QUESTION:
24. Find the sum of first n prime numbers using as user defined function to check for prime. Input the value of n from
the user.

CODE:

```
primes = []


def sieveSundaram(n):
    k = (n - 2) // 2
    integers_list = [True] * (k + 1)
    for i in range(1, k + 1):
        j = i
        while i + j + 2 * i * j <= k:
            integers_list[i + j + 2 * i * j] = False
            j += 1
    if n > 2:
        primes.append(2)
    for i in range(1, k + 1):
        if integers_list[i]:
            primes.append(2*i + 1)


def solve(n):
    sum = 0
    for i in range(n):
        sum += primes[i]
    return sum


sieveSundaram(100000)
num = int(input("Enter a number to find nth prime sum: "))
print("Sum of prime numbers upto %f is %f" % (num, solve(num)))
```

OUTPUT:

```
Enter a number to find nth prime sum: 12
Sum of prime numbers upto 12.000000 is 197.000000
```

QUESTION:

25. Input a limit n and print all prime fibonacci numbers up to n using a user defined function int prime(int) which
returns a 1 if the argument is a prime or else 0.

CODE:

```
def prime(num):
    flag = True
    for i in range(2, num):
```

```
        if num % i == 0:
            flag = False
            break
    return flag

def fibonacci(num):
    return num if num<= 1 else (fibonacci(num-1)+fibonacci(num-2))

def findPrimeFibonacchi(num):
    sequence = [fibonacci(i) for i in range(num)]
    primeList = [i for i in sequence if prime(i) and i>1]
    return sequence, primeList

def main():
    fibSeq, primeSeq = findPrimeFibonacchi(int(input("Enter number of Fibonacchi
terms: ")))
    print("Fibonacchi Sequence:  {}\nPrime Fibonacchi Sequence:  {}".format(fibSeq,
primeSeq))

if __name__ == "__main__":
    main()
```

OUTPUT:

```
Enter number of Fibonacchi terms: 13
Fibonacchi Sequence:  [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144]
Prime Fibonacchi Sequence:  [2, 3, 5, 13, 89]
```

QUESTION:

26. Input a limit n and print all twin prime numbers up to n.

CODE:

```
n = int(input('Enter n : '))
prime = [True for i in range(n + 2)]
p = 2

while (p * p <= n + 1):
    if (prime[p] == True):
        for i in range(p * 2, n + 2, p):
            prime[i] = False
            p += 1

for p in range(2, n-1):
    if prime[p] and prime[p + 2]:
```

```
        print("(", p, ",", (p + 2), ")", end='')
```

OUTPUT:
Enter n : 13
(3 , 5)(5 , 7)(7 , 9)(9 , 11)(11 , 13)

QUESTION:
27. Input the values of two variables n and r and calculate nCr

CODE:
```
def factorial(n):
    if(n == 1 or n == 0):
        return 1
    return n * factorial(n-1)


def calculateCombination(n, r):
    return factorial(n)//(factorial(r) * factorial(n-r))


def main():
    n = int(input("Enter n: "))
    r = int(input("Enter r: "))
    print("nCr = \t ", calculateCombination(n, r))


if __name__ == "__main__":
    main()
```

OUTPUT:
Enter n: 10
Enter r: 3
nCr = 120

QUESTION:
28. Generate a pascal's triangle upto n rows (value of n is to be taken as input from the user).

CODE:
```
def pascal(n):
    for line in range(1, n+1):
        co_eff = 1
```

```
        for i_loop in range(1, line+1):
            print(co_eff, end=" ")
            co_eff = int(co_eff*(line-i_loop)/i_loop)

        print("")


row_num = int(
    input("Enter the number of rows of Pascal's triangel you want to generate : "))
pascal(row_num)
```

OUTPUT:

```
Enter the number of rows of Pascal's triangel you want to generate : 6
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
```

QUESTION:

29. Input a number n and print its prime factors using a user defined function int prime(int) which returns a 1 if the
argument is a prime or else 0.

CODE:

```
def Factors(num):
    factorList = [i for i in range(1, num+1) if num % i == 0]
    return factorList


def prime(num):
    flag = True
    for i in range(2, num):
        if num % i == 0:
            flag = False
            break
    return flag


def primeFact(num):
```

```
    return [i for i in Factors(num) if prime(i) and i > 1]


def main():
    print("Prime Factors are: {}".format(
        primeFact(int(input("Enter a number: ")))))


if __name__ == "__main__":
    main()
```

OUTPUT:
Enter a number: 36
Prime Factors are: [2, 3]

QUESTION:
30. Calculate the below series. Use a user defined function fact(n) which will return the factorial of n.
x/1! – x^3/3! + x^5/5! – x^7/7! + …….. (up to n terms)

CODE:

```
def fact(num):
    if num <= 0:
        return 1

    f = 1
    for i in range(1, num):
        f = f*i
    return f


x = int(input('Enter x : '))
n = int(input('Enter n : '))

series_sum = 0.0
i = 1
while i <= n:
    series_sum = series_sum + (x**i)/fact(i)
    i += 2

print("Series sum = ", series_sum)
```

OUTPUT:
Enter x : 10
Enter n : 100
Series sum = 110132.32920103321

QUESTION:
31. Print the sum of natural numbers up to n using recursion.

CODE:
```
def sumUptoN(n):
    if(n == 1):
        return 1
    return n + sumUptoN(n-1)


def main():
    limit = int(
        input("Enter n, limit till which sum of natural numbers is to be found: "))
    print("Sum of natural numbers upto " + str(limit) + ": ", sumUptoN(limit))


if __name__ == "__main__":
    main()
```

OUTPUT:
Enter n, limit till which sum of natural numbers is to be found: 12
Sum of natural numbers upto 12: 78

QUESTION:
32. Find the factorial of a number using recursion.

CODE:
```
def factorial(num):
    if (num == 0):
        return 1
    elif (num == 1):
        return 1
    else:
        return num*factorial(num-1)


n = int(input("Enter the number whose factorial you want to find : "))
```

```
print("The factorial of the number is {}".format(factorial(n)))
```

OUTPUT:
Enter the number whose factorial you want to find : 8
The factorial of the number is 40320

QUESTION:
33. Find the nth Fibonacci number using recursion. The value of n should be taken as input.

CODE:
```
def fibonacci(num):
    return num if num <= 1 else (fibonacci(num-1)+fibonacci(num-2))


def nthFib(num):
    sequence = [fibonacci(i) for i in range(num)]
    return sequence[len(sequence)-1]


def main():
    print("Nth Fibonacchi number: {}".format(
        nthFib(int(input("Enter a number: ")))))


if __name__ == "__main__":
    main()
```

OUTPUT:
Enter a number: 12
Nth Fibonacchi number: 89

QUESTION:
34. Find the nth Lucas number using recursion. The value of n should be taken as input.

CODE:
```
def lucas(n):

    # Base cases
    if (n == 0):
```

```
    return 2
  if (n == 1):
    return 1

  # recurrence relation
  return lucas(n - 1) + lucas(n - 2)


# Driver code
n = int(input('Enter n : '))
print(lucas(n))
```

OUTPUT:
Enter n : 5
11

QUESTION:
35. Input two numbers a and b and calculate a^b using recursion.

CODE:

```
def power(a, b):
  if(b == 1):
    return a
  return a * power(a, b-1)


def main():
  base = int(input("Enter base: "))
  exponent = int(input("Enter exponent: "))
  print(str(base) + "^" + str(exponent) + " = ", power(base, exponent))


if __name__ == "__main__":
  main()
```

OUTPUT:
Enter base: 4
Enter exponent: 3
4^3 = 64

QUESTION:
36. Input two numbers and find their GCD using recursion.

CODE:

```
def gcd(first, second):
    if (first == 0):
        return second
    if (second == 0):
        return first
    if (first == second):
        return first
    if (first > second):
        return gcd(first-second, second)
    return gcd(first, second-first)


first = int(input("Enter the first number : "))
second = int(input("Enter the second number : "))

print("The Greatest Common Divisor of the two numbers is {}".format(gcd(first,
second)))
```

OUTPUT:

```
Enter the first number : 24
Enter the second number : 36
The Greatest Common Divisor of the two numbers is 12
```

QUESTION:

37. Solve the Tower of Hanoi problem for 3 discs using recursion.

CODE:

```
def towerOfHanoi(disk, source, temp, destination):
    if disk == 1:
        print('Move disk 1 from rod {} to rod {}'.format(source, destination))
        return
    towerOfHanoi(disk-1, source, destination, temp)
    print('Move disk {} from rod {} to rod {}'.format(disk, source, destination))
    towerOfHanoi(disk-1, temp, source, destination)


def main():
    print('A: Source\nB: Temporary\nC:Destination')
    towerOfHanoi(int(input("Enter number of disks: ")), 'A', 'B', 'C')
```

```
if __name__ == "__main__":
    main()
```

OUTPUT:
A: Source
B: Temporary
C:Destination
Enter number of disks: 3
Move disk 1 from rod A to rod C
Move disk 2 from rod A to rod B
Move disk 1 from rod C to rod B
Move disk 3 from rod A to rod C
Move disk 1 from rod B to rod A
Move disk 2 from rod B to rod C
Move disk 1 from rod A to rod C

QUESTION:
38. Solve the Ackermann function for two non-negative integers m and n.

CODE:
```
def ack(M, N):
    if M == 0:
        return N + 1
    elif N == 0:
        return ack(M - 1, 1)
    else:
        return ack(M - 1, ack(M, N - 1))

m=int(input("1st no.  "))
n=int(input("2nd no.  "))
print(ack(m,n))
```

OUTPUT:
1st no. 3
2nd no. 4
125

QUESTION:

39. Create a tuple with different data types, print the tuple and then add one more item into the tuple and print it again.

CODE:

```
def main():
    i1 = int(input("Enter a number: "))
    i2 = float(input("Enter a decimal number: "))
    i3 = input("Enter a string: ")
    original_tuple = (i1, i2, i3)
    print("Original tuple: \t ", original_tuple)
    intermediate_list = list(original_tuple)
    print("Intermediate list: \t ", intermediate_list)
    intermediate_list.append("App")
    new_tuple = tuple(intermediate_list)
    print("New tuple: \t\t ", new_tuple)


if __name__ == "__main__":
    main()
```

OUTPUT:

```
Enter a number: 34
Enter a decimal number: 3.4
Enter a string: threefour
Original tuple:          (34, 3.4, 'threefour')
Intermediate list:       [34, 3.4, 'threefour']
New tuple:               (34, 3.4, 'threefour', 'App')
```

QUESTION:

40. Create a tuple and find the repeated items of the tuple.

CODE:

```
import collections


def count(tuple_list):
    flag = False
    value = collections.Counter(tuple_list)
    uniq_list = list(set(tuple_list))

    for i_loop in uniq_list:
        if (value[i_loop] >= 2):
```

```
            flag = True
            print(i_loop, " is repeated ", value[i_loop], " times")

    if (flag == False):
        print("Repeated items do not exist")


tuple_list = [('a', 'b'), ('c', 'd'), ('e', 'f'), ('a', 'b'), ('e', 'f')]
print(tuple_list)
count(tuple_list)
```

OUTPUT:

```
[('a', 'b'), ('c', 'd'), ('e', 'f'), ('a', 'b'), ('e', 'f')]
('a', 'b')  is repeated  2  times
('e', 'f')  is repeated  2  times
```

QUESTION:

41. Reverse a given tuple.

CODE:

```
def reverseTuple(inputTuple):
    return inputTuple[::-1]


def main():
    inputTuple = tuple(i.strip() for i in input(
        "Enter a tuple spaced with ',': ").split(','))
    print("Input Tuple: {}".format(inputTuple))
    print("Reversed Tuple: {}".format(reverseTuple(inputTuple)))


if __name__ == '__main__':
    main()
```

OUTPUT:

```
Enter a tuple spaced with ',': a,v,g,hh,j,4,55
Input Tuple: ('a', 'v', 'g', 'hh', 'j', '4', '55')
Reversed Tuple: ('55', '4', 'j', 'hh', 'g', 'v', 'a')
```

QUESTION:

42. Find the sum of digits of a number using an user defined function sumofdigits(x) which returns a tuple containing the digits of the number.

CODE:

```
def sumofdigits(x):
    sum = 0
    while(x > 0):
        sum += int(x % 10)
        x = int(x/10)
    return [int(d) for d in str(sum)]


n = int(input("Enter a number: "))

print(sumofdigits(n))
```

OUTPUT:

```
Enter a number: 456
[1, 5]
```

QUESTION:

43. Input a number and check if it a pefect number or not using an user defined function factors(x) which returns a tuple containing the factors of the number.

CODE:

```
def factors(x):
    factor_list = []
    for i in range(1, x+1):
        if(x % i == 0):
            factor_list.append(i)
    return tuple(factor_list)


def main():
    number = int(input("Enter number to check if it is a perfect number: "))
    factor_tuple = factors(number)
    factor_sum = 0
    for each in range(len(factor_tuple)-1):
        factor_sum += factor_tuple[each]
    if(number == factor_sum):
        print("It is a perfect number")
    else:
        print("It is not a perfect number")
```

```
if __name__ == "__main__":
    main()
```

OUTPUT:
Enter number to check if it is a perfect number: 496
It is a perfect number

QUESTION:
44. Convert a given list into a tuple.

CODE:
```
l = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']
print(l, "\n", tuple(l))
```

OUTPUT:
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']
 ('a', 'b', 'c', 'd', 'e', 'f', 'g', 'h')

QUESTION:
45. Create a list of n tuples and sort them according to the second element of each tuple.

CODE:
```
def sortTupleList(tupleList):
    try:
        return sorted(tupleList, key=lambda tup: tup[1])
    except IndexError:
        print("A tuple may not have a second element")


def main():
    tupleList = list(tuple(j.strip() for j in input("Enter tuple number {} with a ',' separating
each element: ".format(
        i+1)).split(',')) for i in range(int(input('Enter number of tuples: '))))
    print("Your tuple: {}".format(tupleList))
    print("Sorted by second element: {}".format(sortTupleList(tupleList)))


if __name__ == "__main__":
    main()
```

OUTPUT:
Enter number of tuples: 3
Enter tuple number 1 with a ',' separating each element: 1,2,3
Enter tuple number 2 with a ',' separating each element: 6,5,4
Enter tuple number 3 with a ',' separating each element: a,b,d
Your tuple: [('1', '2', '3'), ('6', '5', '4'), ('a', 'b', 'd')]
Sorted by second element: [('1', '2', '3'), ('6', '5', '4'), ('a', 'b', 'd')]

QUESTION:
46. Create a list containing n tuples and replace last value of all tuples in the list.
Sample list: [(10,20,40), (40,50,60), (70,80,90)]
Expected Output: [(10, 20, 100), (40, 50, 100), (70, 80, 100)]

CODE:
```
sample_tuple = [(10, 20, 30), (1, 2, 3), (2323, 32432, 234189)]
final_tuple = [t[:-1] + (100,) for t in sample_tuple]
print(final_tuple)
```

OUTPUT:
[(10, 20, 100), (1, 2, 100), (2323, 32432, 100)]

QUESTION:
47. Create a list containing n tuples and remove all empty tuple(s) from the list.
Sample data: [(),(), ('',), ('a', 'b'), ('a', 'b', 'c'), ('d')]
Expected output: [('',), ('a', 'b'), ('a', 'b', 'c'), 'd']

CODE:
```
def removeEmpty(full_list):
    new_list = []
    for each in full_list:
        if(len(each) == 0):
            continue
        new_list.append(each)
    return new_list


def main():
    original_list = [(), (1, "ss"), (), (), ('',), ("abcd", 1)]
    print("Original list: ", original_list)
    print("New list: ", removeEmpty(original_list))
```

```
if __name__ == "__main__":
    main()
```

OUTPUT:
Original list: [(), (1, 'ss'), (), (), ('',), ('abcd', 1)]
New list: [(1, 'ss'), ('',), ('abcd', 1)]

QUESTION:
48. Input a list and split the odd and even numbers into two separate lists.

CODE:

```
def splitlist(list1):
    evenlist = []
    oddlist = []

    for i_loop in list1:
        if (i_loop % 2 == 0):
            evenlist.append(i_loop)
        else:
            oddlist.append(i_loop)

    print("Elements in the even list are : ")
    print(evenlist)
    print("Elements in the odd list are : ")
    print(oddlist)


list1 = []

size = int(input("Enter the number of elements in the Original List ::"))

print("Enter the elements of the Original List ::")
for i_loop in range(int(size)):
    ele = int(input(""))
    list1.append(ele)

splitlist(list1)
```

OUTPUT:
Enter the number of elements in the Original List ::6
Enter the elements of the Original List ::
2

4
3
1
0
-7
Elements in the even list are :
[2, 4, 0]
Elements in the odd list are :
[3, 1, -7]

QUESTION:
49. Enter the roll, name and marks of n students print their records. Finally find the student name with the highest
marks.

CODE:
```
class studentDetails:
    def __init__(self, name, roll, marks):
        self.name = name
        self.roll = roll
        self.marks = marks

    def getMarks(self):
        return self.marks

    def getRoll(self):
        return self.roll

    def getName(self):
        return self.name

    def show(self):
        return 'Name: '+self.name+'  Roll: '+str(self.getRoll())+'  Marks:
'+str(self.getMarks())


def getHighestMarks(record):
    marks = max([i.getMarks() for i in record])
    flag = True
    for j in record:
        if marks == j.getMarks():
            print("{} got highest marks.".format(j.getName()))
```

```
            flag = False
    if flag == True:
        print("Invalid")


def main():
    record = []
    confirmation = 'y'
    while confirmation == 'y':
        name = input("Enter Student Name: ")
        roll = int(input("Enter Student Roll: "))
        marks = int(input("Enter Student Marks: "))
        record.append(studentDetails(name, roll, marks))
        confirmation = input("Want to insert another record?(y/n)")
    for i in record:
        print(i.show())

    getHighestMarks(record)


if __name__ == "__main__":
    main()
```

OUTPUT:

```
Enter Student Name: Ram Pal
Enter Student Roll: 5
Enter Student Marks: 89
Want to insert another record?(y/n)y
Enter Student Name: Sam Pal
Enter Student Roll: 6
Enter Student Marks: 98
Want to insert another record?(y/n)n
Name: Ram Pal  Roll: 5  Marks: 89
Name: Sam Pal  Roll: 6  Marks: 98
Sam Pal got highest marks.
```

QUESTION:

50. Input two strings and concatenate them.

CODE:

```
str1 = input('Enter string 1: ')
str2 = input('Enter string 2: ')
```

```
str3 = str1+str2
print("Concatenated string=\n", str3,sep='')
```

OUTPUT:
Enter string 1: qwerty
Enter string 2: keyboard
Concatenated string=
qwerty keyboard

QUESTION:
51. Input a string and reverse it.

CODE:
```
s=input("enter a string ")
res=''
for i in range(len(s)-1,-1,-1):
    res+=str(s[i])
print(res)
```
OUTPUT:
enter a string qwerty
ytrewq

QUESTION:
52. Enter a sentence and find number of vowels, consonants, spaces and special characters.

CODE:
```
str = input("Enter the string : ")
vowels = 0
digits = 0
consonants = 0
spaces = 0
symbols = 0
str = str.lower()
for i in range(0, len(str)):
    if(str[i] == 'a' or str[i] == 'e' or str[i] == 'i' or str[i] == 'o' or str[i] == 'u'):
        vowels = vowels + 1
    elif((str[i] >= 'a' and str[i] <= 'z')):
        consonants = consonants + 1
    elif(str[i] >= '0' and str[i] <= '9'):
        digits = digits + 1
```

```
    elif (str[i] == ' '):
        spaces = spaces + 1
    else:
        symbols = symbols + 1

print("Vowels: ", vowels)
print("Consonents: ", consonants)
print("Digits: ", digits)
print("White Spaces: ", spaces)
print("Symbols: ", symbols)
```

OUTPUT:
```
Enter the string : qweqwerwer erg srth rg
Vowels:  4
Consonents:  15
Digits:  0
White Spaces:  4
Symbols:  0
```

QUESTION:
54. Input a word and print it vertically.

CODE:
```
word = input("enter a word:   ")
for i in range(len(word)):
    print(word[i])
```

OUTPUT:
```
enter a word:   qwerty
q
w
e
r
t
y
```

QUESTION:
55. Input a string and check if it a palindrome or not.

CODE:
```
s=input("enter a string ")
n=len(s)
```

```
flg=0
for i in range(n):
    if(s[i]!=s[n-i-1]):
        print("not Palindrome")
        flg=1
        break;
if flg==0:
    print("Plaindrome")
```

OUTPUT:
enter a string assa
Plaindrome

QUESTION:
56. Input a string and count the number of words in it.

CODE:
```
str = input("Enter the sentence : ")
print("The original string is : ", str)
res = len(str.split())
print("The number of words in string are : ", res)
```

OUTPUT:
Enter the sentence : qwe qwe qwe qwe
The original string is : qwe qwe qwe qwe
The number of words in string are : 4

QUESTION:
57. Input a string and reverse it using recursion.

CODE:
```
def revstr(s):
    if(len(s)==1):
        return s
    return revstr(s[1:])+s[0]

st=input("enter a string ")
print(revstr(st))
```
OUTPUT:
enter a string qwe wer ert rty
ytr tre rew ewq

QUESTION:
58. Input a sentence and find the number of words starting with ‘S’.

CODE:

```
str1 = input("Input a sentence and find the number of words starting with S:   ")

words = str1.split()
num = 0
for i in range(len(words)):
    if(words[i][0] == 's' or words[i][0] == 'S'):
        num += 1

print("The number of words starting with'S' is: ", num)
```

OUTPUT:

```
Input a sentence and find the number of words starting with S:   Input a sentence and
find the number of words starting with S:
The number of words starting with'S' is:  3
```

QUESTION:
59. Input a string and count number of palindromic words present in it using an user defined function palin(word)
which returns 1 if the argument is palindrome.

CODE:

```
def palin(word):
    n=len(word)
    flg=1
    for i in range(n):
        if(word[i]!=word[n-i-1]):
            flg=0
            break;
    return flg

sen=input("enter a string ")
sen=sen.split()
print('Palindrome words are:- ')
c=0
for i in sen:
    if palin(i)==1:
        c=c+1
print(c)
```

OUTPUT:
enter a string qwe assa erre qwee
Palindrome words are:-
2

QUESTION:
60. Input a name and find its initial (e.g., Subhash Chandra Bose should be printed as S. C. B).

CODE:
```
name = input("Enter your name:")
words = name.split()
short = ""
for i in range(len(words)):
    short = short+words[i][0].upper()
    short = short+". "

print(short)
```

OUTPUT:
Enter your name:Ram Chand Pal
R. C. P.

QUESTION:
61. Input a name and find its initial (e.g., Subhash Chandra Bose should be printed as S. C. Bose).

CODE:
```
sen=input("enter a name ")
sen=sen.split()
res=''
for i in range(len(sen)):
    if(i<len(sen)-1):
        res+=sen[i][0]+'. '
    else:
        res+=sen[i]
print(res)
```

OUTPUT:
enter a name Ram Chand Pal
R. C. Pal

QUESTION:
62. Input a string and delete all consecutive occurrences of characters.

CODE:

```
def removeDuplicates(S):

    n = len(S)

    if (n < 2):
        return
    j = 0

    for i in range(n):
        if (S[j] != S[i]):
            j += 1
            S[j] = S[i]

    j += 1
    S = S[:j]
    return S


S1 = input("Enter a string: ")
S1 = list(S1.rstrip())
S1 = removeDuplicates(S1)
print(*S1, sep="")
```

OUTPUT:

```
Enter a string: aaa dddd ddaaadddaaa
a d dada
```

QUESTION:
63. Input a string and a pattern and count the occurrences of the pattern in the string.

CODE:

```
s=input("enter a string ")
p=input("enter a pattern ")

print('no of occurance of "'+p+'" in "'+s+'" is',s.count(p))
```

OUTPUT:

```
enter a string qwerty rty rerty tyu
enter a pattern rty
no of occurance of "rty" in "qwerty rty rerty tyu" is 3
```

QUESTION:
64. Input a string and a pattern and delete all occurrences of the pattern from the string.

CODE:

```
def removeCharRecursive(str, X):

    if (len(str) == 0):
        return ""

    return str.replace(X, "")


str=input("enter a string:  ")

X = input("enter pattern:  ")

str = removeCharRecursive(str, X)

print(str)
```

OUTPUT:

```
enter a string:  qwerty erty rty tyur
enter pattern:  rty
qwe e  tyur
```

QUESTION:
65. Input a string and two patterns pattern1 and pattern2. Find all occurrences of pattern1 and replace them by
pattern2.

CODE:

```
s=input("enter a string ")
p=input("enter pattern to be replaced ")
r=input("enter pattren to replace with ")
res=s.replace(p,r)
print(res)
```

OUTPUT:
enter a string qwerty erty rty tyuir rety
enter pattern to be replaced rty
enter pattren to replace with asd
qweasd easd asd tyuir rety

QUESTION:
67. Find the frequency of occurrence of each character in a given string.

CODE:

```
s=input("enter a string ")
di={}
for i in s:
    if i in di.keys():
        di[i]+=1
    else:
        di[i]=1
res=str(di).replace(',','\n')
print('frequencies of each character:-'+' '+res[1:len(res)-1])
```

OUTPUT:
enter a string asd werty qwertysd asdrte rw rtwdfd eyhe
frequencies of each character:- 'a': 2
's': 3
'd': 5
' ': 6
'w': 4
'e': 5
'r': 5
't': 4
'y': 3
'q': 1
'f': 1
'h': 1

QUESTION:
68. Create a class called Rectangle having two attributes – length and breadth and find the area and perimeter of the
rectangle using two methods – showarea() and showperimeter()

CODE:

```
class Rectangle:
```

```
    def __init__(self, l, b):
        self.length = l
        self.breadth = b

    def showarea(self):
        return self.length*self.breadth

    def showperimeter(self):
        return 2*(self.length+self.breadth)


l = int(input("Enter Length: "))
b = int(input("Enter breadth: "))
r = Rectangle(l, b)
print("Area: ", r.showarea())
print("Perimeter: ", r.showperimeter())
```

OUTPUT:
Enter Length: 4
Enter breadth: 6
Area: 24
Perimeter: 20

QUESTION:
69. Create a class called Complex having two attributes – real and imag. Create two instances of the class and perform
addition of two complex numbers. The class should use the following methods:
show() – that displays the complex numbers in proper format
add() – that adds the two complex instances
sub() – that subtracts one complex instance from another
displaymodulus() – that displays the modulus of the resultant complex number after addition or subtraction

CODE:

```
import math
class Complex:
    def __init__(self,real,imag):
        self.real=real
        self.imag=imag
    def add(self,num2):
        return Complex(self.real+num2.real,self.imag+num2.imag)
    def  sub(self,num2):
```

```
        return Complex(self.real-num2.real,self.imag-num2.imag)
    def displaymodulus(self):
        print(math.sqrt(self.real**2+self.imag**2))
    def show(self):
        print(self.real,end='')
        if self.imag<0:
            print(' - i',self.imag,sep='')
        else:
            print(' + i',self.imag,sep='')


a=Complex(3,2)
b=Complex(2,1)
c=a.add(b)
print("the complex numbers are: \n")
a.show()
b.show()
print("addition:")
c.show()
```

OUTPUT:

```
the complex numbers are:

3 + i2
2 + i1
addition:
5 + i3
```

QUESTION:

70. Design a class to represent a bank account. Include the following members:
Data members - Name of the depositor, Account number, Type of account, Account balance
Methods – To deposit an amount, To withdraw an amount after checking balance, to display the
name and balance.
Incorporate a constructor to provide the initial values.

CODE:

```
class bank:
    def __init__(self, name, ac_no, type, bal):
        self.name = name
        self.ac_no = ac_no
```

```
        self.type = type
        self.balance = bal

    def deposite(self, value):
        self.balance += value

    def withdraw(self, value):
        if(value > self.balance):
            print("Insufficient Balance!!!")
        else:
            self.balance -= value

    def show(self):
        print("Name:", self.name)
        print("Balance: ", self.balance)

a=input("enter name: ")
b=input("enter ac no.  ")
c=input("enter type: ")
d=int(input("enter bal.  "))
c1 = bank(a,b,c,d)
c1.show()
e=int(input("enter deposit:  "))
c1.deposite(e)
c1.show()
f=int(input("enter withdraw: "))
c1.withdraw(f)
c1.show()
```

OUTPUT:

```
enter name: Ram Pal
enter ac no.  30ART456
enter type: Basic
enter bal.  123234345
Name: Ram Pal
Balance:  123234345
enter deposit:  45
Name: Ram Pal
Balance:  123234390
enter withdraw: 46
Name: Ram Pal
Balance:  123234344
```