Task:

1.WAP to product of 2 matrices...

```
#Matrix Multiplication
def matrix_multiply(A, B):
    rows_A, cols_A = len(A), len(A[0])
    rows_B, cols_B = len(B), len(B[0])

    if cols_A != rows_B:
        return "Matrix multiplication not possible!"

    result = [[0] * cols_B for _ in range(rows_A)]

    for i in range(rows_A):
        for j in range(cols_B):
            for k in range(cols_A):
                result[i][j] += A[i][k] * B[k][j]

    return result

A = [[1, 2], [3, -1]]
B = [[1, -2, 3], [2, 3, -1]]
res = matrix_multiply(A, B)
for row in res:
    print(row)
print()
print()
```

2. WAP to get nearest prime of given number...

```
def is_prime(n):
    if n<2:
        return False
    for i in range(2,int(n**0.5)+1):
        if n%i == 0:
            return False
    return True
def nearest_prime(num):
    left = num-1
    right = num+1

    while True:
        if left > 1 and is_prime(left):
            return left
        if is_prime(right):
            return right
        left -=1
        right+=1
n = int(input("Enter a number: "))
print(nearest_prime(n))
```

3. WAP to find repeated values in a list...

```
l = [1,2,4,3,5,2,3,6,8,7,9]
ll = []
for i in range(len(l)):
    if l.count(l[i])>1 and l[i] not in ll:
        print(l[i],end=" ")
        ll.append(l[i])
```

4. WAP to print hallow diamond...

```
n = 5
for i in range(1,n):
    s = " "*(n-i)
    if i == 1:
        print(s+"* "*i)
    else:
        print(s+"* "+  "*(i-2)+"*")

for i in range(n,0,-1):
    s = " "*(n-i)
    if i == 1:
        print(s+"* "*i)
    else:
        print(s+"* "+  "*(i-2)+"*")
```

5. WAP for replace built in function of string...

```
def replacee(s,old,new):
    sl=len(s)
    ol = len(old)
    i = 0
    res = ""
    while i<sl:
        if s[i]==old[0]:
            if s[i:i+ol]==old:
                res+=new
                i+=ol
            else:
                res+=s[i]
                i+=1
```

```
        else:
            res+=s[i]
            i+=1
    return res


s = "sravan"
old="a"
new ="b"
print(replacee(s,old,new))
```