Merkle damgard transform code explanation:

- First we will generate some random message which is having length 2^6
- Next , an Initialization vector IV will be generated of length N=16 (block size)
- We will pass this message and IV to CRHF() function
- In this function , message will be divided into block of length N and padding will be done if necessary
- At the end , we will add message length which is encoded into binary
- Now we will apply fixed length hash function on each blocks with first input as message block and second input as previous hash value Z(In first iteration it is IV)
- At the end we will return Z(hash value generated by last block)

```
def CRHF(msg, IV):
    L = len(msg)
    rem = L % N
    if rem != 0:
        temp = msg[-1:rem:].zfill(N)
        msg = msg[:-1*rem]
        msg = msg+temp
    bin_len=bin(math.ceil(math.log2(L))).replace('0b','').zfill(N)

    L = len(msg)
    hash_list = list()
    Z = IV  # inital second input is Z which is IV
    B = L//N  # each block length is N=16 and output length will be 16
    msg=msg+bin_len
    L=len(msg)
    for i in range(0, L, N):
        x = msg[i:i+N]  # taking msg part
        Z = FLHF(x, Z)  # passing msg and Z
    return Z
```