HMAC_code explanation:

- We will generate random message of length 2^6 and an Initialization vector IV ,key K of size N=16
- Here we have some extra variables ipad=0x36,opad=0x5c which are encoded into binary.
- HMAC() function takes message,IV,key,ipad,opad as inputs.
- We will repeat opad and ipad until their length becomes key length (16)
- First we perform hs(k xor ipad,IV) ,lets call it as z
- Now we perform a merkle-damgard transform on a message using the Initialization vector as Z0 and let's assume the output of this process is z.
- Now we perform hs(k xor opad,IV) and let's call it as Zn
- Finally to get HMAC we will apply hs(Z0,Zn)

```
def HMAC(msg,IV,key,ipad,opad):
    #key is also length of N =16
    #IV also length of N=16
    ipad_len=len(ipad)
    opad_len=len(opad)
    while(ipad_len!=N):
        ipad=ipad+ipad
        opad=opad+opad
        ipad_len=2*ipad_len

    kxoripad=bin(int(key,2)^int(ipad,2)).replace('0b','').zfill(N)
    kxoropad=bin(int(key,2)^int(opad,2)).replace('0b','').zfill(N)
    Z0=FLHF(kxoripad,IV)
    Z=CRHF(msg,Z0)
    Zn=FLHF(kxoropad,IV)
    HMAC_TAG=FLHF(Z,Zn)
    return HMAC_TAG
```