Matrice DCT

```
A= Matrix([[27,45,18],[36,27,36],[36,54,27]]); print A
```

```
[27 45 18]
[36 27 36]
[36 54 27]
```

```
def encodingDCT(A,n):
    nlist = []
    for i in range(0,n):
        for j in range(0,n):
            b=0
            for x in range(0,n):
                for y in range(0,n):
                    b=b+A[x]
[y]*cos(((2*x+1)*i*pi)/(2*n))*cos(((2*y+1)*j*pi)/(2*n))/n
            nlist.append(b)
            #print b,
        #print
    return nlist
```

```
encodingDCT(A,3)
```

```
[102, 3*sqrt(3), -12, -9/2*sqrt(3), 0, 0, 3/2, 3/2*sqrt(3), -21/2]
```

```
def encodingDCT_floor(A,n):
    nlist = []
    for i in range(0,n):
        for j in range(0,n):
            b=0
            for x in range(0,n):
                for y in range(0,n):
                    b=b+A[x]
[y]*cos(((2*x+1)*i*pi)/(2*n))*cos(((2*y+1)*j*pi)/(2*n))/n
            nlist.append(b)
            print floor(b),
        #print
    return
```

```
encodingDCT_floor(A,3)
```

```
102 5 -12 -8 0 0 1 2 -11
```

```
def quantifTable(n):
    nlist = []
    for x in range(0,n):
        for y in range(0,n):
            if x>=y:
```

```
                z=x
            else:
                z=y
            a=4-z
            #print a,
        #print
            nlist.append(a)
    return nlist
```

```
quantifTable(3)
```

```
[4, 3, 2, 3, 3, 2, 2, 2, 2]
```

```
def quantification(A,B,n):
    k=0
    for x in range(0,n):
        #for y in range(0,n):
        k=k+1
        z=floor(A[x]/B[x])
        print z,
        if (mod(k,3)==0):
            print
```

```
quantification(encodingDCT(A,3),quantifTable(3),9)
```

```
25 1 -6
-3 0 0
0 1 -6
```

```

```