# Iniciació a la Criptografia


### Artur Travesa


**(versió 2024-07)**

# Apendix 0. Manual de les funcions

## A0.0. Introducció

En aquest apèndix es recullen les funcions definides en els capítols del text, així com també una breu guia del seu funcionament.

Comencem amb les funcions més bàsiques.

### A0.0.0. La funció Codis(text)

Transforma una tira de caràcters en la corresponent llista de codis.

In [ ]:
```
def Codis(text):
    return [ord(i) for i in text]
```

### A0.0.1. La funció Caracters(codis)

Transforma una llista de codis en la tira de caràcters corresponent. De fet, primerament redueix mòdul $m = 1114112$ el valor del codi a fi que sigui un codi de *SageMath*.

In [ ]:
```
def Caracters(codis):
    m=1114112
    cars=""
    for i in range(len(codis)):
        cars=cars+chr(codis[i]%m)
    return cars
```

## A0.1. Criptosistemes de Cèsar, i afins

### A0.1.0. El criptosistema de Cèsar

La funció Cesar(mis,clau) s'aplica a un missatge (tira de caràcters) i una clau (nombre enter) i proporciona el missatge xifrat amb el criptosistema de Cèsar com a tira de caràcters.

In [ ]:
```
def Cesar(mis,clau):
    m=1114112
    xif=""
    for i in range(len(mis)):
        xif=xif+chr((ord(mis[i])+clau) %m)
    return xif
```

### A0.1.1. Versions millorades

Proporcionem tres funcions millorades del criptosistema de Cèsar.

CesarX(mis,clau): permet l'entrada del missatge en format de tira de caràcters o bé en format de llista de codis, i el retorna en el mateix format que l'entrada.

CesarC(mis,clau): permet l'entrada del missatge en format de tira de caràcters o bé en format de llista de codis, i el retorna només com a llista de codis.

CesarD(mis,clau): permet l'entrada del missatge en format de tira de caràcters o bé en format de llista de codis, i el retorna només com a tira de caràcters.

In [ ]:
```
 def CesarX(mis,clau):
    m=1114112
    if type(mis)==str:
        b=1
        lta=Codis(mis)
    else:
        lta=mis
        b=0
    xif=[(lta[i]+clau)%m for i in range(len(lta))]
    if b==1:
        xif=Caracters(xif)
    return xif

```

In [ ]:
```
def CesarC(mis,clau):
    m=1114112
    if type(mis)==str:
        lta=Codis(mis)
    else:
        lta=mis
    xif=[(lta[i]+clau)%m for i in range(len(lta))]
    return xif

```

In [ ]:
```
def CesarD(mis,clau):
    m=1114112
    if type(mis)==str:
        lta=Codis(mis)
    else:
        lta=mis
    xif=Caracters([(lta[i]+clau)%m for i in range(len(lta))])
    return xif

```

### A0.1.2. Criptosistemes afins

La funció AfiX(mis,clau) s'aplica a un missatge en format de tira de caràcters o bé en format de llista de codis, i a una clau (llista de dos codis, mòdul $m = 1114112$, el primer invertible mòdul $m$), i el retorna xifrat amb el criptosistema afí i aquesta clau, en el mateix format que l'entrada.

Les versions funció AfiXC(mis,clau) i funció AfiXD(mis,clau) permeten el missatge d'entrada en format de tira de caràcters o bé en format de llista de codis, i el retornen xifrat, exclusivament, en format de llista de codis, o de tira de caràcters, respectivament.

La funció AfiDX(mis,clau) s'aplica a un missatge xifrat amb una clau afí (llista de dos codis, mòdul $m = 1114112$, el primer invertible mòdul $m$), en format de tira de caràcters o bé en format de llista de codis, i el retorna desxifrat, en el mateix format que l'entrada.

Anàlogament, les funcions AfiDXC(mis,clau) i AfiDXD(mis,clau) per al desxifratge dels missatges.

In [ ]:
```
def AfiX(mis,clau):
    if type(mis)==str:
        lta=Codis(mis)
        b=1
    else:
        lta=mis
        b=0
    m=1114112
    c1=clau[0]
    c2=clau[1]
    xif=[(c1*lta[i]+c2)%m for i in range(len(lta))]
    if b==1:
        xif=Caracters(xif)
    return xif

```

In [ ]:
```
def AfiXC(mis,clau):
    if type(mis)==str:
        lta=Codis(mis)
    else:
        lta=mis
    m=1114112
    c1=clau[0]
    c2=clau[1]
    xif=[(c1*lta[i]+c2)%m for i in range(len(lta))]
    return xif

```

In [ ]:
```
def AfiXD(mis,clau):
    if type(mis)==str:
        lta=Codis(mis)
    else:
        lta=mis
    m=1114112
    c1=clau[0]
    c2=clau[1]
    xif=[(c1*lta[i]+c2)%m for i in range(len(lta))]
    xif=Caracters(xif)
    return xif

```

In [ ]:
```
def AfiDX(mis,clau):
    if type(mis)==str:
        lta=Codis(mis)
        b=1
    else:
        lta=mis
        b=0
    m=1114112
    c1=((clau[0]%m)^(-1))%m
    c2=-c1*clau[1]%m
    pla=[(c1*lta[i]+c2)%m for i in range(len(lta))]
    if b==1:
        pla =Caracters(pla)
    return pla

```

In [ ]:
```
def AfiDXC(mis,clau):
    if type(mis)==str:
        lta=Codis(mis)
    else:
        lta=mis
    m=1114112
    c1=((clau[0]%m)^(-1))%m
    c2=-c1*clau[1]%m
    pla=[(c1*lta[i]+c2)%m for i in range(len(lta))]
    return pla

```

In [ ]:
```
def AfiDXD(mis,clau):
    if type(mis)==str:
        lta=Codis(mis)
    else:
        lta=mis
    m=1114112
    c1=((clau[0]%m)^(-1))%m
    c2=-c1*clau[1]%m
    pla=Caracters([(c1*lta[i]+c2)%m for i in range(len(lta))])
    return pla

```

## A0.2. Criptosistema de Vigenère

La funció VigenereX(miss,clau) s'aplica a un missatge miss en format de tira de caràcters o bé en format de llista de codis, i una clau, també en format de tira de caràcters o bé en format de llista de codis, i retorna el missatge xifrat amb la clau en el mateix format que l'entrada.

a funció VigenereDX(miss,clau) s'aplica a un missatge xifrat miss en format de tira de caràcters o bé en format de llista de codis, i una clau, també en format de tira de caràcters o bé en format de llista de codis, i retorna el missatge desxifrat amb la clau exclusivament en format de tira de caràcters.

In [ ]:
```
def VigenereX(mis,clau):
    m=1114112
    if type(mis)==str:
        lta=[ord(i) for i in mis]
        b=1
    else:
        lta=mis
        b=0
    if type(clau)==str:
        cc=[ord(i) for i in clau]
    else:
        cc=clau
    lc=len(cc)
    vgn=[(lta[i]+cc[i%lc])%m for i in range(len(lta))]
    if b==0:
        xif=vgn
    else:
        xif=""
        for i in range(len(vgn)):
            xif=xif+chr(vgn[i])
    return xif

```

In [ ]:
```
def VigenereDX(mis,clau):
    m=1114112
    if type(mis)==str:
        lta=[ord(i) for i in mis]
    else:
        lta=mis
    if type(clau)==str:
        cc=[ord(i) for i in clau]
    else:
        cc=clau
    lc=len(cc)
    vgn=[(lta[i]-cc[i%lc])%m for i in range(len(lta))]
    desx=""
    for i in range(len(vgn)):
        desx=desx+chr(vgn[i])
    return desx

```

## A0.10. RSA

La funció RSA(mc,clau) s'aplica a un nombre enter mc, mòdul n, i una clau RSA (n,e) o (n,d) per a obtenir el missatge mc xifrat o desxirat, segons la clau.

In [ ]:
```
def RSAX0(mc,clau):
    return [mod(mc[i],clau[0])^clau[1] for i in range(len(mc))]
```

## A0.12. ElGamal

La funció ElGamalX0(mc,claupublica) s'aplica a una clau pública per al criptosistema Elgamal (una llista adequada $[p, q, g, h]$) i a una llista mc de missatges (classes mòdul p de nombres enters), i propotciona el missatge xifrat amb aquesta clau; és a dir, una llista de parelles de la forma [c1,c2], on $c1 = g^y \pmod p$, i $c2 = m \cdot h^y \pmod p$, una per a cada unitat de missatge $m$.

In [ ]:
```
def ElGamalX0(mc,claupublica):
    [p,q,g,h]=claupublica
    l=len(mc)
    return [[Mod(g,p)^(y:=ZZ.random_element(1,q)),mod(mc[i]*Mod(
```

La funció ElGamalDX0(mx,clauprivada) s'aplica a una clau privada per al criptosistema Elgamal (una llista adequada $[p, q, g, x]$) i a una llista mx de missatges xifrats (parelles de classes mòdul p de nombres enters), i propotciona el missatge desxifrat amb aquesta clau; és a dir, una llista nombres enters $m$ mòdul $p$ tals que les parelles $c1 = g^y \pmod p$, $c2 = m \cdot h^y \pmod p$, de mx en són el missatge xifrat.

In [ ]:

```
def ElGamalDX0(mx,clauprivada):
    [p,q,g,x]=clauprivada
    l=len(mx)
    return [Mod(mx[i][1],p) * Mod(mx[i][0],p)^(-x) for i in rang
```

## Fi de l'apèndix A0