Pràctica 2: Codis Cíclics

1. Primer repassarem com treballar amb paraules-codi i polinomis sobre $GF(2)$. Executeu les següents instruccions i observeu el resultat obtingut.

   (a) Necessitem definir un cos base: `F2=GF(2); F2`.
   (b) Es pot definir una paraula-codi com un vector: `v=vector(F2,[1,0,1,1]); v`
   (c) Obtenir el pes de Hamming d'un vector: `w=v.hamming_weight(); w`
   (d) I convertir un vector a una llista: `v.list()`.
   (e) Podem definir una matriu,
   `G = matrix(F2, [(0,1,0,1,0),(0,1,1,1,0),(0,0,1,0,1),(0,1,0,0,1)]); G`
   (f) I el codi lineal amb aquesta matriu: `C=LinearCode(G); C`
   (g) Construcció de l'anell de polinomis sobre $GF(2)$: `Z2X.<x>=PolynomialRing(F2); Z2X`.
   (h) Definir un polinomi amb coeficients a $GF(2)$: `pX=Z2X(1+x^2+x^4+x^5), pX`.
   (i) Alternativament: `pX=Z2X([1,0,1,0,1,1])`.
   (j) Extreure els coeficients d'un polinomi: `v=vector(F2,pX), v`.
   (k) Desplaçament dels coeficients d'un polinomi (multiplicar per `x^2`): `pX.shift(2)`.
   (l) També en l'altre sentit (dividir per `x`), `pX.shift(-1)`.
   (m) Operar amb polinomis: `qX=pX*(x+1); qX`.
   (n) Comprovar si un polinomi és irreductible: `pX.is_irreducible(), qX.is_irreducible()`.
   (o) I si no ho és, descompondre'l en factors: `qX.factor()`.
   (p) Construir un codi cíclic de longitud 3 i `g(x) = x+1`: `C=codes.CyclicCode(x+1,3); C`.
   (q) Veure el conjunt de paraules d'un codi: `S = set(C.list()); S`.

2. Considereu la paraula-codi determinada per la llista de bits: `[1,0,1,1]`. Definiu la funció `UAB_right_shift(s,L)` que permeti obtenir un cíclic shift de s ($s \geq 0$) posicions de la llista `L`. Per exemple, `UAB_right_shift(1,[1,0,1,1])=[1,1,0,1]`. (**1 punt**)

3. Implementeu la funció `UAB_code_is_cyclic(M)` que determini si un codi és cíclic. Com a paràmetre, aquesta funció rebrà la llista `M` de totes les paraules del codi. La funció ha de retornar `True` o `False`. (**1 punt**)

4. Implementeu la funció `UAB_code_dimension_distance(gx,n)` que calculi la dimensió i la distància mínima d'un codi lineal cíclic definit a partir del polinomi generador `gx` i la longitud del codi `n`. Per implementar aquesta funció, **no** podeu utilitzar les següents funcions ja existents a Sage: `minimum_distance()` i `dimension()`.

   La funció ha de retornar una llista $[k, d]$ on k és la dimensió i d és la distància mínima. (**1 punt**)

5. Implementeu la funció `UAB_Generators_of_length_10()` que trobi tots els possibles polinomis generadors `gx` de codis lineals cíclics de longitud 10. La funció ha de retornar aquests polinomis generadors en una llista.

   Podeu utilitzar la funció `polynomials`, com hem vist a la Pràctica 1, per implementar aquesta funció. (**1,5 punts**)

6. Codificació de missatges.

   (a) Implementeu la funció `UAB_message_coding_with_gx(gx,n,v)` que permeti codificar vectors d'informació de mida k. El resultat ha de donar-se també en forma de vector (no utilitzant la seva representació polinomial). Com a paràmetres, aquesta funció rebrà el polinomi generador `gx`, la longitud `n` i el vector d'informació a codificar. Per a fer el càlcul, **no** podeu utilitzar la matriu generadora del codi. (**1,5 punts**)

   (b) Implementeu la funció `UAB_message_coding_with_G(gx,n,v)` que codifiqui, com abans, vectors d'informació de mida k. Aquest cop, però, la vostra funció sí ha d'utilitzar la matriu generadora del codi per calcular el vector a retornar. (**1 punt**)

7. Matriu de control i taula estàndard.

   (a) Implementeu la funció `UAB_standard_array()` que calculi una matriu de control i la taula estàndard del codi cíclic de longitud 7 generat pel polinomi $x^3 + x^2 + 1$. (**1,5 punts**)

   La funció ha de retornar una llista amb dos elements:

   - Matriu de control
   - Taula estàndard. Aquesta taula la podeu implementar a través del tipus de dades *Dictionary* de Python, fent que les claus del diccionari siguin les síndromes (*sindrome: vector error*). Heu de tenir en compte que els vectors s'hauran de passar a tuples abans de poder-se guardar en el diccionari. Per a fer-ho, podeu utilitzar la funció `tuple()`.

   (b) Implementeu la funció `UAB_correct_with_standard_array(m)` que, a partir de la taula estàndard anterior, corregeixi errors en vectors rebuts i retorni les paraules-codi corresponents. Per poder utilitzar aquesta taula, haureu de fer una crida a `UAB_standard_array()` dins de la pròpia funció.

   Com a paràmetre, la funció rebrà el vector m rebut, i retornarà la paraula-codi corresponent després de corregir els possibles errors. En cas de no poder-se corregir l'error, la funció retornarà `None`. (**1,5 punts**)