PROJECTS

Project 1 (2pt). Implement a function which, for a given a code C, returns some properties of the code, i.e. Is C MDS? Is C perfect? Is C self-dual? Is C a divisible code? Etc.

A divisible code C is a code such that there exist a constant c which divides all the weights of C.

Compare the outputs with the function already implemented in MAGMA, like `IsMDS`, `IsPerfect`, `IsSelfDual`, etc.

COMMENTS:

- *Do not use* MAGMA *functions in your implementation which return properties of the code.*
- *You can use the functions* `Binomial`, `Dual`, `GeneratorMatrix` *and* `ParityCheckMatrix`.

Project 2 (2pt)**.** Implement the following functions.

(1) A function `Puncture(C,i)`:

INPUT:	An $\mathbb{F}_q - [n,k,d]$ linear code C and an integer $1 \leq i \leq n$.
OUTPUT:	The code C' obtained by puncturing the i-th coordinate from each codeword of C.

(2) A function `Shorten(C,i)`:

INPUT:	An $\mathbb{F}_q - [n,k,d]$ linear code C and an integer $1 \leq i \leq n$.
OUTPUT:	The code $\overline{C}$ obtained by shortening the i-th coordinate from each codeword of C.

Compare the outputs of your functions with them of the functions `PunctureCode(C,i)` and `ShorthenCode(C,i)`, already present in Magma.

COMMENTS:
Do not use the Magma *functions* `PunctureCode(C,i)` *and* `ShorthenCode(C,i)` *in your implementation.*

Project 3 (2pt)**.** Implement the following functions.

(1) A function `Sum(C,D)`:

INPUT:	An $\mathbb{F}_q - [n, k_1, d_1]$ linear code C and an $\mathbb{F}_q - [n, k_2, d_2]$ linear code D.
OUTPUT:	The direct sum code E of C and D. *(Check Problemsheet 5 - Ex 5.)*

(2) A function `Plotkin(C,D)`:

INPUT:	An $\mathbb{F}_q - [n, k_1, d_1]$ linear code C and an $\mathbb{F}_q - [n, k_2, d_2]$ linear code D.
OUTPUT:	The Plotkin sum code E of C and D. *(Check Problemsheet 5 - Ex 6.)*

Compare the outputs of your functions with them of the functions `DirectSum(C,D)` and `PlotkinSum(C,D)`, already present in MAGMA.

COMMENTS:
Do not use the functions `DirectSum(C,i)` *and* `PlotkinSUm(C,i)` *in your implementation.*

Project 4 (3pt)**.** Investigate the properties of the $[23, 12, 7]$ Golay code and of the $[24, 12, 8]$ extended Golay code over $\mathbb{F}_2$. Compute their weight distribution. Are they divisible-codes? Compare the properties of the two codes. Do they attain any bound? Etc.

A divisible code C is a code such that there exist a constant c which divides all the weights of C.

<u>COMMENTS</u>:
Check the bounds you studied and find other bounds in the literature. You can use the function `GolayCode` *to construct the codes.*

Project 5 (3pt)**.** Implement a function that simulate a transmission of a message through a noisy channel.

INPUT:	A message vector $\boldsymbol{m} \in \mathbb{F}_q^k$ and an $\mathbb{F}_q - [n, k, d]$ linear code C.
OUTPUT:	The message vector $\boldsymbol{m}$, the code C, the codeword $\boldsymbol{c}$ associated to $\boldsymbol{m}$, the received vector $\boldsymbol{r}$ and a string which says if the decoding was successful of not. In case of successful decoding, the function should also return the error $\boldsymbol{e}$ and the decoded word $\overline{\boldsymbol{c}}$.

Compare the outputs of your functions with them of the function `Decode(C,y)`, already present in MAGMA.

COMMENTS:
Do not use the function `Decode(C,y)` *in your implementation.*

Project 6 (2pt+2pt)**.** An Hadamard matrix H_{2^n} is a $2^n \times 2^n$ square matrix whose entries are either $+1$ or -1 and whose rows are mutually orthogonal. H_{2^n} satisfies $H_{2^n} H_{2^n}^t = 2^n I_{2^n}$. It is possible to construct an Hadamard matrix recursively. Indeed,

$$H_{2^n} = \begin{bmatrix} H_{2^{n-1}} & H_{2^{n-1}} \\ H_{2^{n-1}} & -H_{2^{n-1}} \end{bmatrix}$$

with $H_1 = [1]$.

Implement a function that construct the $2^n \times 2^n$ Hadamard matrix using the recursive approach explained above.

INPUT:	An integer n.
OUTPUT:	The Hadamard matrix H_{2^n}.

It it possible to construct a non-linear $\mathbb{F}_2 - (2^n, 2^{n+1}, 2^{n-1})$ code C using an Hadamard matrix H_{2^n}. The 2^{n+1} codewords of C are the rows of H_{2^n} and the rows of $-H_{2^n}$. Notice that to obtain the binary code C, the mapping $-1 \mapsto 1$, $1 \mapsto 0$ is applied to the matrix elements.

Implement a function that return the $\mathbb{F}_2 - (2^n, 2^{n+1}, 2^{n-1})$ Hadamard code C.

INPUT:	An integer n.
OUTPUT:	The Hadamard code C.

Investigate the properties of these code for some values of n. Are they MDS? Are they perfect? Are they self-dual? Are they divisible codes? Etc.

<u>**COMMENTS**</u>:

- *Do not use the function* `HadamardMatrixFromInteger` *in your implementation.*
- *You can use the functions that returns properties of the code (such as* `IsMDS`, `IsPerfect`, `IsSelfDual`, *etc.) in the investigation part.*

Project 7 (5pt)**.** Implement the algorithm below for decoding the $\mathbb{F}_2 - [24, 12, 8]$ extended Golay code C with generator matrix $G = [I|B]$.

INPUT:	A received vector $\boldsymbol{r} \in \mathbb{F}_2^{24}$.
OUTPUT:	The codeword $\boldsymbol{c}$ obtained by decoding $\boldsymbol{r}$ and the error vector $\boldsymbol{e}$ in case of successful decoding. A request for retransmission, otherwise.

Let $\boldsymbol{e} = (\boldsymbol{e}_L|\boldsymbol{e}_R)$ be the error vector. Notice that, since C is self-dual, G is also a parity-check matrix for C. Therefore, we can easily compute two different syndromes:

$$S_1(\boldsymbol{e}) = \boldsymbol{e}H^t = (\boldsymbol{e}_L|\boldsymbol{e}_R)(B^t|I_{12})^t = \boldsymbol{e}_L B + \boldsymbol{e}_R$$

$$S_2(\boldsymbol{e}) = \boldsymbol{e}G^t = (\boldsymbol{e}_L|\boldsymbol{e}_R)(I_{12}|B)^t = \boldsymbol{e}_L + \boldsymbol{e}_R B^t$$

Notice that $S_2(\boldsymbol{e}) = S_1(\boldsymbol{e})B^t$.

ALGORITHM:

(1) Compute the syndrome $S_1(\boldsymbol{r}) = \boldsymbol{r}H^t = \boldsymbol{r}(B^t|I_{12})^t$.
  - (a) If $\mathrm{wt}(S_1(\boldsymbol{r})) \leq 3$, then the error vector is $e = (0|S_1(\boldsymbol{r}))$ and you can decode.
  - (b) If $\mathrm{wt}(S_1(\boldsymbol{r})) > 3$, then compute $\mathrm{wt}(S_1(\boldsymbol{r}) + B_i)$ for all $i = 1, \dots, 12$, where B_i is the i-th row of B.
    - (i) If $\mathrm{wt}(S_1(\boldsymbol{r})+B_i) \leq 2$ for some i, then the error vector is $e = (S_1(\boldsymbol{r})+B_i|\boldsymbol{\delta_i})$, where $\boldsymbol{\delta_i}$ is the vector in $\mathbb{F}_2^{24}$ with 1 in position i and 0 elsewhere. You can decode.
    - (ii) If $\mathrm{wt}(S_1(\boldsymbol{r}) + B_i) \leq 2$ for more than one i, choose the one(s) with smallest Hamming weight and decode as in point (1)(b)(i).

(2) If $\mathrm{wt}(S_1(\boldsymbol{r}) + B_i) > 3$ for all $i = 1, \dots, 12$, then compute the syndrome $S_2(\boldsymbol{e})$.
  - (a) If $\mathrm{wt}(S_2(\boldsymbol{r})) \leq 3$, then the error vector is $e = (S_2(\boldsymbol{r})|0)$ and you can decode.
  - (b) If $\mathrm{wt}(S_2(\boldsymbol{r})) > 3$, then compute $\mathrm{wt}(S_2(\boldsymbol{r}) + B_i)$ for all $i = 1, \dots, 12$.
    - (i) If $\mathrm{wt}(S_2(\boldsymbol{r})+B_i) \leq 2$ for some i, then the error vector is $e = (\boldsymbol{\delta_i}|S_2(\boldsymbol{r})+B_i)$, and you can decode.
    - (ii) If $\mathrm{wt}(S_2(\boldsymbol{r}) + B_i) \leq 2$ for more than one i, choose the one(s) with smallest Hamming weight and decode as in point (2)(b)(i).

(3) If $\boldsymbol{e}$ is not determined (i.e. if $\mathrm{wt}(S_1(\boldsymbol{r}) + B_i) > 3$ and $\mathrm{wt}(S_2(\boldsymbol{r}) + B_i) > 3$ for all $i = 1, \dots, 12$), then request retransmission.

COMMENTS:

You can use the line `GolayCode(GF(2),true)` *to construct the* $\mathbb{F}_2 - [24, 12, 8]$ *extended Golay code in* MAGMA.

Project 8 (5pt)**.** Let C be an $\mathbb{F}_2 - [2^m, k, 2^{m-r}]$ Reed-Muller code, where $k = \sum_{i=0}^{r} \binom{m}{i}$. Let V the list of the elements of $\mathbb{F}_2^m$ sorted by lexicographic order, $K\langle x_1, \ldots, x_m\rangle$ the multivariate polynomial ring over $\mathbb{F}_2$ with m variables. Every vector $\boldsymbol{y} \in \mathbb{F}_2^{2^m}$ can be represented as

$$\boldsymbol{y} = (f(V_1), f(V_2), \ldots, f(V_{2^m}))$$

for a suitable $f \in K\langle x_1, \ldots, x_m\rangle$. Notice that f is of the form

$$f(x_1, \ldots, x_m) = \sum_{t=0}^{m} \sum_{S \subseteq \{1 \ldots m\}, |S|=t} f_S \prod_{i \in S} x_i \qquad \text{with } f_S \in \mathbb{F}_2 \text{ for all } S \subseteq \{1 \ldots m\}.$$

Implement the Majority Logic Decoder for Reed Muller codes.

INPUT:	The received vector $\boldsymbol{y}$, the parameters r, m of C, the list V defined above.
OUTPUT:	The codeword $\boldsymbol{c}$ obtained by decoding $\boldsymbol{y}$.

For a subset S of $\{1, \ldots, m\}$, a vector $\boldsymbol{a} \in \mathbb{F}_2^t$ and a vector $\boldsymbol{b} \in \mathbb{F}_2^{m-t}$, we define the vector $\boldsymbol{v}_{S,\boldsymbol{a},\boldsymbol{b}}$ of length m whose coordinates in S are given by $\boldsymbol{a}$ and the remaining by $\boldsymbol{b}$.

ALGORITHM:

(1) Find the polynomial $f \in K\langle x_1, \ldots, x_m\rangle$ such that
$$\boldsymbol{y} = (f(V_1), f(V_2), \ldots, f(V_{2^m}))$$
(2) Initialize $p \in K\langle x_1, \ldots, x_m\rangle$ to be 0 and $t = r$.
(3) Do the following for $t \geq 0$.
  (a) Set $f_t = f - p$
  (b) Do the following for every subset S of $\{1, \ldots, m\}$ with $S = t$.
    (i) Create an empty list L_S.
    (ii) Do the following for every $\boldsymbol{b} \in \mathbb{F}_2^{m-t}$.
      - Compute the vector $\boldsymbol{v}_{S,\boldsymbol{a},\boldsymbol{b}}$.
      - Compute the value
$$C_{S,\boldsymbol{b}} := \sum_{\boldsymbol{a} \in \mathbb{F}_2^T} f_t(\boldsymbol{v}_{S,\boldsymbol{a},\boldsymbol{b}}).$$
      - Store the value $C_{S,\boldsymbol{b}}$ in the list L_S.
    (iii) Compute the value C_S as following. Set $C_S := 1$ if the number of 1s is greater or equal then the number of 0s in L_S, set $C_S := 0$ otherwise.
    (iv) Set $p := p - C_S \prod_{i \in S} x_i$.
(4) Return the vector $\boldsymbol{c} := (p(V_1), p(V_2), \ldots, p(V_m))$ if $\boldsymbol{c} \in C$. Ask for a retransmission otherwise.

COMMENTS:

- *Notice that you can only evaluate a multivariate polynomial on a sequence. Therefore in order to evaluate f (or p) on an element V_i, you have to convert this latter in sequence.*
- *You can use the function* `ReedMullerCode(r,m)` to construct the code C.
- *In order to find the polynomial f, you can solve a system of linear equations. You can use the functions* `Solution` or `EchelonForm`*. In order to evaluate a polynomial in an element, you can use the function* `Evaluate`*.*

Project 9 (5pt)**.** Let C be an $\mathbb{F}_{q^m}-[q^m-1, k, d]$ Reed-Solomon code and let $\{1, \alpha, \alpha^2, \dots, \alpha^{q^m-2}\}$ a set of evaluation point, where α is a primitive element of $\mathbb{F}_{q^m}/\mathbb{F}_q$. Let $\boldsymbol{r}$ a received vector, then we think of $\boldsymbol{y}$ as the set of ordered pairs $\{(1, r_1), (\alpha, r_2), \dots, (\alpha^{q^m-2}, r_{q^m-1})\}$

Implement the Welch-Berlekamp algorithm for decoding Reed-Solomon codes, under the assumption that we know the weight of the error vector $\boldsymbol{e}$.

INPUT:	$\mathrm{wt}(e) < t$ and the ordered pairs $\{(\alpha^{i-1}, r_i\}_{i=1}^{q^m-1}$ associated to the received vector $\boldsymbol{r}$.
OUTPUT:	The codeword $\boldsymbol{c}$ obtained by decoding $\boldsymbol{r}$ or a request of retransmission.

ALGORITHM:

(1) Compute the polynomial $E(x)$ of degree $\mathrm{wt}(\boldsymbol{e})$ and the polynomial $Q(x)$ of degree $\mathrm{wt}(\boldsymbol{e}) + k - 1$ such that

$$y_i E(\alpha_{i-1}) = Q(\alpha_{i-1})$$

for all $i = 1, \dots, q^m - 1$.

(2) If $E(x)$ and $Q(x)$ as above do not exist or $E(x)$ does not divide $Q(x)$, then ask for a retransmission.

(3) If $E(x)$ and $Q(x)$ as above exist and $E(x)$ divides $Q(x)$, than set $P(x) := \frac{Q(x)}{E(x)}$.

(4) Create the vector $\boldsymbol{p} := (P(1), P(\alpha), \dots, P(\alpha^{q^m-1}))$.

(5) If $d(\boldsymbol{y}, \boldsymbol{p}) \leq \mathrm{wt}(e)$ then return $\boldsymbol{p}$ as the decoded codeword. Otherwise, ask for a retransmission.

COMMENTS:

- *You can use the function* `ReedSolomonCode` to construct the code C.
- *In order to find the polynomial $E(x)$ and $Q(x)$, you can solve a system of linear equations. You can use the functions* `Solution` or `EchelonForm`. *In order to evaluate a polynomial in a element, you can use the function* `Evaluate`.

Project 10 (5pt)**.** Let C be an $\mathbb{F}_2 - [2^m-1, k, d]$ BCH code and α be a primitive element of $\mathbb{F}_{2^m}/\mathbb{F}_2$, i.e. $\mathbb{F}[\alpha] = \mathbb{F}_{2^m}$. Define for every vector $\boldsymbol{v} \in \mathbb{F}_2^{2^m-1}$ the polynomial $f_{\boldsymbol{v}}(x) := \sum_{i=0}^{2^m-1} c_i x^i$. Suppose the received vector $\boldsymbol{r}$, then the syndrome vector if

$$\boldsymbol{s} := (f_{\boldsymbol{r}}(\alpha), f_{\boldsymbol{r}}(\alpha^2), \ldots, f_{\boldsymbol{r}}(\alpha^{2t}))$$

where t is the correction capability of C.

Implement a decoding algorithm for binary BCH codes based on the Berlekamp-Massey algorithm.

INPUT:	The syndrome vector $\boldsymbol{r}$.
OUPUT:	A codeword $\boldsymbol{c}$ obtained by decoding $\boldsymbol{r}$ or a request for retransmission.

Given the syndrome vector $\boldsymbol{s}$, the Berlekamp-Massey algorithm finds the associated **locator polynomial**

$$c(x) := c_0 + c_1 x + \ldots + c_{2^m-1} x^{2^m-1} \in \mathbb{F}_{2^m}[x].$$

The roots of this polynomial give information about the location of errors. In particular, if α^i is a root for $c(x)$ then there is an error in $\boldsymbol{r}$ in position j where $\alpha^j = (\alpha^i)^{-1}$.

ALGORITHM:

(1) Find the syndrome vector $\boldsymbol{s}$ associated to $\boldsymbol{r}$.
(2) Initialize the following parameters:
  - $L = 0$, it represent the length of the LFSR;
  - $c(x) = 1$, it will be the locator polynomial;
  - $p(x) = 1$, it represent the locator polynomial before last length change;
  - $l = 1$, it represent the amount of shift in update;
  - $d_m = 1$, it represent the previous discrepancy.

(3) Do the following for $k = 1, \ldots, 2t$ in steps of 2 (i.e. $k = 1, 3, \ldots$).
  (a) Compute the discrepancy
  $$d := \boldsymbol{s}_k + \sum_{i=1}^{L} c_i \boldsymbol{s}_{k-i}.$$
  (b) If $d = 0$ then increase the shift l by 1.
  (c) If $d \neq 0$ then
    (i) If $2L \geq k$ then set $c(x) = c(x) - d d_m^{-1} x^l p(x)$ and increase the shift l by 1.
    (ii) Otherwise, temporary store the polynomial $p(x)$, that is define $t(x) = c(x)$, set $c(x) = c(x) - d d_m^{-1} x^l p(x)$ and then $p(x) = t(x)$. Finally, set $L = k - L$, $d_m = d$ and $l = 1$.

  Increase the shift l by 1 and go back to point (2).
(4) Find the multiplicative inverse of the roots of $c(x)$.
(5) Compute the error vector and decode.
(6) If the decoded vector is not a codeword of C, then ask for a retransmission.

COMMENTS:

- *Notice that,* MAGMA *indexes sequences starting from* 1 *and not from* 0*. Therefore if* α^i *is a root for* $c(x)$ *and* $\alpha^j = (\alpha^i)^{-1}$*, then there will be an error in* $\boldsymbol{r}$ *in position* $j+1$.
- *You can use the function* `BCHCode` to construct the code and the function `Coefficients` to retrieve the coefficients of a given polynomial.