Домашно упражнение №1

Петко Борджуков (ФN№ 61322)

Задача 1

За да решим задачата, дефинираме класовете *HadamardMatrix* и *GolayCode*. Класът *HadamardMatrix* няма полета и предлага на потребителя два статични метода за създаване на матрица на Адамар – *#paley(order)* и *#sylvester(order)*.

Класът *GolayCode* има полета *dimension* и *matrix*, съдържащи съответно размерността на кода и матрицата му. Инстанция на този клас може да се създаде, чрез подаване на желана размерност чрез извикване на *GolayCode#new(dimension)*.

Отпечатване на матрица на Адамар от ред n, съставена по метода на Пейли

Методът *HadamardMatrix#paley(order)* съставя матрица по метода на Пейли, както е описан в Двоични шумозащитни кодове, 2004, Е. Великова-Бандова.

При подаване на ред, матрица от който не може да бъде създадена по този метод, програмата връща грешка:

```
[38] pry(main)> HadamardMatrix.paley 18
RuntimeError: The given order 18 is not valid
```

При подаване на ред, матрица от който може да бъде създадена по метода на Пейли, програмата връща инстанция на класа *Matrix*, която отговаря на условията, изисквани, за да бъде наречена Адамарова:

```
[43] pry(main)> m = HadamardMatrix.paley(12).to_a
=> [
 [ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
 [ 1,-1, 1,-1, 1, 1, 1,-1,-1,-1, 1,-1],
 [ 1,-1,-1, 1,-1, 1, 1, 1,-1,-1,-1, 1],
 [ 1, 1,-1,-1, 1,-1, 1, 1, 1,-1,-1,-1],
 [ 1,-1, 1,-1,-1, 1,-1, 1, 1, 1,-1,-1],
 [ 1,-1,-1, 1,-1,-1, 1,-1, 1, 1, 1,-1],
 [ 1,-1,-1,-1, 1,-1,-1, 1,-1, 1, 1, 1],
 [ 1, 1,-1,-1,-1, 1,-1,-1, 1,-1, 1, 1],
 [ 1, 1, 1,-1,-1,-1, 1,-1,-1, 1,-1, 1],
 [ 1, 1, 1, 1,-1,-1,-1, 1,-1,-1, 1,-1],
 [ 1,-1, 1, 1, 1,-1,-1,-1, 1,-1,-1, 1],
 [ 1, 1,-1, 1, 1, 1,-1,-1,-1, 1,-1,-1]
]
```

```
[52] pry(main)> m.each {|row| row.collect!\
   {|val| val == -1 ? "*" : " "}; p row.join ' '}
"                       "
"  *   *       * * *   *"
"  * *   *       * * *  "
"    * *   *       * * *"
"  *   * *   *       * *"
"  * *   * *   *       *"
"  * * *   * *   *      "
"    * * *   * *   *    "
"      * * *   * *   *  "
"        * * *   * *   *"
"  *       * * *   * *  "
"    *       * * *   * *"
```

Отпечатване на матрица на Адамар от ред n, съставена по метода на Силвестър

Методът *HadamardMatrix#sylvester(order)* рекурсивно съставя матрица, чрез удвояване на реда.

При подаване на ред, матрица от който не може да бъде създадена по този метод, програмата връща грешка:

```
[60] pry(main)> HadamardMatrix.sylvester 12
RuntimeError: The given order 12 is not valid
```

При подаване на валиден ред, методът връща инстанция на класа *Matrix*, която отговаря на условията, изисквани, за да може да бъде наречена Адамарова.

```
[63] pry(main)> m = HadamardMatrix.sylvester(32).to_a
[65] pry(main)> m.each {|row| row.collect! {|val| val == -1 ? "*" : " "}; p row.join ' '}
"                                                                "
"  *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *"
"    * *     * *     * *     * *     * *     * *     * *     * *"
"  * *     * *     * *     * *     * *     * *     * *     * *  "
"        * * * *         * * * *         * * * *         * * * *"
"  *   * *   *     *   * *   *     *   * *   *     *   * *   *  "
"    * * * *         * * * *         * * * *         * * * *    "
"  * *   *     *   * *   *     *   * *   *     *   * *   *     *"
"                  * * * * * * * *                 * * * * * * * *"
"  *   *   *   * *   *   *   *     *   *   *   * *   *   *   *  "
"    * *     * * * *     * *         * *     * * * *     * *    "
"  * *     * *   *     * *     *   * *     * *   *     * *     *"
"        * * * * * * * *                 * * * * * * * *        "
"  *   * *   *     *   * *   *   *   *     *   * *   *     *   *"
"    * * * *         * * * *     * *         * * * *         * *"
"  * *   *     *   * *   *     * *     *   * *   *     *   * *  "
"                  * * * * * * * * * * * * * * * *              "
"  *   *   *   * *   *   *   *   *   *   *     *   * *   *   *  "
"    * *     * * *     * *     * * * *     * *     * *     * *    "
"  * *     * *     * *     * *   *     * *     * *     * *     *"
"        * * * *         * * * * * * * *         * * * *        "
"  *   * *   *     *   * *   *   *   *     *   * *   *     *   *"
"    * * * *         * * * *     * *         * * * *         * *"
"  * *   *     *   * *   *     * *     *   * *   *     *   * *  "
"                  * * * * * * * * * * * * * * * *              "
"  *   *   *   * *   *   *   *   *   *     *   * *   *   *  "
"    * *     * * * *     * *     * *     * *     * *     * *"
"  * *     * *   *     * *     * *     * *     * *     * *  "
"        * * * * * * * *         * * * *                 * * * *"
"  *   * *   *   *   *     *   * *   *     *   *   *   * *   *  "
"    * * * *     * *         * * * *         * *     * * * *    "
"  * *   *     * *     *   * *   *     *   * *     * *   *     *"
```

```
[66] pry(main)> m = HadamardMatrix.sylvester(64)
[69] pry(main)> m * m.transpose == Matrix.scalar(64, 64)
=> true
```

Съставяне на удължен код на Голей

Методът *GolayCode#new(dimension)* съставя матрица на Адамар от ред *dimension* и с нейна помощ образува матрицата на разширения код на Голей с размерност *dimension*.

```
[2] pry(main)> g = GolayCode.new 12
[3] pry(main)> g.matrix.to_a
=> [
 [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0],
 [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1],
 [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1],
 [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1],
 [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1],
 [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1],
 [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1],
 [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1],
 [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1]
]
```

```
[7] pry(main)> g.matrix.to_a.each {|row| \
        row.collect! {|val| val == 1 ? '*' : " "}; p row.join ' '}
"*                       * * * * * * * * * * * *  "
"  *                     * *   * * *       *   *"
"    *                     * *   * * *       * *"
"      *                   *   * *   * * *       *"
"        *                   *   * *   * * *     *"
"          *                   *   * *   * * *   *"
"            *                   *   * *   * * * *"
"              *          *       *   * *   * * *"
"                *        * *       *   * *   * *"
"                  *      * * *       *   * *   *"
"                    *      * * *       *   * * *"
"                      * *   * * *       *   * *"
```

Изходен код

hadamard_matrix.rb

```
#UTF-8

require 'matrix'
require 'prime'

class HadamardMatrix
  H1 = Matrix[[1]]
  H2 = Matrix[[1, 1],[1,-1]]

  def HadamardMatrix.paley(order)
    raise "The given order #{order} is not valid" unless is_valid order
    return H1 if order == 1
    return H2 if order == 2

    raise "The given order #{order} is not valid" unless Prime.prime? order - 1
    p = order - 1

    expand Q(p) - Matrix.I(p)
  end

  def HadamardMatrix.sylvester(order)
    raise "The given order #{order} is not valid" unless is_valid order
    raise "The given order #{order} is not valid" unless is_square? order
    return H1 if order == 1
    return H2 if order == 2
    lower_order = sylvester(order/2)
    parent1 = parent2 = parent3 = lower_order.to_a
    parent4   = (-1 * lower_order).to_a
    daughter = []
    parent1.each_index {|index| daughter << parent1[index] + parent2[index]}
    parent3.each_index {|index| daughter << parent3[index] + parent4[index]}
    Matrix.rows(daughter)
  end

  private
  def initialize
  end
  def HadamardMatrix.is_valid(order)
    return false if !order.integer?
    return false if order <= 0
    return true  if order <= 2
    return true  if order.divmod(4)[1] == 0
    return false
  end
  def self.quadratic(p)
    Array.new(p/2) {|index| ((index+1)**2).divmod(p)[1] }
  end
  def self.is_square?(number)
    (number & (number - 1)) == 0
  end
```

```
  def self.Q(p)
    quadratic_remainders = quadratic p
    chi = ->(a) do
      quotient, remainder = a.divmod(p)
      return  0 if remainder == 0
      return  1 if quadratic_remainders.include? remainder
      return -1
    end
    Matrix.build(p, p) {|i, j| chi.(j - i)}
  end

  def self.expand(matrix)
    rows = matrix.to_a.each {|row| row.unshift 1 }
    rows.unshift Array.new(matrix.row_size + 1, 1)
    Matrix.rows rows
  end
end
```

golay_code.rb

```
#UTF-8
require './hadamard_matrix.rb'

class GolayCode
  attr_reader :dimension, :matrix, :minimum_distance

  def initialize(dimension)
    @dimension = dimension
    @minimum_distance = dimension/2 + 2
    construct_matrix
  end

  private
  def construct_matrix
    right = a.collect { |row| row << 1 }
    right.unshift(Array.new(@dimension - 1, 1) << 0)
    left = Matrix.I(@dimension).to_a
    @matrix = Matrix.rows left.zip(right).each {|row| row.flatten!}
  end
  def a
    begin
      hadamard = HadamardMatrix.paley(@dimension).to_a
    rescue
      raise "Invalid dimension"
    end
    hadamard.shift
    hadamard.reverse!.each do |row|
      row.shift
      row.reverse!.collect! do |value|
        if value == -1
          1
        elsif value == 1
          0
        end
      end
    end
  end
end
```

Спецификация на тестовете

Изходният код, изведен по-горе, удовлетворява следните тестове:

```
GolayCode
  can be instantiated only by passing a valid code dimension
  #matrix
    returns the generator matrix of the code
  #minimal_distance
    returns the minimal distance of the code
```

```
HadamardMatrix
  cannot be instantiated
  #paley
    behaves like a Hadamard matrix constructor
      when passed an invalid order
        raises an error
      when passed a valid order
        does not raise an error
        constructs a matrix
        constructs a square matrix
        constructs a matrix that contains only 1s and -1s
        constructs a matrix that contains only mutually orthogonal rows
        constructs a matrix that matches the given reference Hadamard matrix
  #sylvester
    behaves like a Hadamard matrix constructor
      when passed an invalid order
        raises an error
      when passed a valid order
        does not raise an error
        constructs a matrix
        constructs a square matrix
        constructs a matrix that contains only 1s and -1s
        constructs a matrix that contains only mutually orthogonal rows
        constructs a matrix that matches the given reference Hadamard matrix
```

Задача 2

Нека имаме кода C със следната пораждаща матрица:

$$G = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

- Размерността на кода C е $k = 3$; минималното му разстояние е $d(C) = wt(C) = 3$.
- Стандартната таблица на Слепян за кода C:

(0,0,0,0,0,0)	(0,0,0,1,1,1)	(0,1,1,0,0,1)	(0,1,1,1,1,0)	(1,0,1,0,1,1)	(1,0,1,1,0,0)	(1,1,0,0,1,0)	(1,1,0,1,0,1)
(0,0,0,0,0,1)	(0,0,0,1,1,0)	(0,1,1,0,0,0)	(0,1,1,1,1,1)	(1,0,1,0,1,0)	(1,0,1,1,0,1)	(1,1,0,0,1,1)	(1,1,0,1,0,0)
(0,0,0,0,1,0)	(0,0,0,1,0,1)	(0,1,1,0,1,1)	(0,1,1,1,0,0)	(1,0,1,0,0,1)	(1,0,1,1,1,0)	(1,1,0,0,0,0)	(1,1,0,1,1,1)
(0,0,0,1,0,0)	(0,0,0,0,1,1)	(0,1,1,1,0,1)	(0,1,1,0,1,0)	(1,0,1,1,1,1)	(1,0,1,0,0,0)	(1,1,0,1,1,0)	(1,1,0,0,0,1)
(0,0,1,0,0,0)	(0,0,1,1,1,1)	(0,1,0,0,0,1)	(0,1,0,1,1,0)	(1,0,0,0,1,1)	(1,0,0,1,0,0)	(1,1,1,0,1,0)	(1,1,1,1,0,1)
(0,1,0,0,0,0)	(0,1,0,1,1,1)	(0,0,1,0,0,1)	(0,0,1,1,1,0)	(1,1,1,0,1,1)	(1,1,1,1,0,0)	(1,0,0,0,1,0)	(1,0,0,1,0,1)
(1,0,0,0,0,0)	(1,0,0,1,1,1)	(1,1,1,0,0,1)	(1,1,1,1,1,0)	(0,0,1,0,1,1)	(0,0,1,1,0,0)	(0,1,0,0,1,0)	(0,1,0,1,0,1)
(0,0,1,0,1,0)	(0,0,1,1,0,1)	(0,1,0,0,1,1)	(0,1,0,1,0,0)	(1,0,0,0,0,1)	(1,0,0,1,1,0)	(1,1,1,0,0,0)	(1,1,1,1,1,1)

  - Векторът (0,1,1,0,1,0) се декодира до думата (0,1,1,1,1,0) с грешка (0,0,0,1,0,0).
  - Векторът (0,1,0,1,0,1) се декодира до думата (1,1,0,1,0,1) с грешка (1,0,0,0,0,0).
  - Векторът (1,1,1,0,0,0) не може да се декодира еднозначно до кодова дума.

Задача 3

Ако двоичният линеен код C с нотация $[n, k]$ съдържа думи с нечетно тегло, то те са точно 2^{k-1} на брой.