Solve exercises using C#. Answers will be on source code files. Document your solution inside source codes. You can use GITHub if desire to share source codes.

Exercise 1:

Given Generic Class Vector, with its definition as next:

```
#include "ExcepctionVectorRange.h"
template <typename T, long msize=10>
class Vector {
	T array[msize];
	int nelem;

	public: // Implementation of methods is omitted
	Vector() : nelem(0) {}
	~Vector() { nelem=0; }
	bool addElement(const T& elem) {
		if (nelem<msize) {
			array[nelem++]=elem; return true;
			} else return false; }
	T operator[](int n) const {
	if (n<nelem) return array[n];
	else throw ExcepctionVectorRange (n, nelem);
	 }
	int getNumElements() const { return nelem; }
	int getCapacity() const { return msize; }
};
```

Questions:
(a) Implement a class member method called msearch which, given an object of type T, returns true if the element is stored in the vector or false otherwise. As an example, the following code should work without errors: int main () {Vector <double> vd; ... if (vd.msearch (0.0)) {...}}
(b) Indicate what requirements type T must fullfill in order to be used with the Vector class (including the msearch method).
(c) Implement the same method to search as a non-member function of the Vector class.
(d) Define and implement the ExcepctionVectorRange class, so that the following code compiles and functions without errors:

```
int main() {

Vector<string> vs;

while (vs.addElement("Hello"));
try {
	for (int i=vs.getNumElementos(); i>=0; i++)
	std::cout << vs[i] << endl;
	} catch (ExcepctionVectorRange &ex) { std::cerr << ex.what() << "posicion: " << ex.getPos() << ",
num. elements: " << ex.getNumElem() << endl; }
}
```

Exercise 2:

Create and load a matrix of 4 rows by 4 columns of decimal numbers. Program must to print the main diagonal in binary code.

x - - -
- x - -
- - x –
- - - x

Exercise 3:

To solve this exercise you must to do:

a) Create class SpanishPlayingCard with number (1-12) and sign ("Oros/Golden","Copas/Cups","Espadas/Swords", "Bastos/Sticks").

Constructor of the class will receive as parameters number and sign.
Methods:
WriteCardValue, that shows number and sign.

b) Create Class DeckOfCards as member a list of SpanishPlayingCard. Constructor of the class must to create a Deck of Cards with the 48 SpanishPlayingCard.

This class must to include the next methods:
Number of Cards in the pack of cards.
Take Card from the pack
Take the card in the position n
Take a random card
Write all cards remaining in the Deck of Cards
Mix the cards randomly in the deck of cards

c) Write a program that create a deck of cards using the previous classes and execute

1: Show the deck of cards
2: Mix the deck of cards
3: Show after it the deck of cards
4: Take Card from the pack
5: Take the card in the position 5
6: Take a random card
7: Show the number of cards remaining
8: Show the deck of cards.