Laboratorium 5

1. Ćwiczenie 1 Pętla For

Napisz program, korzystając z każdego typu pętli (for, while, do while) wypisujący wzór trójkątny z liczb o zadanej z klawiatury wielkości. Np. dla 5:

```
1
12
123
1234
12345
```

Oraz drugi, który wypisuje następujący wzór trójkątny, np. dla 5;

```
1
22
333
4444
55555
```

2. Ćwiczenie 2 Boxing i unboxing

Wykonaj następujący fragment kodu:

```
Int32 i = 23;
object o = i;
i = 123;
Console.WriteLine(i + ", " + (Int32) o);
```

1. Uzasadnij, czemu wypisane wartości są takie a nie inne.
2. Co się stanie (i dlaczego) jeśli dokonamy konwersji :

```
long j = (long)o;
```

3. Ćwiczenie 3 Typy Nullable

Zdefiniuj zmienną *nullabe* (dowolną). Na początku jej nie inicjalizuj.

1. Spróbuj wypisać jej wartość w konsoli.
2. Zastosuj metody `T GetValueOrDefault(T defaultValue)`, `HasValue`, aby wypisać jej zawartość bez generowania wyjątku.
3. Przypisz jej wartość i spróbuj wypisać ją ponownie

4. Ćwiczenie 4 Typy Nullable

Zdefiniuj dwie zmienne typu – zwykłą i nullable. Spróbuj je porównać za pomocą operatorów <,=,>. Jaki jest efekt ?

```
int? i = null;
int j = 10;
```

5. Ćwiczenie 5 P/Invoke

W programie c# wywołaj funkcję puts(char*) oraz _flushall(void) (usuwa wszystkie bufory związane ze strumieniami wejściowymi) zawartą w bibliotece **msvcrt.dll.**
Z ich pomocą wypisz w konsoli dowolny łańcuch znaków. Nie zapomnij dołączyć przestrzeni System.Runtime.InteropServices;

6. Ćwiczenie 6 Elementy obiektowości – klasa Stack

Zaimplementuj klasę **Stack**, posiadającą następujące metody.

1. AddItem – dodaje na górze stosu.
2. DeleteItem – usuwa element z samego dołu stosu.
3. ShowTheNumberOfItems.
4. ShowMinItem.
5. ShowMaxItem.
6. FindAnItem – zwraca numer na stosie od dołu jeśli istnieje podany. Jeśli brak to -1.
7. PrintAllItems.
8. ClearAll.

Utwórz trzy stosy, spośród których dwa są już zainicjowane 100 losowymi wartościami, a trzeci jest pusty. Przenieś na pusty wartości parzyste z dwóch już zapełnionych, ale bez powtórzeń.

7. Ćwiczenie 7 Elementy obiektowości – Klasa Matrix

Zaimplementuj klasę **Matrix**, przechowującą elementy int w strukturze int [].

```
public class Matrix
{
 int[]_matrix;
 int c;
 int l;
 //...
 }
```

1. Zdefiniuj konstruktory dla obiektu **Matrix**, które umożliwią:
   a. Utworzenie obiektu o określonym rozmiarze z tablicą zainicjowaną podanymi wartościami. W przypadku, gdy elementów inicjujących jest za mało lub za dużo, to do tablicy są dopisywane zera lub nadmiarowe elementy są obcinane.
2. Zdefiniuj metody, które umożliwią:
   a. Dodanie od określonego wiersza i linii jakiejś wartości z kontrolą zakresu (AddElem).
   b. Uzyskanie informacji o rozmiarze macierzy (`c` i `l`) .
   c. Dostęp do wartości w _matrix, ale bez możliwości ich modyfikacji w obiekcie (np. zwracając kopię).
   d. Metodę, która umożliwi dodatnie do siebie dwóch obiektów typu **Matrix**. (**AddMatrix**). Jej wynikiem jest nowa tablica. Dodawanie odbywa się wierszami, a wynikowa tablica ma rozmiar odpowiadający maksymalnej liczbie wierszy lub kolumn jednej z dodawanych **Tablic**.

8. Ćwiczenie 8 Elementy obiektowości – Klasa Matrix

Zaimplementuj klasę **Matrix**, przechowującą elementy int w strukturze `int[][]_matrix`, posiadającą taki sam konstruktor jak w poprzednim zadaniu.
Tak zdefiniuj poziom dostępności do pól klasy, aby nie było konieczności zwracania na zewnątrz tablicy zawierającej wartości zawarte w `_matrix`, ale były one dostępne dla wszystkich obiektów tej klasy i po niej dziedziczącej.
Dla dwóch macierzy wykonaj operacje dodawania i odejmowania. Wypisz ich wynik

9. Ćwiczenie 8 Elementy obiektowości – Klasa Book i BookLibrary.

1. Wszystkie książki, instancje klasy **Book** mają różną, przypisaną raz na początku i już nie zmienną wartość numeru ISBN (po inicjalizacji w konstruktorze nie można go już modyfikować).

```
class Book
{
   string _title;
   string _author;
   double _price;
   string _isbn;
   DateTime   date;
…
}
```

2. Instancja klasy **BookLibrary**. Jest zagwarantowane, że istnieje jedna tylko jedna i jest dostępna w całym programie. Jest też tworzona automatycznie prze CLR.
3. **BookLibrary** jest kontenerem zawierającym książki. Umożliwia dodawanie i usuwanie książek (Add, Remove). Nie można dodać książki o tym samym numerze ISBN. Można ją przeszukiwać po isbn, autorze, tytule lub cenie. Można również wylistować całą jej zawartość, a także sprawdzić czy dana książka w niej istnieje.
4. Utwórz przykładową bibliotekę i zaprezentuj jej działanie na kilku przykładach.