Ćwiczenie 1:

1. W funkcji ***Fun1*** nie możliwe jest zmodyfikowanie zmienną wejściową, ponieważ jest tylko do odczytu. Zmiana nastąpiła tylko dla ***Fun2*** i ***Fun3***.
2. Fun1(i\_1);  
   Fun2(out i\_2);  
   Fun3(ref i\_3);  
   Fun4(i\_4);
3. Przekazanie zmiennej typu ***short*** możliwe jest dla ***Fun1*** i ***Fun4***. Dla ***Fun2*** i ***Fun3*** nie można przekonwertować zmiennej.  
   Fun1(i);  
   Fun4(i);
4. 'Element „Program” nie może definiować przeciążonego elementu metoda, który różni się tylko modyfikatorami parametru „out” i „in”
6. static void Fun5(Point p){

Point p\_new = new Point(3, 3);  
p = p\_new;}

static void Fun6(ref Point p){

Point p\_new = new Point(3, 3);  
p = p\_new;}

1. 1, 1

1, 1

Po:

1, 1

3, 3

1. p1 = null;
2. Null jest nadpisany przez nowy obiekt.

Ćwiczenie 2:

1. Wskaźniki i bufory o ustalonym rozmiarze mogą zostać użyte tylko w kontekście słowa kluczowego „unsafe”.
2. Wartości zmiennych zostały zamienione.
3. int[] buffer = new int[1024];

fixed (int\* bufferPtr = buffer)

{

for (int i = 0; i < 1024; i++)

{

\*(bufferPtr + i) = i % 4;

}  
for (int i = 0; i < 10; i++)  
{

var val = \*(bufferPtr + i);  
Console.WriteLine($"Wartość [{i}]={val}");

}

}

Wartość [0]=0 Wartość [1]=1 Wartość [2]=2 Wartość [3]=3 Wartość [4]=0

Wartość [5]=1 Wartość [6]=2 Wartość [7]=3 Wartość [8]=0 Wartość [9]=1

Ćwiczenie 3:

static void cw3()

{

char[] table = new char[10];

var numElems = 0;

while (numElems < 10)

{

Console.WriteLine("\nZnak:");

char el = Console.ReadKey().KeyChar;

for (var i = numElems; i > 0; i--)

table[i] = table[i - 1];

table[0] = el;

numElems++;

}

Console.WriteLine("\nTablica:");

foreach (char e in table)

Console.Write($" {e}");

Console.WriteLine("\n");

Console.ReadLine();

}

Ćwiczenie 4:

static void cw4()

{

int[] table = new int[5];

for (var i = 0; i < table.Length; i++)

{

Console.WriteLine("\nLiczba:");

int el = int.Parse(Console.ReadLine());

table[i] = el;

}

Console.WriteLine("\nTablica:");

for (var i = 4;i >= 0; i--)

Console.Write($" {table[i]}");

Console.ReadLine() ;

}

Ćwiczenie 5:

static void cw5()

{

var array = new int[5];

for (var i = 0; i < array.Length; i++)

array[i] = int.Parse(Console.ReadLine());

var dict = new Dictionary<int, int>();

foreach (var value in array)

{

if (dict.ContainsKey(value))

{

dict[value]++;

}

else

{

dict[value] = 1;

}

}

foreach (var item in dict)

Console.WriteLine($"Val: {item.Key}, count: {item.Value}");

Console.ReadLine();

}

Ćwiczenie 6:

static void cw6()

{

int[,] tab\_1 = { { 1, 2, 3, 4, 5 },

{ 1, 2, 1, 1, 1},

{ 2, 2, 2, 2, 2},

{ 1, 1, 1, 1, 1},

{ 1, 1, 1, 1, 1}};

int[,] tab\_2 = { { 1, 2, 3, 4, 5 },

{ 1, 2, 1, 1, 1},

{ 2, 2, 2, 2, 2},

{ 1, 1, 1, 1, 1},

{ 1, 1, 1, 1, 1}};

int[,] tab\_3 = new int[5, 5];

for (int i = 0; i < tab\_1.GetLength(0); i++)

{

for (int j = 0; j < tab\_1.GetLength(1); j++)

{

tab\_3[i, j] = tab\_1[i, j] + tab\_2[i, j];

}

}

Console.WriteLine(tab\_3.Length);

Console.WriteLine(tab\_3.LongLength);

Console.WriteLine(tab\_3.Rank);

Console.ReadLine();

}

Ćwiczenie 7:

static void cw7()

{

int[,] tab = { { 1, 0, -1 },

{ 0, 0, 1 },

{ -1, -1, 0} };

int det = 0;

for (var i = 0; i < 3; i++)

{

det += (tab[0, i] \* (tab[1, (i + 1) % 3] \* tab[2, (i + 2) % 3]

- tab[1, (i + 2) % 3] \* tab[2, (i + 1) % 3]));

}

Console.WriteLine(det);

Console.ReadLine();

}

Ćwiczenie 8:

static void cw8()

{

var tab\_1 = new int[5][];

tab\_1[0] = new[] { 1, 2, 3 };

tab\_1[1] = new[] { 4, 5, 6, 7, 8, 9 };

tab\_1[2] = new[] { 10, 11, 12, 13 };

tab\_1[3] = new[] { 14, 15, 16, 17, 18 };

tab\_1[4] = new[] { 19, 20, 21 };

int[][] tab\_2 =

{

new[] { 1, 2, 3 },

new[] { 4, 5, 6, 7, 8, 9 },

new[] { 10, 11, 12, 13 },

new[] { 14, 15, 16, 17, 18 },

new[] { 19, 20, 21 }

};

foreach(var el in tab\_1)

{

Console.Write("{");

for(var j = 0; j < el.Length; j++)

{

Console.Write(j + " ");

}

Console.Write("}\n");

}

Console.ReadLine();

}