Detyra 1.

Write a program to simulate **n nested loops** from **1** to **n**.

Kodi:

static void Loops(int[] arr, int index)

{

if (index >= arr.Length)

{

foreach (int element in arr) Console.Write("{0}", element);

Console.WriteLine();

}

else

{

for(int i = 1; i <= arr.Length; i++)

{

arr[index] = i;

Loops(arr, index + 1);

}

}

}

static void Main(string[] args)

{

Console.Write("N: ");

int n = Int32.Parse(Console.ReadLine());

int[] arr = new int[n];

Loops(arr, 0);

}

Rezultati:

Detyra 2.

Write a program to generate **all variations with duplicates** of **n** elements class **k**. Sample input: n=3 k=2

Kodi:

static void GetCombinations(int[] arr, int index, int start, int end)

{

if (index >= arr.Length)

{

Console.Write("(");

for (int i = 0; i < arr.Length; i++)

if (i < arr.Length - 1) Console.Write("{0} ", arr[i]);

else Console.Write(arr[i]);

Console.Write("), ");

}

else

for (int i = start; i <= end; i++)

{

arr[index] = i;

GetCombinations(arr, index + 1, i, end);

}

}

static void Main(string[] args)

{

Console.Write("Enter N: ");

int n = Int32.Parse(Console.ReadLine());

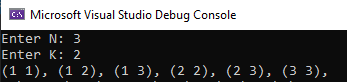
Console.Write("Enter K: ");

int k = Int32.Parse(Console.ReadLine());

int[] arr = new int[k];

GetCombinations(arr, 0, 1, n);

}

Rezultati:

Detyra 3.

Write a program to generate and print **all combinations with duplicates** of **k** elements from a set with **n** elements. Sample input:

Kodi:

static void GetCombinations(int[] arr, int index, int start, int end)

{

if (index >= arr.Length)

{

Console.Write("(");

for (int i = 0; i < arr.Length; i++)

if (i < arr.Length - 1) Console.Write("{0} ", arr[i]);

else Console.Write(arr[i]);

Console.Write("), ");

}

else

for (int i = start; i <= end; i++)

{

arr[index] = i;

GetCombinations(arr, index + 1, i, end);

}

}

static void Main(string[] args)

{

Console.Write("Enter N: ");

int n = Int32.Parse(Console.ReadLine());

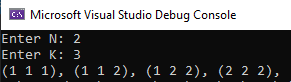
Console.Write("Enter K: ");

int k = Int32.Parse(Console.ReadLine());

int[] arr = new int[k];

GetCombinations(arr, 0, 1, n);

}

Rezultati:

Detyra 4.

You are given a **set of strings**. Write a **recursive program**, which **generates all subsets**, consisting exactly **k** strings chosen among the elements of this set. Sample input:

Kodi:

static string[] wordsArr;

static void FindSubsets(int[] arr, int index, int start, int end)

{

if (index >= arr.Length)

{

Console.Write("(");

for (int i = 0; i < arr.Length; i++)

{

Console.Write("{0}", wordsArr[arr[i]]);

if (i != arr.Length - 1) Console.Write(" ");

}

Console.Write("), ");

}

else

for (int i = start; i < end; i++)

{

arr[index] = i;

FindSubsets(arr, index + 1, i + 1, end);

}

}

static void Main(string[] args)

{

Console.Write("Enter array length: ");

int length = Int32.Parse(Console.ReadLine());

wordsArr = new string[length];

Console.WriteLine();

for (int i = 0; i < wordsArr.Length; i++)

{

Console.Write("Enter {0} word: ", i + 1);

wordsArr[i] = Console.ReadLine();

}

Console.Write("\nEnter K: ");

int k = Int32.Parse(Console.ReadLine());

int[] arr = new int[k];

Console.WriteLine();

FindSubsets(arr, 0, 0, length);

Console.ReadLine();

}

Rezultati: