**C# Program Részletes Leírása**

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

* **using System;**: A System névtér használatának engedélyezése, amely alapvető funkciókat (például a konzol bemenet/kimenetet) tartalmaz.
* **using System.Collections.Generic;**: A generikus gyűjtemények (pl. List) használatához szükséges.
* **using System.IO;**: Fájlkezelési funkciókhoz (pl. fájl beolvasás/írás) szükséges.
* **using System.Linq;**: LINQ (Language Integrated Query) funkciók használatát engedélyezi, amely lehetővé teszi a gyűjtemények könnyebb lekérdezését.

class Program

{

static void Main(string[] args)

{

* **class Program**: A Program osztály definiálása. Ez a fő osztály, ahol a program kezdődik.
* **static void Main(string[] args)**: A Main metódus a program belépési pontja. Itt kezdődik a program végrehajtása.

Console.WriteLine("Válasszon egy lehetőséget:");

Console.WriteLine("1. Véletlen számok generálása");

Console.WriteLine("2. Véletlen szöveg generálása");

Console.WriteLine("3. Számok ellenőrzése a fájlból");

Console.WriteLine("4. Szövegek ellenőrzése a fájlból");

* **Console.WriteLine(...)**: A konzolra írja a lehetőségeket, amelyeket a felhasználó választhat.

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

* **int choice = int.Parse(Console.ReadLine());**: Beolvassa a felhasználó választását és konvertálja egész számra.

switch (choice)

{

case 1:

GenerateRandomNumbers();

break;

case 2:

GenerateRandomStrings();

break;

case 3:

CheckNumbersInFile();

break;

case 4:

CheckStringsInFile();

break;

default:

Console.WriteLine("Érvénytelen választás.");

break;

}

* **switch (choice)**: A felhasználó választásának ellenőrzése.
* **case ...:**: A különböző lehetőségek kezelése. Minden esetben a megfelelő metódus hívódik meg, hogy végrehajtsa a feladatot.
* **default:**: Ha a választás nem érvényes, értesíti a felhasználót.

**Véletlen számok generálása**

static void GenerateRandomNumbers()

{

Console.Write("Darabszám: ");

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

Console.Write("Alsó határ: ");

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

Console.Write("Felső határ: ");

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

* **static void GenerateRandomNumbers()**: A véletlen számok generálásáért felelős metódus.
* **Console.Write(...)**: Kéri a felhasználót, hogy adja meg a darabszámot és a határokat.
* **int count = int.Parse(Console.ReadLine());**: Beolvassa a darabszámot.
* **int lowerBound = int.Parse(Console.ReadLine());**: Beolvassa az alsó határt.
* **int upperBound = int.Parse(Console.ReadLine());**: Beolvassa a felső határt.

Random rand = new Random();

List<int> numbers = new List<int>();

* **Random rand = new Random();**: Inicializál egy új véletlenszám-generátort.
* **List<int> numbers = new List<int>();**: Létrehoz egy új listát, amely tárolja a generált számokat.

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

{

numbers.Add(rand.Next(lowerBound, upperBound + 1));

}

* **for (int i = 0; i < count; i++)**: Ciklus, amely a megadott darabszámig iterál.
* **numbers.Add(rand.Next(lowerBound, upperBound + 1));**: Véletlen számot generál az alsó és a felső határ között, és hozzáadja a listához.

WriteToFile(numbers.Select(n => n.ToString()).ToList());

Console.WriteLine("Véletlen számok generálva és mentve a ki.txt fájlba.");

* **WriteToFile(...)**: A generált számokat karakterláncokká alakítja, és a fájlba írja őket.
* **Console.WriteLine(...)**: Értesíti a felhasználót a sikeres generálásról.

**Véletlen szöveg generálása**

static void GenerateRandomStrings()

{

Console.Write("Darabszám: ");

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

Console.Write("Kis- vagy nagybetűs szöveg? (k/n): ");

char caseChoice = Console.ReadKey().KeyChar;

Console.WriteLine();

* **static void GenerateRandomStrings()**: A véletlen szövegek generálásáért felelős metódus.
* **Console.ReadKey().KeyChar**: Beolvassa a felhasználó által megadott karaktert (kicsi vagy nagybetű).

Random rand = new Random();

List<string> strings = new List<string>();

* **Random rand = new Random();**: Inicializál egy új véletlenszám-generátort.
* **List<string> strings = new List<string>();**: Létrehoz egy új listát, amely tárolja a generált szövegeket.

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

{

int length = rand.Next(1, 21);

strings.Add(new string(Enumerable.Range(0, length)

.Select(\_ => caseChoice == 'k' ?

(char)rand.Next('a', 'z' + 1) :

(char)rand.Next('A', 'Z' + 1)).ToArray()));

}

* **for (int i = 0; i < count; i++)**: Ciklus, amely a megadott darabszámig iterál.
* **int length = rand.Next(1, 21);**: Véletlen hosszúságú szöveget generál (1 és 20 karakter között).
* **strings.Add(...)**: Létrehozza a véletlen szöveget, és hozzáadja a listához.

WriteToFile(strings);

Console.WriteLine("Véletlen szövegek generálva és mentve a ki.txt fájlba.");

* **WriteToFile(strings);**: A generált szövegeket a fájlba írja.
* **Console.WriteLine(...)**: Értesíti a felhasználót a sikeres generálásról.

**Számok ellenőrzése a fájlból**

static void CheckNumbersInFile()

{

Console.Write("Alsó határ: ");

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

Console.Write("Felső határ: ");

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

* **static void CheckNumbersInFile()**: A számok fájlból történő ellenőrzéséért felelős metódus.
* **Console.ReadLine()**: A felhasználótól kéri az alsó és a felső határt.

List<string> data = ReadFromFile();

if (data.All(item => int.TryParse(item, out \_)))

{

* **List<string> data = ReadFromFile();**: Beolvassa a fájl tartalmát.
* **data.All(...)**: Ellenőrzi, hogy az összes elem szám-e.

Console.WriteLine("A fájl számokat tartalmaz.");

bool valid = data.All(num => int.Parse(num) >= lowerBound && int.Parse(num) <= upperBound);

Console.WriteLine(valid ? "Az adatok érvényesek." : "Az adatok nem érvényesek.");

}

 **Console.WriteLine(...)**: Értesíti a felhasználót, hogy a fájl számokat tartalmaz.

 **bool valid = ...**: Ellenőrzi, hogy az összes szám a megadott határok között van-e.

 **Console.WriteLine(...)**: Kiírja, hogy az adatok érvényesek-e.

 **else**: Ha a fájl nem számokat tartalmaz, értesíti a felhasználót.

**Fájl olvasása**

static List<string> ReadFromFile(string filename = "ki.txt")

{

return File.ReadAllText(filename).Split(';').ToList();

}

* **static List<string> ReadFromFile(...)**: Fájl olvasását végző metódus.
* **File.ReadAllText(...)**: Beolvassa a fájl teljes tartalmát.
* **.Split(';')**: A beolvasott tartalmat pontosvessző alapján darabolja fel és listává alakítja.

**Rendezés és beillesztés**

A rendezés és beillesztés funkciók külön metódusba kerültek, hogy a kód tiszta maradjon, de ezek nincsenek megvalósítva a bemutatott kódban. Íme egy példa a rendezéshez:

static List<string> BubbleSort(List<string> arr, bool reverse)

{

int n = arr.Count;

for (int i = 0; i < n - 1; i++)

{

for (int j = 0; j < n - i - 1; j++)

{

if ((reverse && string.Compare(arr[j], arr[j + 1]) < 0) ||

(!reverse && string.Compare(arr[j], arr[j + 1]) > 0))

{

// Swap

var temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

return arr;

}

* **static List<string> BubbleSort(...)**: Buborékrendezést megvalósító metódus.
* **int n = arr.Count;**: A lista hosszának tárolása.
* **for (int i = 0; i < n - 1; i++)**: Külső ciklus a rendezéshez.
* **for (int j = 0; j < n - i - 1; j++)**: Belső ciklus az elemek összehasonlításához.
* **if (...)**: Ellenőrzi, hogy szükséges-e az elemek cseréje.
* **var temp = arr[j];**: Az elemek cseréjéhez ideiglenes változó.

<!DOCTYPE html>

<html lang="hu">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Egyszerű Weboldal</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>

    <header>

        <h1>Üdvözöljük a Weboldalunkon!</h1>

    </header>

    <main>

        <section>

            <h2>Rólunk</h2>

            <p>Ez egy példa weboldal HTML és CSS segítségével.</p>

        </section>

        <section>

            <h2>Kapcsolat</h2>

            <p>Elérhetőségeink: info@pelda.hu</p>

        </section>

    </main>

    <footer>

        <p>&copy; 2024 Példa Cég. Minden jog fenntartva.</p>

    </footer>

</body>

</html>

body {

    font-family: Arial, sans-serif;

    margin: 0;

    padding: 0;

    background-color: #f4f4f4;

}

header {

    background: #35424a;

    color: #ffffff;

    padding: 10px 20px;

    text-align: center;

}

main {

    padding: 20px;

}

section {

    background: #ffffff;

    border: 1px solid #dddddd;

    margin: 20px 0;

    padding: 15px;

    border-radius: 5px;

}

footer {

    text-align: center;

    padding: 10px 0;

    background: #35424a;

    color: #ffffff;

    position: relative;

    bottom: 0;

    width: 100%;

}

        <section>

            <h2>Hatékonyság és Tárigény</h2>

            <p><strong>Időbeli hatékonyság:</strong> A Bubble Sort algoritmus legrosszabb és átlagos időbeli hatékonysága O(n<sup>2</sup>), ahol n a lista elemszáma. A legjobb esetben, ha a lista már rendezett, az algoritmus O(n) időbeli hatékonysággal rendelkezik.</p>

            <p><strong>Tárigény:</strong> Az algoritmus helyigénye O(1), mivel csak egy néhány segédváltozóra van szükség.</p>

        </section>

<div class="main">

            <h2>Kód Példa</h2>

            <div class="row ">

                <div class="col-md-6">

                    <p>A konzolra írja a lehetőségeket, amelyeket a felhasználó választhat.</p>

                    <pre class="r">Console.WriteLine("Válasszon egy lehetőséget:")

                    Console.WriteLine("1. Véletlen számok generálása");

                    Console.WriteLine("2. Véletlen szöveg generálása");

                    Console.WriteLine("3. Számok ellenőrzése a fájlból");

                    Console.WriteLine("4. Szövegek ellenőrzése a fájlból");

                    </pre>

                </div>

                <div class="col-md-6">

                    <p>Beolvassa a felhasználó választását és konvertálja egész számra.</p>

                    <pre>int choice = int.Parse(Console.ReadLine());</pre>

                </div>

            </div>

            <div class="row">

                <div class="col-md-6">

                    <p>A felhasználó választásának ellenőrzése, majd a különböző lehetőségek kezelése. Minden esetben a megfelelő metódus hívódik meg, hogy végrehajtsa a feladatot. Ha a választás nem érvényes, értesíti a felhasználót.

                        </p>

                    <pre>switch (choice)

                        {

                            case 1:

                                GenerateRandomNumbers();

                                break;

                            case 2:

                                GenerateRandomStrings();

                                break;

                            case 3:

                                CheckNumbersInFile();

                                break;

                            case 4:

                                CheckStringsInFile();

                                break;

                            default:

                                Console.WriteLine("Érvénytelen választás.");

                                break;

                        }

                </pre>

                </div>

                <div class="col-md-6">

                    <p>A véletlen számokat generál az adott határokon belül. Először kéri a felhasználót, hogy adja meg a darabszámot és a határokat. Ez után beolvassa mindezt.

                        </p>

                    <pre>static void GenerateRandomNumbers()

                        {

                            Console.Write("Darabszám: ");

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

                            Console.Write("Alsó határ: ");

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

                            Console.Write("Felső határ: ");

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

                    </pre>

                </div>

            </div>

            <div class="row">

                <div class="col-md-6">

                    <p>Inicializál egy új véletlenszám-generátort, majd létrehoz egy új listát, amely tárolja a generált számokat. Ha megtörtént a számok generál az alsó és a felső határ között, és hozzáadja a listához.

                        </p>

                    <pre>

                        Random rand = new Random();

                        List< int > numbers = new List< int >();

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

                        {

                            numbers.Add(rand.Next(lowerBound, upperBound + 1));

                        }

                </pre>

                </div>

                <div class="col-md-6">

                    <p>A generált számokat karakterláncokká alakítja, és a fájlba írja őket, és értesíti a felhasználót a sikeres generálásról.

                        </p>

                    <pre>

WriteToFile(numbers.Select(n => n.ToString()).ToList());

Console.WriteLine("Véletlen számok generálva és mentve a ki.txt fájlba.");

                </pre>

                </div>

            </div>

            <div class="row">

                <div class="col-md-6">

                    <p>A számok fájlból történő ellenőrzéséért felelős metódus. A felhasználótól kéri az alsó és a felső határt. </p>

                    <pre>

                        static void CheckNumbersInFile()

                        {

                            Console.Write("Alsó határ: ");

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

                            Console.Write("Felső határ: ");

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

                    </pre>

                </div>

                <div class="col-md-6">

                    <p>Beolvassa a fájl tartalmát. Ellenőrzi, hogy az összes elem szám-e. Értesíti a felhasználót, hogy a fájl számokat tartalmaz.Ellenőrzi, hogy az összes szám a megadott határok között van-e.

                    Kiírja, hogy az adatok érvényesek-e. Ha a fájl nem számokat tartalmaz, értesíti a felhasználót.

                        </p>

                    <pre>

List< string > data = ReadFromFile();

if (data.All(item => int.TryParse(item, out \_)))

{

Console.WriteLine("A fájl számokat tartalmaz.");

bool valid = data.All(num => int.Parse(num) >= lowerBound && int.Parse(num) <= upperBound);

Console.WriteLine(valid ? "Az adatok érvényesek." : "Az adatok nem érvényesek.");

                        }

                </pre>

                </div>

            </div>

        </div>