Izveidot programmu C#:

Uzdevums: Programma, kas simulē failu sistēmu

Katram tā **objektam ir īpašības** :

* **nosaukums,**
* **vecaks (viens- kam pieder)**
* **berni (vairāki (masīvs) - kas viņam pieder)**
* **tips (mape/ datne)**

**metodes :**

* ***parvietoties()*** – iespēja **pārvietoties** pa objektiem, **kas ir mapes, datnēm viņa neko nedara**
* ***paradit()-*** mapēm rāda bērnus- **datnēm: /~[datne]~ , mapes: /[mape]**
* ***dzest()***- iespēja izdzēst mapi. Ja ir mape ar bērniem – jautā vai tiešām gribam dzēst, kuru apstiprina vai noliedz
* ***izveidot()*** – mapei iespēja pievienot bērnu mapi vai datni, datnei izveido īstu datni (pie dir >)

Programmu (*FailuSistemu*) palaižot jūs atrodaties tukšā diskā, kur sākums objekts ir “c:”

Uz ekrāna parādās PILNA atrašanās vieta un “>”. Tas notiek pēc katras darbības izņemot, kad ievada **exit** vai **by,** tad programma beidzas!

Piemēram c:/mana\_mape/test\_mape/>

Šīnī brīdī ir Console.Readline režīms, kas var saņemt komandas (neskatoties uz burtu lielumiem):

Info:

Mape vai vairākas mapes, kas atdalītas ar tukšumu (max 3), tas pats arī pa datni!

Mapes ietvaros nevar atkārtoties mapes un datņu nosaukumi

* **mkdir [mape`s]** – piesaista objektam jaunu mapi (bērnu`s), kas nevar atkārtoties,
* **create [datne`s]** – piesaista objektam jaunu datni (bērnu`s), kas nevar atkārtoties
* **rm [mape`s]** -dzēš mapi`es, pie izdzēšanas parādās, ka izdzēsts (destruktors)
* **dir** – parāda bērnus
* **dir >datne** – izvada bērnus reāla datnē+ izveido objektu datni, veco dzēšot! Ja nav datnes, to IZVEIDO!
* **cd [mape]** -pārvietojas uz mapi, ja ir mape , vai paziņojums
* **cd ..** -pārvietojas uz mapi, kas ir vecāks vai **cd /** uz c: (sākumu)
* **edit [datne]** – parāda datnes nosaukumu un īstās datnes saturu (ko izveidojis dir >)
* **del [datne`s]-** dzēš datnes un ja piesaistīti reālas datnes.
* **Ja kļūdas izķer ar Try..Catch – bonuss**
* **OBLIGĀTI komentāri, par programmas kodu darbību**
* **OBLIGĀTI avotu resursu adreses, kur smeltas idejas**
* **Ja plaģiāts – atzīme 1 !**

Bez adresēm, komentāriem un kļūdu izķeršanām max atzīme 5.

AR adresēm, komentāriem, bet bez kļūdu izķeršanām max atzīme 8

Mierinājumam!

Kodam nav jābūt perfektam, bet jums jāspēj savu kodu aizstāvēt!!!

es pajautāš, kar kādu kodu daļu klātienē!

using System;

using System.Collections.Generic;

using System.IO;

// Abstrakta klase priekš failu sistēmas objektiem (mapes un datnes)

abstract class FileSystemObject

{

public string Name { get; set; } // Objekta nosaukums

public Directory Parent { get; set; } // Atsauce uz vecāku mapi

public string Type { get; set; } // Objekta tips: "mape" vai "datne"

// Konstruktors, kas iestata īpašības

public FileSystemObject(string name, Directory parent, string type)

{

Name = name;

Parent = parent;

Type = type;

}

// Atgriež pilno ceļu līdz objektam (rekursīvi)

public string GetPath()

{

return Parent == null ? Name : Parent.GetPath() + "/" + Name;

}

}

// Datnes klase, kas pārstāv failu (ar tipu "datne")

class File : FileSystemObject

{

public File(string name, Directory parent) : base(name, parent, "datne") { }

}

// Mapes klase, kas var saturēt citus objektus (bērnus)

class Directory : FileSystemObject

{

// Saglabā bērnus vārdnīcā (pēc vārda)

public Dictionary<string, FileSystemObject> Children { get; set; }

public Directory(string name, Directory parent) : base(name, parent, "mape")

{

Children = new Dictionary<string, FileSystemObject>();

}

// Pievieno jaunu bērnu objektu mapei

public void AddChild(FileSystemObject obj)

{

if (Children.ContainsKey(obj.Name))

throw new Exception($"Objekts ar nosaukumu '{obj.Name}' jau eksistē.");

Children[obj.Name] = obj;

}

// Noņem bērnu pēc nosaukuma

public void RemoveChild(string name)

{

if (!Children.ContainsKey(name))

throw new Exception($"Nav atrasts: {name}");

Children.Remove(name);

}

// Izvada visu bērnu sarakstu terminālī

public void ListChildren()

{

foreach (var child in Children.Values)

{

// Formatēts attēlojums: mape - /[nosaukums], datne - /~[nosaukums]

Console.WriteLine(child.Type == "mape" ? $"/[{child.Name}]" : $"/~[{child.Name}]");

}

}

// Atgriež bērnu pēc nosaukuma, ja tāds eksistē

public FileSystemObject GetChild(string name)

{

return Children.ContainsKey(name) ? Children[name] : null;

}

}

// Galvenā programma

class Program

{

static Directory root = new Directory("c:", null); // Saknes mape

static Directory current = root; // Aktuālā (darba) mape

static void Main()

{

Console.WriteLine("Failu Sistēma sākta. Lieto 'exit' lai izietu.");

while (true)

{

// Rāda pilno ceļu un gaida ievadi

Console.Write(current.GetPath() + "> ");

var input = Console.ReadLine().Trim();

if (string.IsNullOrEmpty(input)) continue;

if (input.ToLower() == "exit" || input.ToLower() == "by") break;

var parts = input.Split(' ', 2);

var command = parts[0].ToLower(); // Komanda

var args = parts.Length > 1 ? parts[1].Split(' ', StringSplitOptions.RemoveEmptyEntries) : new string[0]; // Argumenti

try

{

switch (command)

{

// Izveido jaunu mapi

case "mkdir":

foreach (var name in args) current.AddChild(new Directory(name, current));

break;

// Izveido jaunu datni (programmas objektu)

case "create":

foreach (var name in args) current.AddChild(new File(name, current));

break;

// Dzēš mapi (ar apstiprinājumu, ja ir bērni)

case "rm":

foreach (var name in args)

{

var child = current.GetChild(name);

if (child is Directory dir)

{

if (dir.Children.Count > 0)

{

Console.Write($"Mapei '{name}' ir bērni. Dzēst? (j/n): ");

if (Console.ReadLine().ToLower() != "j") continue;

}

current.RemoveChild(name);

Console.WriteLine($"Mape '{name}' izdzēsta.");

}

}

break;

// Dzēš datni (gan objektu, gan īsto .txt failu)

case "del":

foreach (var name in args)

{

var child = current.GetChild(name);

if (child is File)

{

current.RemoveChild(name);

var filePath = $"{name}.txt";

if (System.IO.File.Exists(filePath)) System.IO.File.Delete(filePath);

Console.WriteLine($"Datne '{name}' izdzēsta.");

}

}

break;

// Maina aktuālo mapi

case "cd":

if (args[0] == ".." && current.Parent != null)

current = current.Parent; // Uz vecāku

else if (args[0] == "/")

current = root; // Uz sakni

else

{

var next = current.GetChild(args[0]) as Directory;

if (next != null) current = next;

else Console.WriteLine("Mape nav atrasta.");

}

break;

// Parāda bērnus vai saglabā tos failā

case "dir":

if (args.Length > 0 && args[0].StartsWith(">"))

{

var fileName = args[0].Substring(1);

var filePath = fileName + ".txt";

// Saglabā bērnu nosaukumus reālā failā

using (StreamWriter sw = new StreamWriter(filePath))

{

foreach (var child in current.Children.Values)

{

sw.WriteLine(child.Name);

}

}

// Ja objekts ar šo vārdu jau eksistē, izdzēš

if (current.GetChild(fileName) is File) current.RemoveChild(fileName);

// Pievieno kā datnes objektu sistēmā

current.AddChild(new File(fileName, current));

Console.WriteLine($"Bērni saglabāti datnē '{filePath}'.");

}

else current.ListChildren(); // Vienkārši rāda bērnus

break;

// Parāda datnes saturu (.txt fails)

case "edit":

var fileToEdit = args[0];

var editPath = fileToEdit + ".txt";

if (System.IO.File.Exists(editPath))

{

Console.WriteLine($"Saturs no '{editPath}':");

Console.WriteLine(System.IO.File.ReadAllText(editPath));

}

else Console.WriteLine("Datne nav atrasta.");

break;

// Ja komanda nav atpazīta

default:

Console.WriteLine("Nezināma komanda.");

break;

}

}

catch (Exception ex)

{

// Parāda kļūdas paziņojumu

Console.WriteLine("Kļūda: " + ex.Message);

}

}

Console.WriteLine("Programma izbeigta.");

}

}

<https://www.youtube.com/watch?v=9mUuJIKq40M&t=910s> <https://www.userfilesystem.com/programming/creating_virtual_file_system/> <https://www.geeksforgeeks.org/c-sharp-inheritance/>

<https://www.w3schools.com/cs/index.php>

<https://www.w3schools.com/cs/cs_methods.php>

<https://www.w3schools.com/cs/cs_conditions.php>

<https://www.w3schools.com/cs/cs_files.php>

<https://www.w3schools.com/cs/cs_comments.php>

<https://www.w3schools.com/cs/cs_exceptions.php>

<https://www.w3schools.com/cs/cs_variables.php>

<https://www.w3schools.com/cs/cs_data_types.php>

<https://www.geeksforgeeks.org/c-sharp-dictionary-with-examples/>

<https://www.w3schools.blog/c-sharp-dictionary>

<https://www.w3schools.com/cs/cs_properties.php>

pamats ņemts ar gpt, ari gpt tutorials lietojot

kods strada ari visual studio code un veido realus mapes