using System;

using System.Collections.Generic;

using System.IO;

//mkdir [mape] - izveido jaunu mapi

//create [datne] - izveido jaunu datni

//rm [mape] - dzēš mapi

//dir - parāda mapes saturu

//dir > datne - saglabā saturu reālā datnē

//cd [mape] - pārvietojas uz mapi

//cd .. - atgriežas vecākmapē

//edit [datne] - rediģē datni

//del [datne] - dzēš datni

//exit/by - iziet no programmas

// Failu sistēmas objekta klase

public class FailuSistObj

{

public string Nosaukums { get; set; }

public FailuSistObj Vecaks { get; set; }

public List<FailuSistObj> Berni { get; set; }

public string Tips { get; set; } // "mape" vai "datne"

public string FiziskaDatne { get; set; } // reālās datnes ceļš

// Konstruktors

public FailuSistObj(string nosaukums, string tips, FailuSistObj vecaks = null)

{

Nosaukums = nosaukums;

Tips = tips;

Vecaks = vecaks;

Berni = new List<FailuSistObj>();

FiziskaDatne = null;

}

// Pārvietošanās metode

public FailuSistObj Parvietoties(string mape)

{

if (Tips != "mape") return null;

if (mape == "..")

{

return Vecaks ?? this; // Ja nav vecāka, paliekam te

}

else if (mape == "/")

{

// Atgriežamies uz sakni

FailuSistObj current = this;

while (current.Vecaks != null)

{

current = current.Vecaks;

}

return current;

}

else

{

// Meklējam bērnu mapi

foreach (var bern in Berni)

{

if (bern.Tips == "mape" && bern.Nosaukums.Equals(mape, StringComparison.OrdinalIgnoreCase))

{

return bern;

}

}

return null; // Mape nav atrasta

}

}

// Parādīt saturu

public void Paradit()

{

if (Tips == "mape")

{

Console.WriteLine($"/[{Nosaukums}]");

foreach (var bern in Berni)

{

Console.WriteLine($"- {bern.Nosaukums} ({bern.Tips})");

}

}

else

{

Console.WriteLine($"/[{Nosaukums}]");

}

}

// Dzēšanas metode

public bool Dzest()

{

if (Tips == "mape" && Berni.Count > 0)

{

Console.Write($"Vai tiešām vēlaties dzēst mapi '{Nosaukums}' un visus tās saturu? (y/n): ");

string atbilde = Console.ReadLine().ToLower();

if (atbilde != "y") return false;

}

// Dzēšam reālo datni, ja tāda ir

if (!string.IsNullOrEmpty(FiziskaDatne) && File.Exists(FiziskaDatne))

{

try

{

File.Delete(FiziskaDatne);

}

catch (Exception ex)

{

Console.WriteLine($"Kļūda dzēšot datni: {ex.Message}");

return false;

}

}

// Noņemam sevi no vecāka bērnu saraksta

if (Vecaks != null)

{

Vecaks.Berni.Remove(this);

}

return true;

}

// Izveides metode

public FailuSistObj Izveidot(string nosaukums, string tips)

{

if (Tips != "mape") return null;

// Pārbaudam, vai nosaukums jau eksistē

foreach (var bern in Berni)

{

if (bern.Nosaukums.Equals(nosaukums, StringComparison.OrdinalIgnoreCase))

{

return null;

}

}

var jaunsObjekts = new FailuSistObj(nosaukums, tips, this);

Berni.Add(jaunsObjekts);

return jaunsObjekts;

}

}

// Galvenā programmas klase

public class Program

{

private static FailuSistObj currentDir;

private static FailuSistObj sakne;

// Parādām pašreizējo atrašanās vietu

private static void ParadiAtrasanasVietu()

{

var path = new List<string>();

var temp = currentDir;

while (temp != null)

{

path.Insert(0, temp.Nosaukums);

temp = temp.Vecaks;

}

Console.Write(string.Join("/", path) + "> ");

}

// Inicializējam failu sistēmu

private static void Inicializet()

{

sakne = new FailuSistObj("c:", "mape");

currentDir = sakne;

}

// Apstrādājam komandu

private static void ApstradaKomandu(string komanda)

{

var parts = komanda.Split(new[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

if (parts.Length == 0) return;

string cmd = parts[0].ToLower();

string[] args = parts.Length > 1 ? parts[1..] : Array.Empty<string>();

try

{

switch (cmd)

{

case "mkdir":

foreach (var mape in args)

{

if (currentDir.Izveidot(mape, "mape") == null)

{

Console.WriteLine($"Kļūda: Mape '{mape}' jau eksistē vai nevar izveidot!");

}

}

break;

case "create":

foreach (var datne in args)

{

if (currentDir.Izveidot(datne, "datne") == null)

{

Console.WriteLine($"Kļūda: Datne '{datne}' jau eksistē vai nevar izveidot!");

}

}

break;

case "rm":

foreach (var mape in args)

{

var obj = currentDir.Berni.Find(b => b.Nosaukums.Equals(mape, StringComparison.OrdinalIgnoreCase) && b.Tips == "mape");

if (obj != null)

{

if (obj.Dzest())

{

Console.WriteLine($"Mape '{mape}' veiksmīgi dzēsta.");

}

}

else

{

Console.WriteLine($"Kļūda: Mape '{mape}' nav atrasta!");

}

}

break;

case "dir":

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

{

if (args.Length < 2)

{

Console.WriteLine("Kļūda: Trūkst datnes nosaukuma!");

break;

}

string datnesNosaukums = args[1];

var datne = currentDir.Berni.Find(b => b.Nosaukums.Equals(datnesNosaukums, StringComparison.OrdinalIgnoreCase) && b.Tips == "datne");

// Ja datne neeksistē, izveidojam to

if (datne == null)

{

datne = currentDir.Izveidot(datnesNosaukums, "datne");

if (datne == null)

{

Console.WriteLine($"Kļūda: Nevar izveidot datni '{datnesNosaukums}'!");

break;

}

}

// Izveidojam reālu datni

try

{

string tempPath = Path.GetTempPath();

string fullPath = Path.Combine(tempPath, datnesNosaukums + ".txt");

using (StreamWriter sw = new StreamWriter(fullPath))

{

foreach (var bern in currentDir.Berni)

{

sw.WriteLine($"- {bern.Nosaukums} ({bern.Tips})");

}

}

datne.FiziskaDatne = fullPath;

Console.WriteLine($"Satura izvade saglabāta datnē: {fullPath}");

}

catch (Exception ex)

{

Console.WriteLine($"Kļūda saglabājot datni: {ex.Message}");

}

}

else

{

currentDir.Paradit();

}

break;

case "cd":

if (args.Length == 0)

{

Console.WriteLine("Kļūda: Trūkst argumenta!");

break;

}

if (args[0] == ".." || args[0] == "/")

{

var newDir = currentDir.Parvietoties(args[0]);

if (newDir != null)

{

currentDir = newDir;

}

}

else

{

var newDir = currentDir.Parvietoties(args[0]);

if (newDir != null)

{

currentDir = newDir;

}

else

{

Console.WriteLine($"Kļūda: Mape '{args[0]}' nav atrasta!");

}

}

break;

case "edit":

if (args.Length == 0)

{

Console.WriteLine("Kļūda: Trūkst datnes nosaukuma!");

break;

}

var editDatne = currentDir.Berni.Find(b => b.Nosaukums.Equals(args[0], StringComparison.OrdinalIgnoreCase) && b.Tips == "datne");

if (editDatne != null)

{

if (!string.IsNullOrEmpty(editDatne.FiziskaDatne) && File.Exists(editDatne.FiziskaDatne))

{

Console.WriteLine($"Rediģējam datni: {editDatne.Nosaukums}");

Console.WriteLine("Saturs:");

Console.WriteLine(File.ReadAllText(editDatne.FiziskaDatne));

}

else

{

Console.WriteLine($"Datne '{editDatne.Nosaukums}' ir tukša.");

}

}

else

{

Console.WriteLine($"Kļūda: Datne '{args[0]}' nav atrasta!");

}

break;

case "del":

foreach (var datne in args)

{

var obj = currentDir.Berni.Find(b => b.Nosaukums.Equals(datne, StringComparison.OrdinalIgnoreCase) && b.Tips == "datne");

if (obj != null)

{

if (obj.Dzest())

{

Console.WriteLine($"Datne '{datne}' veiksmīgi dzēsta.");

}

}

else

{

Console.WriteLine($"Kļūda: Datne '{datne}' nav atrasta!");

}

}

break;

case "exit":

case "by":

Environment.Exit(0);

break;

default:

Console.WriteLine($"Nezināma komanda: {cmd}");

break;

}

}

catch (Exception ex)

{

Console.WriteLine($"Kļūda: {ex.Message}");

}

}

public static void Main(string[] args)

{

Inicializet();

Console.WriteLine("Failu sistēmas simulators. Lai izietu, ievadiet 'exit' vai 'by'.");

while (true)

{

ParadiAtrasanasVietu();

string komanda = Console.ReadLine();

ApstradaKomandu(komanda);

}

}

}