using System;

using System.Collections.Generic;

using System.IO;

namespace FailuSistema

{

// Bāzes klase failu sistēmas objektiem

public abstract class FSObject

{

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

public FSObject Parent { get; set; } // Vecākelements (mape)

public List<FSObject> Children { get; set; } // Apakšelementi

public string Type { get; protected set; } // Objekta tips (mape vai datne)

public FSObject(string name, FSObject parent)

{

Name = name;

Parent = parent;

Children = new List<FSObject>();

}

// Abstraktas metodes, kuras realizē atvasinātās klases

public abstract void Parvietoties(); // Pārvietošanās

public abstract void Paradit(); // Satura attēlošana

public abstract void Dzest(); // Dzēšana

public abstract void Izveidot(string name, string type); // Izveide

// Metode, kas atgriež pilnu ceļu

public string GetFullPath()

{

if (Parent == null)

return Name;

else

return Parent.GetFullPath() + "/" + Name;

}

// Destruktors

~FSObject()

{

Console.WriteLine($"Izdzēsts: {Name}");

}

}

// Klase mapēm

public class Folder : FSObject

{

public Folder(string name, FSObject parent) : base(name, parent)

{

Type = "mape";

}

// Pārvietošanās uz šo mapi

public override void Parvietoties()

{

// Realizēta Program klasē, jo nepieciešams mainīt aktuālo atrašanās vietu

}

// Mapes satura attēlošana

public override void Paradit()

{

if (Children.Count == 0)

{

Console.WriteLine("Mape ir tukša.");

return;

}

foreach (var child in Children)

{

if (child.Type == "mape")

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

else

Console.WriteLine($"/~[{child.Name}]~");

}

}

// Mapes dzēšana

public override void Dzest()

{

if (Children.Count > 0)

{

Console.Write($"Vai tiešām vēlaties dzēst mapi '{Name}' ar tās saturu? (j/n): ");

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

if (answer != "j" && answer != "jā")

return;

}

if (Parent != null)

{

Parent.Children.Remove(this);

}

}

// Bērna objekta (mapes vai datnes) izveide

public override void Izveidot(string name, string type)

{

foreach (var child in Children)

{

if (child.Name == name)

{

Console.WriteLine($"Objekts ar nosaukumu '{name}' jau eksistē.");

return;

}

}

if (type == "mape")

{

Children.Add(new Folder(name, this));

}

else if (type == "datne")

{

Children.Add(new File(name, this));

}

}

}

// Klase datnēm

public class File : FSObject

{

private string PhysicalFilePath; // Fiziskais ceļš uz datni

public File(string name, FSObject parent) : base(name, parent)

{

Type = "datne";

PhysicalFilePath = AppDomain.CurrentDomain.BaseDirectory + name;

}

// Pārvietošanās nav iespējama uz datnēm

public override void Parvietoties()

{

Console.WriteLine("Nevar pārvietoties uz datni.");

}

// Datnes attēlošana

public override void Paradit()

{

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

}

// Datnes dzēšana

public override void Dzest()

{

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

{

try

{

System.IO.File.Delete(PhysicalFilePath);

}

catch (Exception ex)

{

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

}

}

if (Parent != null)

{

Parent.Children.Remove(this);

}

}

// Datne nevar saturēt citus objektus

public override void Izveidot(string name, string type)

{

Console.WriteLine("Datnes nevar izveidot citas datnes vai mapes.");

}

// Raksta saturu fiziskajā datnē

public void WriteContent(string content)

{

try

{

System.IO.File.WriteAllText(PhysicalFilePath, content);

}

catch (Exception ex)

{

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

}

}

// Nolasa saturu no fiziskās datnes

public string ReadContent()

{

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

{

try

{

return System.IO.File.ReadAllText(PhysicalFilePath);

}

catch (Exception ex)

{

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

}

}

return "Datne nav atrasta vai tai nav satura.";

}

}

// Galvenā programma

class Program

{

static FSObject currentLocation; // Aktuālā atrašanās vieta failu sistēmā

static Folder rootDisk; // Saknes mape

static void Main(string[] args)

{

// Izveido saknes mapi

rootDisk = new Folder("c:", null);

currentLocation = rootDisk;

bool running = true;

while (running)

{

Console.Write($"{currentLocation.GetFullPath()}/> "); // Parāda ceļu

string input = Console.ReadLine().Trim(); // Lietotāja komanda

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

{

running = false;

continue;

}

try

{

ProcessCommand(input); // Apstrādā komandu

}

catch (Exception ex)

{

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

}

}

}

// Komandas apstrādes metode

static void ProcessCommand(string input)

{

string[] parts = input.Split(' ', 2);

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

string arguments = parts.Length > 1 ? parts[1] : "";

switch (command)

{

case "mkdir":

HandleMkdir(arguments); // Izveido mapi

break;

case "create":

HandleCreate(arguments); // Izveido datni

break;

case "rm":

HandleRm(arguments); // Dzēš mapes

break;

case "dir":

HandleDir(arguments); // Parāda vai saglabā saturu

break;

case "cd":

HandleCd(arguments); // Maina mapi

break;

case "edit":

HandleEdit(arguments); // Parāda datnes saturu

break;

case "del":

HandleDel(arguments); // Dzēš datnes

break;

default:

Console.WriteLine("Nezināma komanda. Mēģiniet vēlreiz.");

break;

}

}

// Apstrādā mkdir komandu - mapju izveide

static void HandleMkdir(string arguments)

{

if (currentLocation.Type != "mape")

{

Console.WriteLine("Tikai mapēs var izveidot jaunas mapes.");

return;

}

string[] folderNames = arguments.Split(' ', 4);

if (folderNames.Length > 3)

{

Console.WriteLine("Var izveidot maksimāli 3 mapes vienlaicīgi.");

return;

}

foreach (string name in folderNames)

{

if (!string.IsNullOrWhiteSpace(name))

{

currentLocation.Izveidot(name, "mape");

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

}

}

}

// Apstrādā create komandu - datņu izveide

static void HandleCreate(string arguments)

{

if (currentLocation.Type != "mape")

{

Console.WriteLine("Tikai mapēs var izveidot jaunas datnes.");

return;

}

string[] fileNames = arguments.Split(' ', 4);

if (fileNames.Length > 3)

{

Console.WriteLine("Var izveidot maksimāli 3 datnes vienlaicīgi.");

return;

}

foreach (string name in fileNames)

{

if (!string.IsNullOrWhiteSpace(name))

{

currentLocation.Izveidot(name, "datne");

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

}

}

}

// Apstrādā rm komandu - mapju dzēšana

static void HandleRm(string arguments)

{

string[] folderNames = arguments.Split(' ', 4);

if (folderNames.Length > 3)

{

Console.WriteLine("Var dzēst maksimāli 3 mapes vienlaicīgi.");

return;

}

foreach (string name in folderNames)

{

if (!string.IsNullOrWhiteSpace(name))

{

var folderToDelete = currentLocation.Children.Find(c => c.Name == name && c.Type == "mape");

if (folderToDelete != null)

{

folderToDelete.Dzest();

}

else

{

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

}

}

}

}

// Apstrādā dir komandu - parāda vai saglabā mapes saturu

static void HandleDir(string arguments)

{

if (arguments.StartsWith(">"))

{

string fileName = arguments.Substring(1).Trim();

if (string.IsNullOrWhiteSpace(fileName))

{

Console.WriteLine("Jānorāda datnes nosaukums.");

return;

}

var targetFile = currentLocation.Children.Find(c => c.Name == fileName && c.Type == "datne") as File;

if (targetFile == null)

{

currentLocation.Izveidot(fileName, "datne");

targetFile = currentLocation.Children.Find(c => c.Name == fileName && c.Type == "datne") as File;

}

string content = $"Mapes '{currentLocation.Name}' saturs:\n";

foreach (var child in currentLocation.Children)

{

if (child.Type == "mape")

content += $"/[{child.Name}]\n";

else

content += $"/~[{child.Name}]~\n";

}

targetFile.WriteContent(content);

Console.WriteLine($"Mapes saturs izvadīts datnē '{fileName}'.");

}

else

{

currentLocation.Paradit();

}

}

// Apstrādā cd komandu - maina aktuālo mapi

static void HandleCd(string arguments)

{

if (arguments == "..")

{

if (currentLocation.Parent != null)

{

currentLocation = currentLocation.Parent;

}

return;

}

if (arguments == "/")

{

currentLocation = rootDisk;

return;

}

var targetFolder = currentLocation.Children.Find(c => c.Name == arguments && c.Type == "mape");

if (targetFolder != null)

{

currentLocation = targetFolder;

}

else

{

Console.WriteLine($"Mape '{arguments}' nav atrasta.");

}

}

// Apstrādā edit komandu - parāda datnes saturu

static void HandleEdit(string arguments)

{

if (string.IsNullOrWhiteSpace(arguments))

{

Console.WriteLine("Jānorāda datnes nosaukums.");

return;

}

var targetFile = currentLocation.Children.Find(c => c.Name == arguments && c.Type == "datne") as File;

if (targetFile != null)

{

Console.WriteLine($"Datnes '{targetFile.Name}' saturs:");

Console.WriteLine(targetFile.ReadContent());

}

else

{

Console.WriteLine($"Datne '{arguments}' nav atrasta.");

}

}

// Apstrādā del komandu - dzēš datnes

static void HandleDel(string arguments)

{

string[] fileNames = arguments.Split(' ', 4);

if (fileNames.Length > 3)

{

Console.WriteLine("Var dzēst maksimāli 3 datnes vienlaicīgi.");

return;

}

foreach (string name in fileNames)

{

if (!string.IsNullOrWhiteSpace(name))

{

var fileToDelete = currentLocation.Children.Find(c => c.Name == name && c.Type == "datne");

if (fileToDelete != null)

{

fileToDelete.Dzest();

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

}

else

{

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

}

}

}

}

}

}