```
using System;
using System.Collections.Generic;
using System.Linq;

enum Veids { Mape, Datne }

class Objekts
{
   public string Vards { get; set; }
   public Veids Tips { get; set; }
   public Objekts Augstaks { get; set; }
   public List<Objekts> Ieksejie { get; set; } = new List<Objekts>();

   public Objekts(string vards, Veids tips, Objekts augstaks = null)
   {
      Vards = vards;
      Tips = tips;
      Augstaks = augstaks;
   }

   public void Paradi()
   {
      if (Tips == Veids.Datne)
      {
         Console.WriteLine("/~" + Vards + "~");
         return;
      }
```

```
    foreach (var vien in Ieksejie)
    {
      if (vien.Tips == Veids.Mape)
        Console.WriteLine("/[" + vien.Vards + "]");
      else
        Console.WriteLine("/~" + vien.Vards + "~");
    }
  }

  public void Pievieno(string nosaukums, Veids veids)
  {
    if (Ieksejie.Any(el => el.Vards.Equals(nosaukums,
StringComparison.OrdinalIgnoreCase)))
    {
      Console.WriteLine("Jau eksistē šāds objekts.");
      return;
    }

    Ieksejie.Add(new Objekts(nosaukums, veids, this));
  }

  public void DzestObjektu(string nosaukums)
  {
    var atrastais = Ieksejie.FirstOrDefault(x => x.Vards.Equals(nosaukums,
StringComparison.OrdinalIgnoreCase));
    if (atrastais == null)
    {
      Console.WriteLine("Objekts nav atrasts.");
```

```
            return;
        }

        if (atrastais.Tips == Veids.Mape && atrastais.Ieksejie.Any())
        {
            Console.Write($"Mape '{nosaukums}' nav tukša. Vai dzēst? (j/n): ");
            if (Console.ReadLine().ToLower() != "j")
            {
                Console.WriteLine("Dzēšana atcelta.");
                return;
            }
        }

        Ieksejie.Remove(atrastais);
        Console.WriteLine($"Objekts '{nosaukums}' ir izdzēsts.");
    }

    public Objekts AtrodiMapi(string nosaukums)
    {
        return Ieksejie.FirstOrDefault(x => x.Vards.Equals(nosaukums,
StringComparison.OrdinalIgnoreCase) && x.Tips == Veids.Mape);
    }

    public void AtvertDatni(string datnesVards)
    {
        var datne = Ieksejie.FirstOrDefault(x => x.Vards.Equals(datnesVards,
StringComparison.OrdinalIgnoreCase) && x.Tips == Veids.Datne);
        if (datne != null)
```

```
        {
            Console.WriteLine($"Datne: /~{datne.Vards}~");
        }
        else
        {
            Console.WriteLine("Datne nav atrasta.");
        }
    }

    public string Cels()
    {
        return Augstaks == null ? $"{Vards}:/" : Augstaks.Cels() + Vards + "/";
    }
}

class Program
{
    static void Main()
    {
        var sakne = new Objekts("c", Veids.Mape);
        var aktivs = sakne;

        while (true)
        {
            Console.Write(aktivs.Cels() + "> ");
            var ievade = Console.ReadLine()?.Trim();
            if (string.IsNullOrWhiteSpace(ievade)) continue;
```

```
var daļas = ievade.Split(' ');
var komanda = daļas[0].ToLower();
var args = daļas.Length > 1 ? daļas[1] : "";

try
{
   switch (komanda)
   {
      case "mkdir":
         if (!string.IsNullOrEmpty(args))
            aktivs.Pievieno(args, Veids.Mape);
         break;

      case "create":
         if (!string.IsNullOrEmpty(args))
            aktivs.Pievieno(args, Veids.Datne);
         break;

      case "rm":
         if (!string.IsNullOrEmpty(args))
            aktivs.DzestObjektu(args);
         break;

      case "dir":
         aktivs.Paradi();
         break;

      case "cd":
```

```
if (args == "..")
{
   if (aktivs.Augstaks != null)
      aktivs = aktivs.Augstaks;
}
else
{
   var nakamais = aktivs.AtrodiMapi(args);
   if (nakamais != null)
      aktivs = nakamais;
   else
```