Ανάλυση Προγράμματος Διαχείρισης Δεδομένων με JSON (C#)

Το πρόγραμμα αποτελεί μια εφαρμογή κονσόλας που διαχειρίζεται δεδομένα μαθητών, αποθηκευμένα σε αρχείο JSON. Παρέχει βασικές λειτουργίες όπως εμφάνιση, προσθήκη, αναζήτηση, διαγραφή και ταξινόμηση.

◆ 1. Ανάγνωση δεδομένων από JSON

```
using System.Text.Json;

string file = "students.json";

var json = File.ReadAllText(file);
var students = JsonSerializer.Deserialize<List<Student>>(json) ??
new List<Student>();
```

Το πρόγραμμα:

- διαβάζει το αρχείο `students.json`
- μετατρέπει τα δεδομένα σε λίστα αντικειμένων (`List<Student>`)

◆ 2. Κεντρικός βρόχος (Menu)

```
while (true)
```

Ο βρόχος εκτελείται συνεχώς, επιτρέποντας την αλληλεπίδραση με τον χρήστη μέσω μενού.

```
Console.WriteLine("1 - Show students");
...
Console.WriteLine("6 - Exit");
```

Ο χρήστης επιλέγει λειτουργία:

```
string choice = Console.ReadLine();
```

◆ 3. Επιλογή λειτουργίας

```
if (choice == "1")
    ShowStudents(students);
```

Ανάλογα με την επιλογή, καλείται η αντίστοιχη μέθοδος.

◆ 4. Εμφάνιση δεδομένων (Show)

```
foreach (var s in students)
{
    Console.WriteLine($"{s.name} - {s.age}");
}
```

Εμφανίζει όλους τους μαθητές της λίστας.

◆ 5. Προσθήκη δεδομένων (Add)

```
string name = Console.ReadLine();
int age = int.Parse(Console.ReadLine());
```

Λαμβάνει δεδομένα από τον χρήστη.
```
students.Add(new Student { name = name, age = age });
```

Προσθέτει νέο αντικείμενο στη λίστα.
```
Save(students);
```

Αποθηκεύει τις αλλαγές στο JSON.

◆ 6. Αναζήτηση (Search)
```
var results = students.Where(s =>
s.name.ToLower().Contains(search.ToLower()));
```

Φιλτράρει τη λίστα με βάση το όνομα.
```
foreach (var s in results)
{
    Console.WriteLine($"{s.name} - {s.age}");
}
```

Εμφανίζει τα αποτελέσματα.

◆ 7. Διαγραφή (Delete)
```
var student = students.FirstOrDefault(...);
```

Βρίσκει έναν μαθητή.
```
students.Remove(student);
```

Τον αφαιρεί από τη λίστα.
```
Save(students);
```

Ενημερώνει το αρχείο JSON.

◆ 8. Ταξινόμηση (Sort)
```
students.Sort((a, b) => a.name.CompareTo(b.name));
```

Ταξινομεί τη λίστα αλφαβητικά ως προς το όνομα.

◆ 9. Αποθήκευση δεδομένων (Save)
```
string newJson = JsonSerializer.Serialize(students, new
JsonSerializerOptions { WriteIndented = true });
File.WriteAllText(file, newJson);
```

Μετατρέπει τη λίστα σε JSON και την αποθηκεύει στο αρχείο.

◆ 10. Ορισμός κλάσης (Model)

```
class Student
{
    public string name { get; set; }
    public int age { get; set; }
}
```

Ορίζει τη δομή των δεδομένων.

🎯 Συνολική Λειτουργία

Το πρόγραμμα ακολουθεί τη ροή:
`Αρχείο JSON → Αντικείμενα (List) → Επεξεργασία → Αποθήκευση σε JSON`

📌 Συμπέρασμα

Η εφαρμογή αποτελεί ένα ολοκληρωμένο παράδειγμα:

- διαχείρισης δεδομένων
- χρήσης JSON
- χρήσης λιστών και αντικειμένων
- οργάνωσης κώδικα με μεθόδους

και μπορεί να θεωρηθεί ένα βασικό αλλά πλήρες console application.