


```
1 namespace HospitalManagementSystem
2 {
3     public class AggressiveTreatment : TreatmentPlan
4     {
5         public string ExecuteTreatment(Patient patient)
6         {
7             return "Aggressive";
8         }
9     }
10 }
11
```

```
1 namespace HospitalManagementSystem
2 {
3     public class Cardiologist : Doctor
4     {
5         public Cardiologist(string name, string contact, DateTime dob,
6             string workingHours, string experience):base(name, contact, dob,
7             workingHours, experience)
8         {
9         }
10        public override void Diagnose()
11        {
12            Console.WriteLine($"{Name} (Cardiologist) is diagnosing a heart
13                condition.");
14        }
15        public override void ProvideTreatment()
16        {
17            Console.WriteLine($"{Name} (Cardiologist) is providing heart
18                treatment.");
19        }
20    }
21
```

```
1 namespace HospitalManagementSystem
2 {
3     public class Doctor : MedicalStaff, IObserver
4     {
5         public Doctor(string name, string contact, DateTime dob, string
            workingHours, string experience):base(name, contact, dob,
            workingHours, experience)
6         {
7         }
8
9         public void Update(Patient patient)
10        {
11            Console.WriteLine($"Doctor {Name} has been notified of the
            patient's update: {patient.Name}");
12
13            Console.WriteLine($"Updated Symptoms: {string.Join(", ",
            patient.Symptoms)}");
14        }
15
16        public override void Diagnose()
17        {
18            Console.WriteLine($"Doctor {Name} is diagnosing the patient.");
19        }
20
21        public override void ProvideTreatment()
22        {
23            Console.WriteLine($"Doctor {Name} is providing treatment.");
24        }
25    }
26
27 }
28
```

```
1 using HospitalManagementSystem;
2
3 public class EmergencyDecorator : MedicalStaff
4 {
5     private MedicalStaff _medical_staff;
6
7     public EmergencyDecorator(MedicalStaff medical_staff) : base      ↗
        (medical_staff.Name, medical_staff.Contact, medical_staff.DOB,    ↗
        medical_staff.WorkingHours, medical_staff.Experience)
8     {
9         _medical_staff = medical_staff;
10    }
11
12    public override void Diagnose()
13    {
14        _medical_staff.Diagnose();
15        Console.WriteLine("Emergency handling diagnosis.");
16    }
17
18    public override void ProvideTreatment()
19    {
20        _medical_staff.ProvideTreatment();
21        Console.WriteLine("Emergency treatment provided.");
22    }
23 }
24
```

```
1 namespace HospitalManagementSystem
2 {
3     public class EmergencyNurse : Nurse
4     {
5         public EmergencyNurse(string name, string contact, DateTime dob,
6             string working_hours, string experience):base(name, contact, dob,
7             working_hours, experience)
8         {
9         }
10
11        public override void UpdatePatientRecord(Manage hospital_manager,
12            string patient_name, string new_contact, string[] new_symptoms)
13        {
14            PatientRecord record_edit =
15                hospital_manager.FindPatientRecordByName(patient_name);
16
17            if (record_edit != null)
18            {
19                record_edit.Contact = new_contact;
20                record_edit.Symptoms = new_symptoms;
21                Console.WriteLine($"Patient record for {patient_name} has
22                    been updated.");
23                hospital_manager.SaveRecordsToFile();
24            }
25            else
26            {
27                Console.WriteLine($"No patient record found for
28                    {patient_name}.");
29            }
30        }
31
32        public override void Diagnose() { }
33
34        public override void ProvideTreatment() { }
35    }
36 }
```

```
1 using System;
2 using System.Collections.Generic;
3
4 namespace HospitalManagementSystem
5 {
6     public class Hospital
7     {
8         private List<Patient> _patients = new List<Patient>();
9         private List<MedicalStaff> _medical_staff = new List<MedicalStaff> ➤
10             ();
11         private List<Receptionist> _receptionists = new List<Receptionist> ➤
12             ();
13
14         public void AddPatient(Patient patient)
15         {
16             _patients.Add(patient);
17             Console.WriteLine("Patient added to the hospital.");
18         }
19
20         public void RemovePatient(Patient patient)
21         {
22             _patients.Remove(patient);
23             Console.WriteLine("Patient removed from the hospital.");
24         }
25
26         public void AddMedicalStaff(MedicalStaff staff)
27         {
28             _medical_staff.Add(staff);
29             Console.WriteLine("Medical staff added.");
30         }
31
32         public void AddReceptionist(Receptionist receptionist)
33         {
34             _receptionists.Add(receptionist);
35             Console.WriteLine("Receptionist added.");
36         }
37     }
38 }
```

```
1 namespace HospitalManagementSystem
2 {
3     public interface IObserver
4     {
5         void Update(Patient patient);
6     }
7 }
8
```

```
1 namespace HospitalManagementSystem
2 {
3     public class Manage
4     {
5         private List<PatientRecord> _patient_records = new
6             List<PatientRecord>();
7         private string _file_path = "E:/COS20007/CustomProgram/
8             HospitalRecords.txt";
9
10        public Manage()
11        {
12            LoadRecordsFromFile();
13        }
14        public void AddPatientRecord(Patient patient, string
15            treatment_plan, string assigned_staff)
16        {
17            PatientRecord new_record = new PatientRecord(patient.Name,
18                patient.DOB, patient.Contact, patient.Symptoms,
19                treatment_plan, assigned_staff);
20
21            _patient_records.Add(new_record);
22            Console.WriteLine($"Patient record for {patient.Name} has been
23                added.");
24
25            SaveRecordsToFile();
26        }
27
28        public void RemovePatientRecord(Patient patient)
29        {
30            PatientRecord record_remove = _patient_records.Find(r =>
31                r.Name == patient.Name);
32
33            if (record_remove != null)
34            {
35                _patient_records.Remove(record_remove);
36                Console.WriteLine($"Patient record for {patient.Name} has
37                    been removed.");
38                SaveRecordsToFile();
39            }
40            else
41            {
42                Console.WriteLine($"No patient record found for
43                    {patient.Name}.");
44            }
45        }
46
47        public void DisplayAllRecords()
```



```
41     {
42         if (_patient_records.Count == 0)
43         {
44             Console.WriteLine("No patient records available.");
45             return;
46         }
47
48         Console.WriteLine("Displaying all patient records:");
49         foreach (var record in _patient_records)
50         {
51             record.DisplayRecord();
52             Console.WriteLine();
53         }
54     }
55
56
57     public PatientRecord FindPatientRecordByName(string patient_name)
58     {
59         return _patient_records.Find(r => r.Name == patient_name);
60     }
61
62     public void SaveRecordsToFile() // Make this public for external saving ↗
63     {
64         using (StreamWriter writer = new StreamWriter(_file_path))
65         {
66             foreach (var record in _patient_records)
67             {
68                 writer.WriteLine(record.Name);
69                 writer.WriteLine(record.DateOfBirth.ToString("yyyy-MM- ↗
70                 dd"));
71                 writer.WriteLine(record.Contact);
72                 writer.WriteLine(string.Join(",", record.Symptoms));
73                 writer.WriteLine(record.TreatmentPlan);
74                 writer.WriteLine(record.AssignedStaff);
75             }
76             Console.WriteLine("Records saved to file.");
77         }
78
79     private void LoadRecordsFromFile()
80     {
81         if (File.Exists(_file_path))
82         {
83             using (StreamReader reader = new StreamReader(_file_path))
84             {
85                 while (!reader.EndOfStream)
86                 {
87                     string name = reader.ReadLine();
```

```
88         DateTime dob = DateTime.Parse(reader.ReadLine());
89         string contact = reader.ReadLine();
90         string[] symptoms = reader.ReadLine().Split(',');
91         string treatment_plan = reader.ReadLine();
92         string assigned_staff = reader.ReadLine();
93
94         PatientRecord record = new PatientRecord(name,
95             dob, contact, symptoms, treatment_plan, assigned_staff);
96         _patient_records.Add(record);
97     }
98     Console.WriteLine("Records loaded from file.");
99 }
100 else
101 {
102     Console.WriteLine("No records file found, starting
103         fresh.");
104 }
105 }
106 }
107
```

```
1 namespace HospitalManagementSystem
2 {
3     public abstract class MedicalStaff : Person
4     {
5         protected MedicalStaff(string name, string contact, DateTime dob,
6             string working_hours, string experience)
7             : base(name, contact, dob)
8         {
9             WorkingHours = working_hours;
10            Experience = experience;
11        }
12        public string WorkingHours
13        {
14            get;
15            set;
16        }
17
18        public string Experience
19        {
20            get;
21            set;
22        }
23
24        public abstract void Diagnose();
25
26        public abstract void ProvideTreatment();
27    }
28 }
29
30
```



```
1 namespace HospitalManagementSystem
2 {
3     public class MedicalStaffFactory
4     {
5         public MedicalStaff CreateStaff(string type, string name, string
6             contact, DateTime dob, string working_hours, string experience)
7         {
8             switch (type)
9             {
10                 case "SurgicalNurse":
11                     return new SurgicalNurse(name, contact, dob,
12                         working_hours, experience);
13                 case "Surgeon":
14                     return new Surgeon(name, contact, dob, working_hours,
15                         experience);
16                 case "Pediatrician":
17                     return new Pediatric(name, contact, dob, working_hours,
18                         experience);
19                 case "EmergencyNurse":
20                     return new EmergencyNurse(name, contact, dob,
21                         working_hours, experience);
22                 case "PediatricNurse":
23                     return new PediatricNurse(name, contact, dob,
24                         working_hours, experience);
25                 case "Psychologist":
26                     return new Psychologist(name, contact, dob,
27                         working_hours, experience);
28                 default:
29                     throw new ArgumentException("Invalid staff type");
30             }
31         }
32     }
33 }
```

```
1 namespace HospitalManagementSystem
2 {
3     public class MildTreatment : TreatmentPlan
4     {
5         public string ExecuteTreatment(Patient patient)
6         {
7             return "Mild";
8         }
9     }
10 }
11
12
```

```
1 namespace HospitalManagementSystem
2 {
3     public abstract class Nurse : MedicalStaff
4     {
5         protected Nurse(string name, string contact, DateTime dob, string
           working_hours, string experience):base(name, contact, dob,
           working_hours, experience)
6         {
7         }
8
9         public abstract void UpdatePatientRecord(Manage hospital_manager,
           string patientName, string newContact, string[] newSymptoms);
10     }
11 }
12
```

```
1 using System;
2 using System.Collections.Generic;
3
4 namespace HospitalManagementSystem
5 {
6     public class Patient : Person
7     {
8         private List<IObserver> _observers = new List<IObserver>();
9
10        public Patient(string name, DateTime dob, string contact, string[] ➤
            symptoms):base(name, contact, dob)
11        {
12            Symptoms = symptoms;
13        }
14
15        public string[] Symptoms
16        {
17            get;
18            set;
19        }
20
21        public void AttachObserver(IObserver observer)
22        {
23            if (!_observers.Contains(observer))
24            {
25                _observers.Add(observer);
26            }
27        }
28
29        public void DetachObserver(IObserver observer)
30        {
31            if (_observers.Contains(observer))
32            {
33                _observers.Remove(observer);
34            }
35        }
36
37        public void Notify()
38        {
39            foreach (var observer in _observers)
40            {
41                observer.Update(this); // Passing the patient data to the ➤
42                observers
43            }
44        }
45    }
46
```

```
1 namespace HospitalManagementSystem
2 {
3     public class PatientRecord
4     {
5         public PatientRecord(string name, DateTime dob, string contact,
6             string[] symptoms, string treatment_plan, string assigned_staff)
7         {
8             Name = name;
9             DateOfBirth = dob;
10            Contact = contact;
11            Symptoms = symptoms;
12            TreatmentPlan = treatment_plan;
13            AssignedStaff = assigned_staff;
14        }
15
16        public void DisplayRecord()
17        {
18            Console.WriteLine($"Name: {Name}");
19            Console.WriteLine($"Date of Birth: {DateOfBirth:yyyy-MM-dd}");
20            Console.WriteLine($"Contact: {Contact}");
21            Console.WriteLine($"Symptoms: {string.Join(", ", Symptoms)}");
22            Console.WriteLine($"Treatment Plan: {TreatmentPlan}");
23            Console.WriteLine($"Assigned Medical Staff: {AssignedStaff}");
24        }
25
26        public string Name
27        {
28            get;
29            set;
30        }
31
32        public DateTime DateOfBirth
33        {
34            get;
35            set;
36        }
37
38        public string Contact
39        {
40            get;
41            set;
42        }
43
44        public string[] Symptoms
45        {
46            get;
47            set;
48        }
49
50        public string TreatmentPlan
51        {
52            get;
53            set;
54        }
55    }
56 }
```

```
49     }
50     public string AssignedStaff
51     {
52         get;
53         set;
54     }
55 }
56 }
57
```

```
1 namespace HospitalManagementSystem
2 {
3     public class Pediatric : Doctor
4     {
5         public Pediatric(string name, string contact, DateTime dob, string
            workingHours, string experience):base(name, contact, dob,
            workingHours, experience)
6         {
7         }
8
9
10    public override void Diagnose()
11    {
12        Console.WriteLine($"{Name} (Pediatrician) is diagnosing a child
            patient.");
13    }
14
15    public override void ProvideTreatment()
16    {
17        Console.WriteLine($"{Name} (Pediatrician) is providing
            treatment.");
18    }
19 }
20 }
21
```



```
1 namespace HospitalManagementSystem
2 {
3     public class PediatricNurse : Nurse
4     {
5         public PediatricNurse(string name, string contact, DateTime dob,
6             string working_hours, string experience):base(name, contact, dob,
7             working_hours, experience)
8         {
9         }
10
11        public override void UpdatePatientRecord(Manage hospital_manager,
12            string patient_name, string new_contact, string[] new_symptoms)
13        {
14            PatientRecord recordToEdit =
15                hospital_manager.FindPatientRecordByName(patient_name);
16
17            if (recordToEdit != null)
18            {
19                recordToEdit.Contact = new_contact;
20                recordToEdit.Symptoms = new_symptoms;
21                Console.WriteLine($"Patient record for {patient_name} has
22                    been updated.");
23                hospital_manager.SaveRecordsToFile(); // Save updated
24                    records to file
25            }
26            else
27            {
28                Console.WriteLine($"No patient record found for
29                    {patient_name}.");
30            }
31        }
32
33        public override void Diagnose() { }
34
35        public override void ProvideTreatment() { }
36    }
37 }
```

```
1 namespace HospitalManagementSystem
2 {
3     public abstract class Person
4     {
5         protected Person(string name, string contact, DateTime dob)
6         {
7             Name = name;
8             Contact = contact;
9             DOB = dob;
10        }
11
12        public string Name
13        {
14            get;
15            set;
16        }
17
18        public string Contact
19        {
20            get;
21            set;
22        }
23
24        public DateTime DOB
25        {
26            get;
27            set;
28        }
29    }
30 }
31
32
33
```

```
1 namespace HospitalManagementSystem
2 {
3     public class Psiychologist : Doctor
4     {
5         public Psiychologist(string name, string contact, DateTime dob,
6             string working_hours, string experience):base(name, contact, dob,
7             working_hours, experience)
8         {
9         }
10
11     public override void Diagnose()
12     {
13         Console.WriteLine($"{Name} (Psiychologist) is diagnosing a
14             mental health condition.");
15     }
16
17     public override void ProvideTreatment()
18     {
19         Console.WriteLine($"{Name} (Psiychologist) is providing mental
20             health treatment.");
21     }
22 }
```

```
1 namespace HospitalManagementSystem
2 {
3     public class Receptionist : Person
4     {
5         public Receptionist(string name, string contact, DateTime dob):base
6             (name, contact, dob)
7         {
8         }
9
10     public void AddPatientRecord()
11     {
12         Console.WriteLine($"{Name} is adding a patient record to the
13             system.");
14     }
15 }
```

```
1 using System;
2
3 namespace HospitalManagementSystem
4 {
5     public class Surgeon : MedicalStaff
6     {
7
8         public Surgeon(string name, string contact, DateTime dob, string
           workingHours, string experience):base(name, contact, dob,
           workingHours, experience)
9     {
10    }
11
12    public override void Diagnose()
13    {
14        Console.WriteLine($"{Name} is diagnosing the patient.");
15    }
16
17    public override void ProvideTreatment()
18    {
19        Console.WriteLine($"{Name} is performing surgery.");
20    }
21 }
22 }
23
```

```
1 namespace HospitalManagementSystem
2 {
3     public class SurgicalNurse : Nurse
4     {
5         public SurgicalNurse(string name, string contact, DateTime dob,
6             string working_hours, string experience) : base(name, contact,
7             dob, working_hours, experience)
8         {
9         }
10
11        public override void UpdatePatientRecord(Manage hospital_manager,
12            string patient_name, string new_contact, string[] new_symptoms)
13        {
14            PatientRecord recordToEdit =
15                hospital_manager.FindPatientRecordByName(patient_name);
16
17            if (recordToEdit != null)
18            {
19                recordToEdit.Contact = new_contact;
20                recordToEdit.Symptoms = new_symptoms;
21                Console.WriteLine($"Patient record for {patient_name} has
22                    been updated.");
23                hospital_manager.SaveRecordsToFile(); // Save updated
24                    records to file
25            }
26            else
27            {
28                Console.WriteLine($"No patient record found for
29                    {patient_name}.");
30            }
31        }
32
33        public override void Diagnose() { }
34        public override void ProvideTreatment() { }
35    }
36 }
```

```
1 namespace HospitalManagementSystem
2 {
3     public interface TreatmentPlan
4     {
5         string ExecuteTreatment(Patient patient);
6     }
7 }
8
```

```
1 using HospitalManagementSystem;
2
3 class Program
4 {
5     static void Main(string[] args)
6     {
7         Manage hospitalManager = new Manage();
8         MedicalStaffFactory staffFactory = new MedicalStaffFactory();
9         Receptionist receptionist = new Receptionist("Alice",
10             "1234567890", new DateTime(1990, 5, 12));
11         SurgicalNurse surgicalNurse = new SurgicalNurse("Nurse Mary",
12             "4567891230", new DateTime(1982, 4, 15), "9 AM - 5 PM", "10
13             years");
14
15         while (true)
16         {
17             Console.WriteLine("\nWelcome to the Hospital Management
18                 System");
19             Console.WriteLine("1 - Add Patient Record");
20             Console.WriteLine("2 - Remove Patient Record");
21             Console.WriteLine("3 - Edit Patient Record");
22             Console.WriteLine("4 - Display All Patient Records");
23             Console.WriteLine("5 - Exit");
24
25             string option = Console.ReadLine();
26
27             switch (option)
28             {
29                 case "1":
30                     Console.Write("Name: ");
31                     string name = Console.ReadLine();
32
33                     Console.Write("Date of Birth (yyyy-mm-dd): ");
34                     DateTime dob = DateTime.Parse(Console.ReadLine());
35
36                     Console.Write("Contact Number: ");
37                     string contact = Console.ReadLine();
38
39                     Console.WriteLine("Please enter your symptoms (comma-
40                         separated): ");
41                     string[] symptoms = Console.ReadLine().Split(',');
42
43                     Patient newPatient = new Patient(name, dob, contact,
44                         symptoms);
45
46                     // Notify observers (Doctor) about the new patient
47                     record
48                     Doctor doctor = new Doctor("Dr. Smith", "9876543210",
```

```
new DateTime(1970, 6, 15), "9 AM - 5 PM", "15 years");
43     newPatient.AttachObserver(doctor); // Attach observer to patient
44
45     Console.WriteLine("Your information has been added to the hospital records.");
46
47     Console.WriteLine("Please select a treatment plan: 1 - Mild, 2 - Aggressive");
48     int treatmentPlanChoice;
49     while (!int.TryParse(Console.ReadLine(), out treatmentPlanChoice) || (treatmentPlanChoice != 1 && treatmentPlanChoice != 2))
50     {
51         Console.WriteLine("Invalid input. Please choose either 1 (Mild) or 2 (Aggressive): ");
52     }
53
54     TreatmentPlan treatmentPlan;
55     MedicalStaff assignedStaff;
56
57     // Strategy and Factory Patterns: Based on the treatment plan, assign the appropriate medical staff
58     if (treatmentPlanChoice == 1)
59     {
60         treatmentPlan = new MildTreatment();
61         assignedStaff = staffFactory.CreateStaff("PediatricNurse", "Nurse Nina", "7890123456", new DateTime(1985, 3, 21), "9 AM - 5 PM", "10 years");
62         hospitalManager.AddPatientRecord(newPatient, treatmentPlan.ExecuteTreatment(newPatient), assignedStaff.Name);
63     }
64     else
65     {
66         treatmentPlan = new AggressiveTreatment();
67         assignedStaff = staffFactory.CreateStaff("Surgeon", "Dr. John", "9876543210", new DateTime(1975, 3, 10), "24/7", "15 years");
68         hospitalManager.AddPatientRecord(newPatient, treatmentPlan.ExecuteTreatment(newPatient), assignedStaff.Name);
69     }
70
71     // Decorator Pattern: Add special abilities or expertise to the assigned medical staff
72     assignedStaff = new MedicalStaffDecorator(assignedStaff, "Cardiology Expertise");
73
```



```
74         // Display the assigned staff and begin treatment
75         Console.WriteLine($"You have been assigned to: {assignedStaff.GetType().Name} {assignedStaff.Name}");
76         Console.WriteLine("Your treatment plan is: " + treatmentPlan.ExecuteTreatment(newPatient));
77
78         // The assigned staff provides diagnosis and treatment
79         assignedStaff.Diagnose();
80         assignedStaff.ProvideTreatment();
81
82         // Update the patient record and notify doctor (Observer Pattern)
83         newPatient.Notify();
84
85
86
87
88         break;
89     case "2":
90         Console.Write("Enter the patient's name to remove: ");
91         string nameToRemove = Console.ReadLine();
92         hospitalManager.RemovePatientRecord(new Patient(nameToRemove, DateTime.MinValue, "", null));
93         break;
94
95     case "3":
96         Console.Write("Enter the patient's name to edit: ");
97         string nameToEdit = Console.ReadLine();
98
99         Console.Write("New Contact Number: ");
100        string newContact = Console.ReadLine();
101
102        Console.WriteLine("New Symptoms (comma-separated): ");
103        string[] newSymptoms = Console.ReadLine().Split(',');
104
105        surgicalNurse.UpdatePatientRecord(hospitalManager, nameToEdit, newContact, newSymptoms);
106        break;
107
108    case "4":
109        hospitalManager.DisplayAllRecords();
110        break;
111
112    case "5":
113        return;
114
115    default:
116        Console.WriteLine("Invalid option. Try again.");
117        break;
```

```
118         }  
119     }  
120 }  
121 }  
122
```