# UNIVERSITY INSTITUTE OF COMPUTING

**Subject Name:** Object Oriented Programming

**Subject Code:** 24CAH-201

**Project:** Hospital Management System using C++

**Submitted by:** **Submitted to:** Ms. Konika

Harman Kaur(24BCA10030)

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

NAAC GRADE A+ ACCREDITED UNIVERSITY

QS WORLD UNIVERSITY RANKINGS 2024
RANKED 1ST AMONGST PVT. UNIVERSITIES IN INDIA

## Acknowledgment

We would like to express our heartfelt gratitude to all those who helped us in the completion of this project.

First and foremost, we are deeply grateful to **Ms. Konika**, our respected teacher and guide, whose guidance, feedback, and continuous encouragement were invaluable throughout the project. Her insightful suggestions and expert knowledge played a crucial role in shaping the direction and successful completion of this work.

We also extend our sincere thanks to **Chandigarh University / UIC** for providing the necessary resources, support, and a conducive environment to carry out this project.

This project has been a great learning experience and has helped us enhance our **practical understanding of Object-Oriented Programming in C++**, as well as develop skills in **problem-solving, program design, and menu-driven application development**.

## 1. Introduction

The **Hospital Management System (HMS)** is a **C++ console-based application** designed to manage **basic hospital operations** efficiently. It allows users to **store, add, and display patient and staff records** in an organized and interactive manner.

This project demonstrates key **Object-Oriented Programming (OOP) concepts** such as **classes, inheritance, polymorphism, constructors, and default parameters**. The system manages **patient information** like ID, Name, Illness, and **staff information** like ID, Name, Role.

It is **menu-driven**, easy to use, and provides a foundation for more advanced hospital management solutions.

## 2. Objectives

The main objective of this project is to develop a **menu-driven C++ program** for hospital data management using **Object-Oriented Programming (OOP)** concepts like classes, inheritance, and polymorphism. It enables easy addition and display of patient and staff records while using default constructors for flexibility. The project also provides practical experience in building real-world applications with scope for future enhancements like billing, appointments, and database integration.

## 3. Features

- **Patient Management:** Add and display patient information (ID, Name, Illness).
- **Staff Management:** Add and display staff information (ID, Name, Role).
- **Menu-Driven Interface:** Interactive and user-friendly console navigation.

- **Polymorphism:** display() function works differently for patients and staff.
- **Constructors with Default Parameters:** Allows objects to be created with or without initial values.
- **Expandable System:** Can be enhanced to include appointments, billing, file handling, or database storage.

## 4. Learning Outcomes

After completing this project, students will be able to:

1. Understand and apply **Object-Oriented Programming concepts** in C++.
2. Implement **classes, objects, inheritance, and polymorphism** effectively.
3. Develop **menu-driven programs** for real-life applications.
4. Use **arrays to store multiple records** efficiently.
5. Create **constructors with default parameters** for flexible object initialization.
6. Gain experience in **designing simple management systems** and applying programming logic.

## 5. Working Process & Technologies Used

### Working Process

- The program displays a **menu repeatedly** until the user selects **Exit**, making it **interactive and easy to use**.
- Users can **add or view** both **patient** and **staff** details.
- All records are stored **temporarily using arrays** during execution.

## Optional Enhancements

- Add **file handling or database** support for permanent data storage.
- Include extra features like **appointments, billing, search, update, and delete** options.

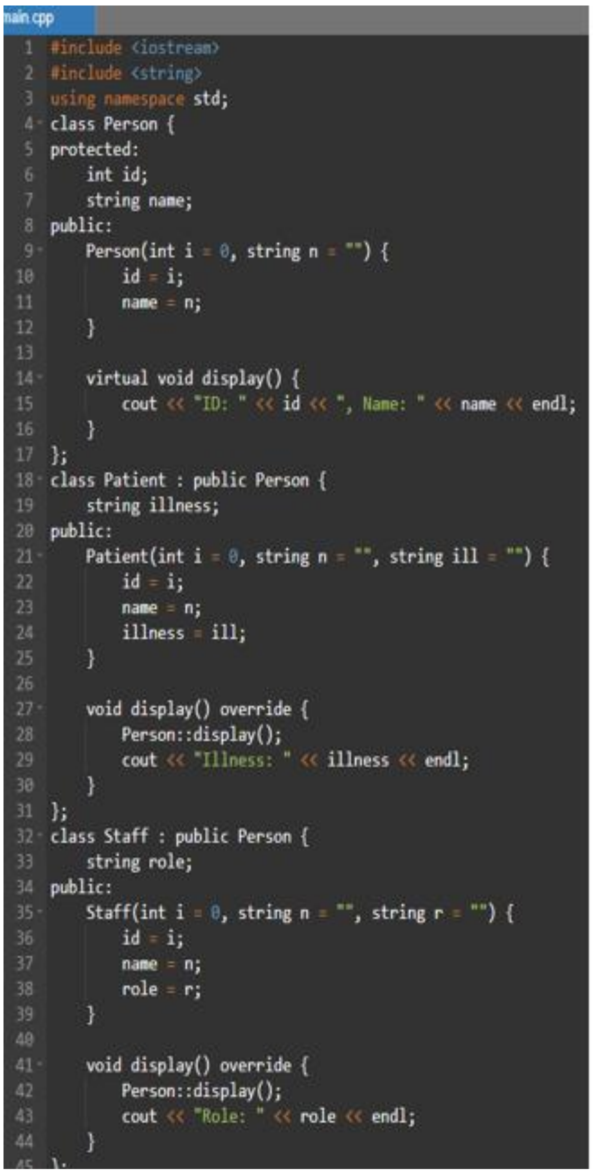
## Technologies Used

1. **Programming Language:**
    - **C++** – Used to apply key **OOP concepts** like *classes, inheritance, and polymorphism.*
2. **Compiler/IDE:**
    - **Code::Blocks**, **Dev-C++**, or **Visual Studio Code** – any standard C++ compiler works.
3. **Key C++ Concepts Applied:**
    - **Classes & Objects:** To define *Patient* and *Staff* entities.
    - **Inheritance:** Derived classes inherit from the base *Person* class.
    - **Polymorphism:** Overridden **display()** function for different outputs.
    - **Default Constructors:** Allow **flexible object creation.**
    - **Arrays:** For **storing multiple records efficiently.**
4. **Input/Output:**
    - Console-based **cin** and **cout** for user interaction.

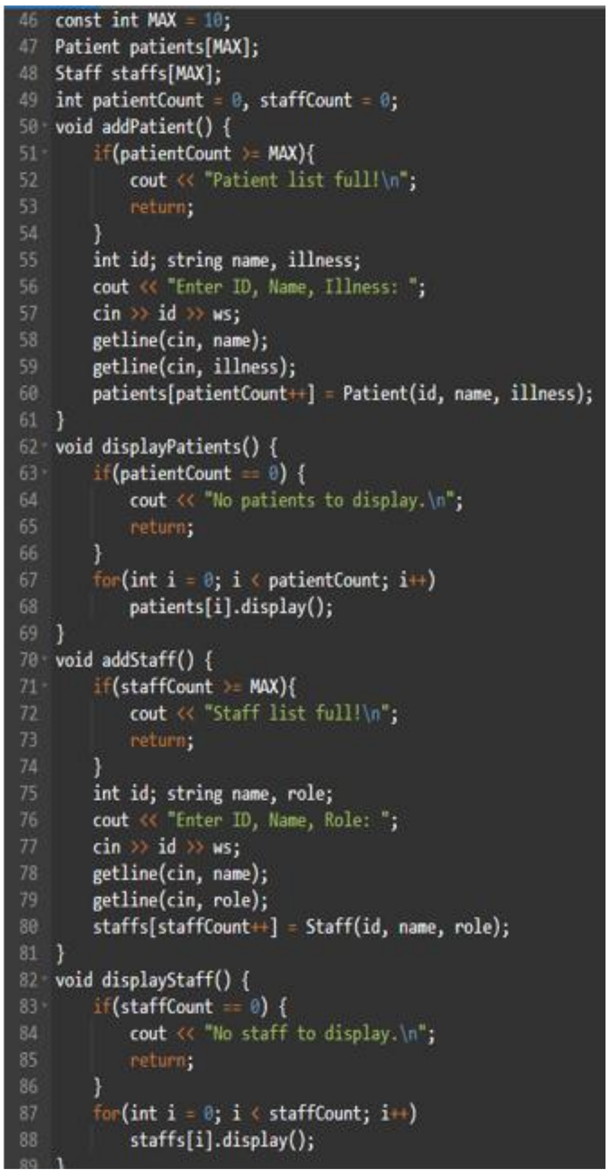
CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

NAAC GRADE A+ ACCREDITED UNIVERSITY

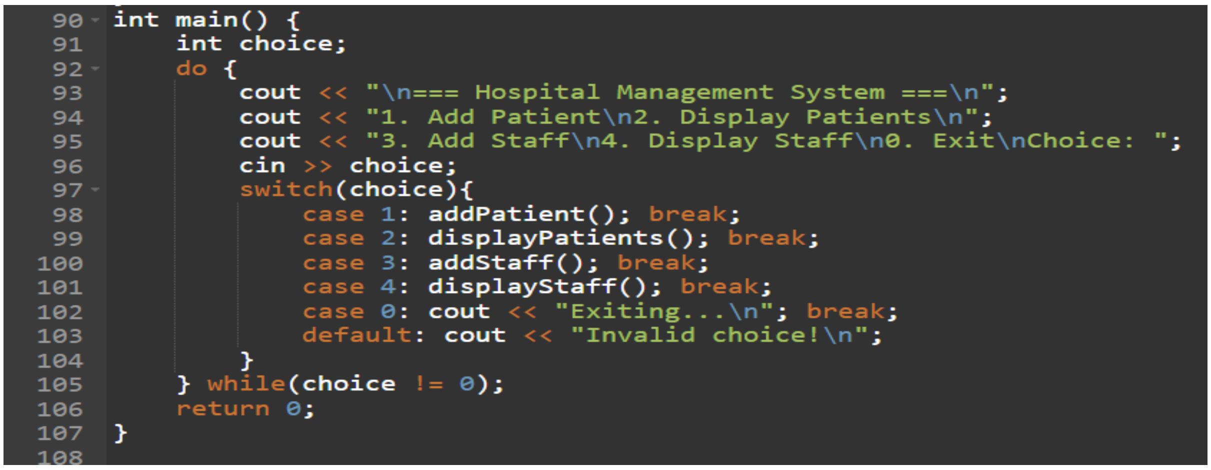
QS WORLD UNIVERSITY RANKINGS 2024
RANKED 1ST AMONGST PVT. UNIVERSITIES IN INDIA


main.cpp

```cpp
#include <iostream>
#include <string>
using namespace std;
class Person {
protected:
    int id;
    string name;
public:
    Person(int i = 0, string n = "") {
        id = i;
        name = n;
    }

    virtual void display() {
        cout << "ID: " << id << ", Name: " << name << endl;
    }
};
class Patient : public Person {
    string illness;
public:
    Patient(int i = 0, string n = "", string ill = "") {
        id = i;
        name = n;
        illness = ill;
    }

    void display() override {
        Person::display();
        cout << "Illness: " << illness << endl;
    }
};
class Staff : public Person {
    string role;
public:
    Staff(int i = 0, string n = "", string r = "") {
        id = i;
        name = n;
        role = r;
    }

    void display() override {
        Person::display();
        cout << "Role: " << role << endl;
    }
};
```

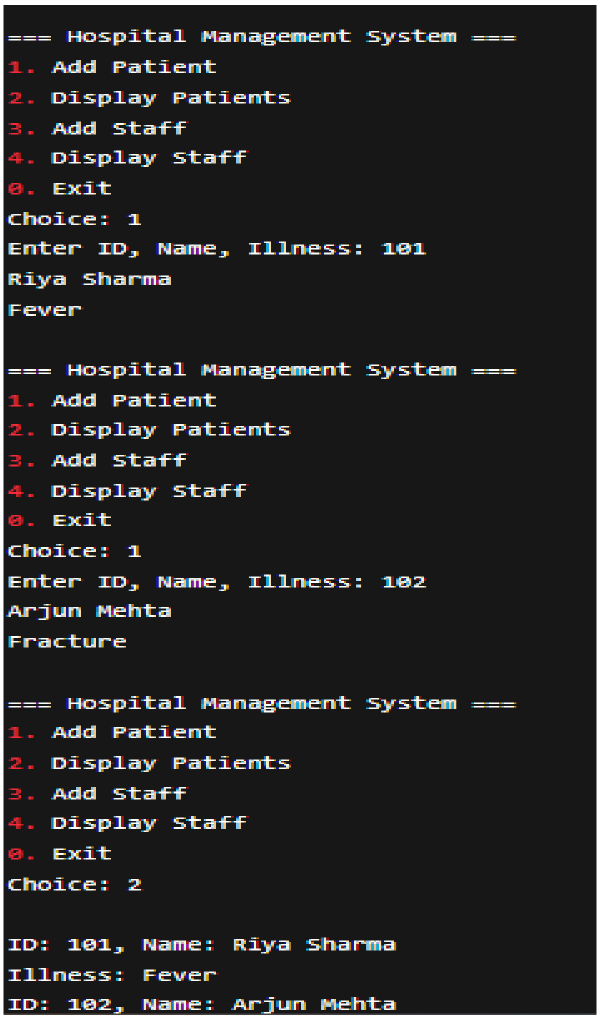


```cpp
const int MAX = 10;
Patient patients[MAX];
Staff staffs[MAX];
int patientCount = 0, staffCount = 0;
void addPatient() {
    if(patientCount >= MAX){
        cout << "Patient list full!\n";
        return;
    }
    int id; string name, illness;
    cout << "Enter ID, Name, Illness: ";
    cin >> id >> ws;
    getline(cin, name);
    getline(cin, illness);
    patients[patientCount++] = Patient(id, name, illness);
}
void displayPatients() {
    if(patientCount == 0) {
        cout << "No patients to display.\n";
        return;
    }
    for(int i = 0; i < patientCount; i++)
        patients[i].display();
}
void addStaff() {
    if(staffCount >= MAX){
        cout << "Staff list full!\n";
        return;
    }
    int id; string name, role;
    cout << "Enter ID, Name, Role: ";
    cin >> id >> ws;
    getline(cin, name);
    getline(cin, role);
    staffs[staffCount++] = Staff(id, name, role);
}
void displayStaff() {
    if(staffCount == 0) {
        cout << "No staff to display.\n";
        return;
    }
    for(int i = 0; i < staffCount; i++)
        staffs[i].display();
}
```

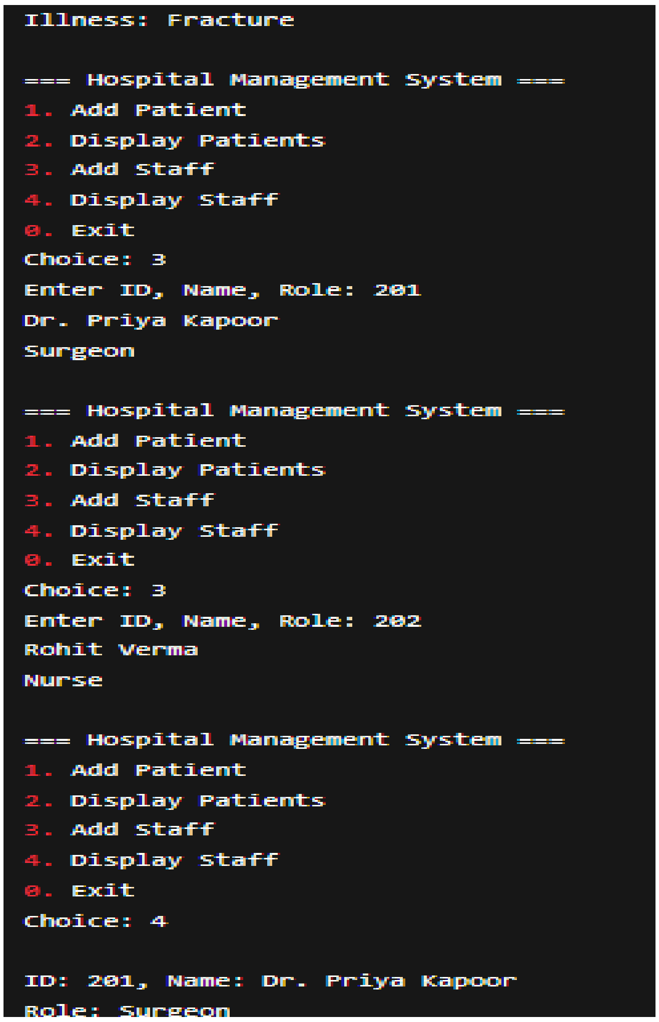


```cpp
int main() {
    int choice;
    do {
        cout << "\n=== Hospital Management System ===\n";
        cout << "1. Add Patient\n2. Display Patients\n";
        cout << "3. Add Staff\n4. Display Staff\n0. Exit\nChoice: ";
        cin >> choice;
        switch(choice){
            case 1: addPatient(); break;
            case 2: displayPatients(); break;
            case 3: addStaff(); break;
            case 4: displayStaff(); break;
            case 0: cout << "Exiting...\n"; break;
            default: cout << "Invalid choice!\n";
        }
    } while(choice != 0);
    return 0;
}
```

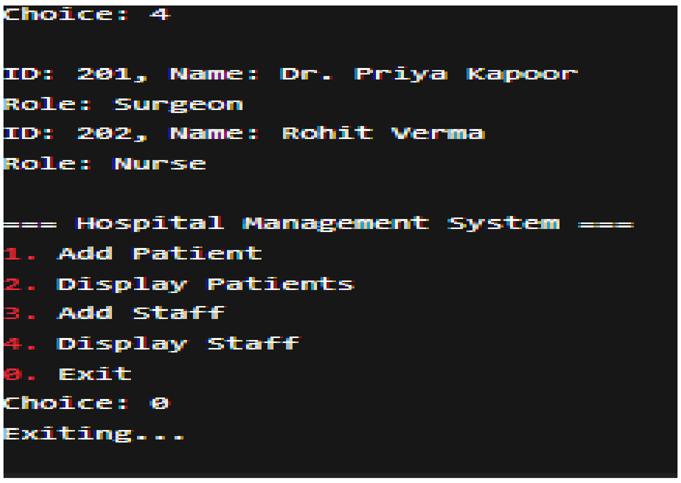
## Output:



```
=== Hospital Management System ===
1. Add Patient
2. Display Patients
3. Add Staff
4. Display Staff
0. Exit
Choice: 1
Enter ID, Name, Illness: 101
Riya Sharma
Fever

=== Hospital Management System ===
1. Add Patient
2. Display Patients
3. Add Staff
4. Display Staff
0. Exit
Choice: 1
Enter ID, Name, Illness: 102
Arjun Mehta
Fracture

=== Hospital Management System ===
1. Add Patient
2. Display Patients
3. Add Staff
4. Display Staff
0. Exit
Choice: 2

ID: 101, Name: Riya Sharma
Illness: Fever
ID: 102, Name: Arjun Mehta
```



```
Illness: Fracture

=== Hospital Management System ===
1. Add Patient
2. Display Patients
3. Add Staff
4. Display Staff
0. Exit
Choice: 3
Enter ID, Name, Role: 201
Dr. Priya Kapoor
Surgeon

=== Hospital Management System ===
1. Add Patient
2. Display Patients
3. Add Staff
4. Display Staff
0. Exit
Choice: 3
Enter ID, Name, Role: 202
Rohit Verma
Nurse

=== Hospital Management System ===
1. Add Patient
2. Display Patients
3. Add Staff
4. Display Staff
0. Exit
Choice: 4

ID: 201, Name: Dr. Priya Kapoor
Role: Surgeon
```



```
Choice: 4

ID: 201, Name: Dr. Priya Kapoor
Role: Surgeon
ID: 202, Name: Rohit Verma
Role: Nurse

=== Hospital Management System ===
1. Add Patient
2. Display Patients
3. Add Staff
4. Display Staff
0. Exit
Choice: 0
Exiting...
```

## 7. Conclusion

The **Hospital Management System** project helped us understand the practical use of **Object-Oriented Programming (OOP)** in **C++**.
We created a simple and user-friendly program to manage patient and staff details effectively.
This project improved our understanding of **classes, inheritance, and polymorphism**, and strengthened our **logic-building and coding skills**.
Overall, it was a great learning experience that gave us confidence in applying OOP concepts to real-life situations.