PROGRAM FOR HOSPITAL MANAGEMENT

```
// C Program to implement Hospital Management System
#include <ctype.h> // Include ctype.h for strcasecmp
#include <stdio.h>
#include <string.h>

// Define a structure for Hospital
struct Hospital {
   char name[50];
   char city[50];
   int beds;
   float price;
   float rating;
   int reviews;
};

// Define a structure for Patient
struct Patient {
   char name[50];
   int age;
};

// Function to print hospital data
void printHospital(struct Hospital hosp)
{
   printf("Hospital Name: %s\n", hosp.name);
   printf("City: %s\n", hosp.city);
   printf("Total Beds: %d\n", hosp.beds);
   printf("Price per Bed: $%.2f\n", hosp.price);
```

```
    printf("Rating: %.1f\n", hosp.rating);
    printf("Reviews: %d\n", hosp.reviews);
    printf("\n");
}

// Function to sort hospitals by beds price (ascending)
void sortByPrice(struct Hospital hospitals[], int n)
{
    // Implement sorting logic (e.g., bubble sort)
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (hospitals[j].price
                > hospitals[j + 1].price) {
                struct Hospital temp = hospitals[j];
                hospitals[j] = hospitals[j + 1];
                hospitals[j + 1] = temp;
            }
        }
    }
}

// Function to sort hospitals by name (ascending)
void sortByName(struct Hospital hospitals[], int n)
{
    // Implement sorting logic (e.g., using strcmp)
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (strcmp(hospitals[j].name,
                       hospitals[j + 1].name)
            > 0) {
            struct Hospital temp = hospitals[j];
            hospitals[j] = hospitals[j + 1];
            hospitals[j + 1] = temp;
          }
        }
    }
}

// Function to sort hospitals by rating and reviews
// (descending)
void sortByRating(struct Hospital hospitals[], int n)
{
    // Implement sorting logic (e.g., based on rating and
    // reviews)
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (hospitals[j].rating * hospitals[j].reviews
                < hospitals[j + 1].rating
                      * hospitals[j + 1].reviews) {
                struct Hospital temp = hospitals[j];
                hospitals[j] = hospitals[j + 1];
                hospitals[j + 1] = temp;
            }
        }
    }
}

// Function to print hospitals in a specific city
// (case-insensitive)
void printHospitalsInCity(struct Hospital hospitals[])
{
    char city[50];
    int hospitalsFound
        = 0; // Counter for hospitals found in the city

    printf("Enter city name (X, Y or Z): ");
    scanf("%s", city);

    printf("Hospitals in %s:\n", city);

    for (int i = 0; i < 5; i++) {
        // Use strcasecmp for case-insensitive comparison
        if (strcasecmp(hospitals[i].city, city) == 0) {
            printf("Hospital Name: %s\n",
                   hospitals[i].name);
            printf("City: %s\n", hospitals[i].city);
            printf("Total Beds: %d\n", hospitals[i].beds);
            printf("Price per Bed: $%.2f\n",
                   hospitals[i].price);
            printf("Rating: %.1f\n", hospitals[i].rating);
            printf("Reviews: %d\n", hospitals[i].reviews);
            printf("\n");
            hospitalsFound++;
        }
    }

    if (hospitalsFound == 0) {
        printf("No hospitals found in %s\n", city);
    }
}

// Function to sort hospitals by available beds (descending)
void sortByBeds(struct Hospital hospitals[], int n)
{
    // Implement sorting logic (e.g., bubble sort)
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (hospitals[j].beds < hospitals[j + 1].beds) {
                struct Hospital temp = hospitals[j];
                hospitals[j] = hospitals[j + 1];
                hospitals[j + 1] = temp;
            }
        }
    }
}

// Function to print patient data
void printPatient(struct Patient patient)
{
    printf("Patient Name: %s\n", patient.name);
    printf("Age: %d\n", patient.age);
    printf("\n");
}

int main()
{
```

```
// Sample hospital data
struct Hospital hospitals[5]
    = { { "Hospital A", "X", 100, 250.0, 4.5, 100 },
        { "Hospital B", "Y", 150, 200.0, 4.2, 80 },
        { "Hospital C", "X", 200, 180.0, 4.0, 120 },
        { "Hospital D", "Z", 80, 300.0, 4.8, 90 },
        { "Hospital E", "Y", 120, 220.0, 4.6, 110 } };

// Sample patient data (associated with hospitals)
struct Patient patients[5][3] = { { { "Amar", 35 },
                                    { "Manish", 45 },
                                    { "Atul", 28 } },
                                  { { "Elvish", 62 },
                                    { "Debolina", 18 },
                                    { "Shruti", 55 } },
                                  { { "Zafar", 50 },
                                    { "Rahul", 30 },
                                    { "Priya", 40 } },
                                  { { "Amir", 22 },
                                    { "Asif", 38 },
                                    { "Prince", 60 } },
                                  { { "Aditya", 28 },
                                    { "Aman", 48 },
                                    { "Sahil", 33 } } };

int n = 5; // Number of hospitals


int choice;
char city[50];
```

```
do {
  printf("\n\n\n*********** Hospital Management "
       "System Menu:************\n\n");
  printf("1. Printing Hospital Data\n");
  printf("2. Printing Patients Data\n");
  printf("3. Sorting Hospitals by Beds Price\n");
  printf("4. Sorting Hospitals by Available Beds\n");
  printf("5. Sorting Hospitals by Name\n");
  printf(
    "6. Sorting Hospitals by Rating and Reviews\n");
  printf("7. Print Hospitals in a Specific City\n");
  printf("8. Exit\n\n");
  printf("Enter your choice: ");
  scanf("%d", &choice);

  switch (choice) {
  case 1:
    printf("\nPrinting Hospital Data:\n\n");
    for (int i = 0; i < n; i++) {
      printHospital(hospitals[i]);
    }
    break;
  case 2:
    printf("Printing Patients Data:\n\n");
    for (int i = 0; i < n; i++) {
      printf("Hospital: %s\n", hospitals[i].name);
      for (int j = 0; j < 3; j++) {
        printPatient(patients[i][j]);
        }
    }
    break;
case 3:
    printf("Sorting Hospitals by Beds Price "
           "(Ascending):\n");
    sortByBeds(hospitals, n);
    for (int i = 0; i < n; i++) {
        printHospital(hospitals[i]);
    }
    break;
case 4:
    printf("Sorting Hospitals by Available Beds "
           "(Descending):\n");
    sortByBeds(hospitals,
               n); // Fix: Sorting by available beds
    for (int i = 0; i < n; i++) {
        printHospital(hospitals[i]);
    }
    break;
case 5:
    printf(
        "Sorting Hospitals by Name (Ascending):\n");
    sortByName(hospitals, n);
    for (int i = 0; i < n; i++) {
        printHospital(hospitals[i]);
    }
    break;
case 6:
      printf("Sorting Hospitals by Rating and "
          "Reviews (Descending):\n");
      sortByRating(hospitals, n);
      for (int i = 0; i < n; i++) {
        printHospital(hospitals[i]);
      }
      break;
    case 7:
      printHospitalsInCity(hospitals);
      break;
    case 8:
      printf("Exiting the program.\n");
      break;
    default:
      printf("Invalid choice. Please enter a valid "
          "option.\n");
    }
  } while (choice != 8);

  return 0;
}
```