SOLUTION OF QUESTION 1:

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

class Product
{
    public int ProductID;
    public string ProductName;
    public double Price;
    public string Category;
    public int StockQuantity;
    public string CompanyName;

    public void Display()
    {
        Console.WriteLine($"{ProductID,-10}{ProductName,-
15}{Price,-10}{Category,-15}{StockQuantity,-
10}{CompanyName,-15}");
    }
}
```

```
class Program
{
    static List<Product> products = new List<Product>();

    static void Main()
    {
        int choice;
        do
        {
            Console.WriteLine("\n===== TUCK SHOP INVENTORY
MANAGEMENT =====");
            Console.WriteLine("1. Add Product");
            Console.WriteLine("2. Update Product");
            Console.WriteLine("3. Remove Product");
            Console.WriteLine("4. Search Product by ID");
            Console.WriteLine("5. Search Products by Price Range");
            Console.WriteLine("6. Display All Products");
            Console.WriteLine("7. Exit");
            Console.Write("Enter your choice: ");
```

```
choice = int.Parse(Console.ReadLine());

switch (choice)
{
    case 1:
        AddProduct();
        break;
    case 2:
        UpdateProduct();
        break;
    case 3:
        RemoveProduct();
        break;
    case 4:
        SearchByID();
        break;
    case 5:
        SearchByPriceRange();
        break;
    case 6:
```

```
                DisplayAllProducts();
                break;
            case 7:
                Console.WriteLine("Exiting program...");
                break;
            default:
                Console.WriteLine("Invalid choice! Try again.");
                break;
        }
    } while (choice != 7);
}

static void AddProduct()
{
    Product p = new Product();
    Console.Write("Enter Product ID: ");
    p.ProductID = int.Parse(Console.ReadLine());

    Console.Write("Enter Product Name: ");
    p.ProductName = Console.ReadLine();
```

```
    Console.Write("Enter Price: ");
    p.Price = double.Parse(Console.ReadLine());

    Console.Write("Enter Category: ");
    p.Category = Console.ReadLine();

    Console.Write("Enter Stock Quantity: ");
    p.StockQuantity = int.Parse(Console.ReadLine());

    Console.Write("Enter Company Name: ");
    p.CompanyName = Console.ReadLine();

    products.Add(p);
    Console.WriteLine("Product added successfully!");
}

static void UpdateProduct()
{
    Console.Write("Enter Product ID to update: ");
```

```
int id = int.Parse(Console.ReadLine());
Product p = products.Find(x => x.ProductID == id);

if (p != null)
{
    Console.Write("Enter New Name: ");
    p.ProductName = Console.ReadLine();

    Console.Write("Enter New Price: ");
    p.Price = double.Parse(Console.ReadLine());

    Console.Write("Enter New Category: ");
    p.Category = Console.ReadLine();

    Console.Write("Enter New Stock Quantity: ");
    p.StockQuantity = int.Parse(Console.ReadLine());

    Console.Write("Enter New Company Name: ");
    p.CompanyName = Console.ReadLine();
```

```
        Console.WriteLine("Product updated successfully!");
    }
    else
    {
        Console.WriteLine("Product not found!");
    }
}

static void RemoveProduct()
{
    Console.Write("Enter Product ID to remove: ");
    int id = int.Parse(Console.ReadLine());
    Product p = products.Find(x => x.ProductID == id);

    if (p != null)
    {
        products.Remove(p);
        Console.WriteLine("Product removed successfully!");
    }
    else
```

```
        {
            Console.WriteLine("Product not found!");
        }
    }

    static void SearchByID()
    {
        Console.Write("Enter Product ID to search: ");
        int id = int.Parse(Console.ReadLine());
        Product p = products.Find(x => x.ProductID == id);

        if (p != null)
        {
            Console.WriteLine("\nProduct Found:");

Console.WriteLine("ID\tName\tPrice\tCategory\tStock\tCompa
ny");
            p.Display();
        }
        else
        {
```

```
            Console.WriteLine("Product not found!");
        }
    }

    static void SearchByPriceRange()
    {
        Console.Write("Enter Minimum Price: ");
        double min = double.Parse(Console.ReadLine());
        Console.Write("Enter Maximum Price: ");
        double max = double.Parse(Console.ReadLine());

        var result = products.FindAll(x => x.Price >= min && x.Price
<= max);

        if (result.Count > 0)
        {
            Console.WriteLine("\nProducts in Price Range:");

Console.WriteLine("ID\tName\tPrice\tCategory\tStock\tCompa
ny");
            foreach (var p in result)
```

```
            p.Display();
        }
        else
        {
            Console.WriteLine("No products found in this range!");
        }
    }

    static void DisplayAllProducts()
    {
        if (products.Count == 0)
        {
            Console.WriteLine("No products in inventory!");
            return;
        }

        Console.WriteLine("\nAll Products:");

Console.WriteLine("ID\tName\tPrice\tCategory\tStock\tCompa
ny");
        foreach (var p in products)
```

```
            p.Display();
        }
}
```

SOLUTION OF QUESTION 2:

```
using System;
using System.Collections.Generic;
using System.Windows.Forms;

namespace StudentCourseManagement
{
    public partial class Form1 : Form
    {
        List<Student> students = new List<Student>();
        StudentManager manager = new
StudentManager();

        public Form1()
```

```
{
    InitializeComponent();
    LoadCities();
    LoadDepartments();
}

private void LoadCities()
{
    comboCity.Items.Clear();
    comboCity.Items.AddRange(new string[]
    {
        // Punjab
        "Lahore", "Faisalabad", "Multan",
        // Sindh
        "Karachi", "Hyderabad", "Sukkur",
        // KPK
        "Peshawar", "Abbottabad", "Mardan",
        // Balochistan
        "Quetta", "Gwadar", "Khuzdar"
```

```
        });
    }

    private void LoadDepartments()
    {
        comboDepartment.Items.Clear();
        comboDepartment.Items.AddRange(new string[]
{ "CS", "Math", "Others" });
    }

    private void
comboDepartment_SelectedIndexChanged(object
sender, EventArgs e)
    {
        comboCourse.Items.Clear();
        if (comboDepartment.Text == "CS")
        {
            comboCourse.Items.AddRange(new string[]
            { "Programming", "Database", "Networking",
"AI", "Web Development" });
```

```
        }
        else if (comboDepartment.Text == "Math")
        {
            comboCourse.Items.AddRange(new string[]
            { "Algebra", "Calculus", "Geometry",
"Statistics", "Linear Algebra" });
        }
        else
        {
            comboCourse.Items.AddRange(new string[]
            { "English", "History", "Economics", "Islamic
Studies", "Pak Studies" });
        }
    }

    private void btnInsert_Click(object sender,
EventArgs e)
    {
        if (students.Count >= 10)
```

```
        {
            MessageBox.Show("Maximum 10 students
allowed!");
            btnInsert.Enabled = false;
            return;
        }

        if (!ValidateInput()) return;

        Student s = new Student();
        s.StudentID = int.Parse(txtID.Text);
        s.Name = txtName.Text;
        s.City = comboCity.Text;
        s.Department = comboDepartment.Text;
        s.Course = comboCourse.Text;
        s.DOB = dateTimePicker1.Value;
        s.Gender = rdoMale.Checked ? "Male" :
"Female";
```

```
        s.MaritalStatus = chkMarried.Checked ?
"Married" : "Single";


        manager.AddStudent(s, students);
        MessageBox.Show("Student added
successfully!");
        ClearFields();
    }


    private void btnUpdate_Click(object sender,
EventArgs e)
    {
        if (!ValidateInput()) return;


        Student s = new Student();
        s.StudentID = int.Parse(txtID.Text);
        s.Name = txtName.Text;
        s.City = comboCity.Text;
        s.Department = comboDepartment.Text;
```

```
        s.Course = comboCourse.Text;
        s.DOB = dateTimePicker1.Value;
        s.Gender = rdoMale.Checked ? "Male" :
"Female";
        s.MaritalStatus = chkMarried.Checked ?
"Married" : "Single";

        bool updated = manager.UpdateStudent(s,
students);
        if (updated)
            MessageBox.Show("Student updated
successfully!");
        else
            MessageBox.Show("Student not found!");
    }

    private void btnDelete_Click(object sender,
EventArgs e)
    {
        if (txtID.Text == "")
```

```
        {
            MessageBox.Show("Enter Student ID to
delete!");
            return;
        }

        int id = int.Parse(txtID.Text);
        bool deleted = manager.DeleteStudent(id,
students);

        if (deleted)
            MessageBox.Show("Student deleted
successfully!");
        else
            MessageBox.Show("Student not found!");
    }

    private void btnGet_Click(object sender, EventArgs
e)
    {
```

```
        if (txtID.Text == "")
        {
            MessageBox.Show("Enter Student ID to
search!");
            return;
        }

        int id = int.Parse(txtID.Text);
        Student s = manager.GetStudent(id, students);

        if (s != null)
        {
            txtName.Text = s.Name;
            comboCity.Text = s.City;
            comboDepartment.Text = s.Department;
            comboCourse.Text = s.Course;
            dateTimePicker1.Value = s.DOB;
            rdoMale.Checked = s.Gender == "Male";
            rdoFemale.Checked = s.Gender == "Female";
```

```
            chkMarried.Checked = s.MaritalStatus ==
"Married";
            MessageBox.Show("Student found!");
        }
        else
        {
            MessageBox.Show("Student not found!");
        }
    }

    private bool ValidateInput()
    {
        if (string.IsNullOrWhiteSpace(txtID.Text) ||
!int.TryParse(txtID.Text, out _))
        {
            MessageBox.Show("Student ID must be
numeric!");
            return false;
        }
```

```
        if (string.IsNullOrWhiteSpace(txtName.Text))
        {
            MessageBox.Show("Student name cannot be
empty!");
            return false;
        }
        if (comboCity.SelectedIndex == -1)
        {
            MessageBox.Show("Select a city!");
            return false;
        }
        if (comboDepartment.SelectedIndex == -1)
        {
            MessageBox.Show("Select a department!");
            return false;
        }
        if (comboCourse.SelectedIndex == -1)
        {
            MessageBox.Show("Select a course!");
```

```
            return false;
        }
        if (dateTimePicker1.Value > DateTime.Now)
        {
            MessageBox.Show("Date of birth cannot be in
the future!");
            return false;
        }
        if (!rdoMale.Checked && !rdoFemale.Checked)
        {
            MessageBox.Show("Select gender!");
            return false;
        }

        return true;
    }

    private void ClearFields()
    {
```

```
        txtID.Clear();
        txtName.Clear();
        comboCity.SelectedIndex = -1;
        comboDepartment.SelectedIndex = -1;
        comboCourse.SelectedIndex = -1;
        rdoMale.Checked = false;
        rdoFemale.Checked = false;
        chkMarried.Checked = false;
    }
}

// ======= Student Class =======
public class Student
{
    public int StudentID;
    public string Name;
    public string City;
    public string Department;
    public string Course;
```

```
        public DateTime DOB;
        public string Gender;
        public string MaritalStatus;
    }

    // ======= Student Manager Class =======
    public class StudentManager
    {
        public void AddStudent(Student s, List<Student>
list)
        {
            list.Add(s);
        }

        public bool UpdateStudent(Student s, List<Student>
list)
        {
            Student existing = list.Find(x => x.StudentID ==
s.StudentID);
```

```
    if (existing != null)
    {
        existing.Name = s.Name;
        existing.City = s.City;
        existing.Department = s.Department;
        existing.Course = s.Course;
        existing.DOB = s.DOB;
        existing.Gender = s.Gender;
        existing.MaritalStatus = s.MaritalStatus;
        return true;
    }
    return false;
}

public bool DeleteStudent(int id, List<Student> list)
{
    Student s = list.Find(x => x.StudentID == id);
    if (s != null)
    {
```

```
            list.Remove(s);
            return true;
        }
        return false;
    }

    public Student GetStudent(int id, List<Student> list)
    {
        return list.Find(x => x.StudentID == id);
    }
  }
}
```