4.2 Classes

This section will guide you to:

* Create a Windows Console project in Visual Studio to demonstrate the use of classes
* Create classes to store data about students, subjects, teachers, and classes

**Development Environment**

* Visual Studio 2019 Community Version
* Windows 10

This guide has nine subsections, namely:

* + 1. Creating a Windows Console project in Visual Studio to demonstrate the use of classes
    2. Creating a Subject class to store data about subjects
    3. Creating a Teacher class to store data about teachers
    4. Creating a Student class to store data about students
    5. Creating a CClass class to store data about students, subjects, and teachers
    6. Adding a method, runApp(), in Program class to create objects and populate them
    7. Building the project
    8. Publishing and running the project
    9. Pushing the code to your GitHub repositories

**Step 4.2.1:** Creating a Windows Console project in Visual Studio to demonstrate the use of classes

* Open Visual Studio.
* From the top menu, select **File->New->Project.**
* In **Create A New Project** screen, select **Console app (.NET Core)** from the list of available project types and click on **Next.**
* Enter **Project Name** as **Phase1Section5.6** and click on **Create.**
* This will create the files for a Windows Console project.

**Step 4.2.2:** Creating a Subject class to store data about subjects

* In the **Solution Explorer** window, right-click **Phase1Section5.6** and choose **Add->Class.**
* Enter Class name as **Subject.cs** and click **Add.**
* Enter the following code:

**using** System;

**using** System.Collections.Generic;

**using** System.Text;

**namespace** Phase1Section5.\_6

{

**public** **class** Subject

{

**private** **string** name;

**private** **string** shortName;

**public** **string** Name

{

**get** { **return** name; }

**set** { **this**.name = value; }

}

**public** **string** ShortName

{

**get** { **return** shortName; }

**set** { shortName = value; }

}

}

}

**Step 4.2.3:** Creating a Teacher class to store data about teachers

* In the **Solution Explorer** window, right-click **Phase1Section5.6** and choose **Add->Class.**
* Enter Class name as **Teacher.cs** and click **Add.**
* Enter the following code:

**using** System;

**using** System.Collections.Generic;

**using** System.Text;

**namespace** Phase1Section5.\_6

{

**public** **class** Teacher

{

**private** **string** name;

**private** **string** address;

**private** **string** contactAddress;

**private** DateTime dateOfJoining;

**public** **string** Name

{

**get** { **return** name; }

**set** { name = value; }

}

**public** **string** ContactAddress

{

**get** { **return** address; }

**set** { address = value; }

}

**public** DateTime DateOfJoining

{

**get** { **return** dateOfJoining; }

**set** { dateOfJoining = value; }

}

}

}

**Step 4.2.4:** Creating a Student class to store data about students

* In the **Solution Explorer** window, right-click **Phase1Section5.6** and choose **Add->Class.**
* Enter Class name as **Student.cs** and click **Add.**
* Enter the following code:

**using** System;

**using** System.Collections.Generic;

**using** System.Text;

**namespace** Phase1Section5.\_6

{

**public** **class** Student

{

**private** **string** name;

**private** DateTime dateOfBirth;

**private** **string** address;

**private** **string** guardianName;

**private** **string** contactNumber;

**public** **string** Name

{

**get** { **return** name; }

**set** { name = value; }

}

**public** DateTime DateOfBirth

{

**get** { **return** dateOfBirth; }

**set** { dateOfBirth = value; }

}

**public** **string** Address

{

**get** { **return** address; }

**set** { address = value; }

}

**public** **string** GuardianName

{

**get** { **return** guardianName; }

**set** { guardianName = value; }

}

**public** **string** ContactNumber

{

**get** { **return** contactNumber; }

**set** { contactNumber = value; }

}

}

}

**Step 4.2.5:** Creating a CClass class to store data about students, subjects, and teachers

* In the **Solution Explorer** window, right-click **Phase1Section5.6** and choose **Add->Class.**
* Enter Class name as **CClass.cs** and click **Add.**
* Enter the following code:

**using** System;

**using** System.Collections.Generic;

**using** System.Text;

**using** System.Collections;

**namespace** Phase1Section5.\_6

{

**public** **class** CClass

{

**private** **string** name;

**private** Teacher classTeacher;

**private** List<Student> students;

**private** List<Subject> subjects;

**public** **string** Name

{

**get** { **return** name; }

**set** { name = value; }

}

**public** Teacher ClassTeacher

{

**get** { **return** classTeacher; }

**set** { classTeacher = value; }

}

**public** List<Student> Students

{

**get** { **return** students; }

**set** { students = value; }

}

**public** List<Subject> Subjects

{

**get** { **return** subjects; }

**set** { subjects = value; }

}

}

}

**Step 4.2.6:** Adding a method, runApp(), in Program class to create objects and populate them

* Select **Program.cs** as the current Code tab.
* Enter the following code:

**using** System;

**using** System.Collections;

**using** System.Collections.Generic;

**namespace** Phase1Section5.\_6

{

**class** Program

{

**static** **void** Main(**string**[] args)

{

runApp();

}

**public** **static** **void** runApp()

{

**string**[] arrSubjects = { "English Literature", "English Language", "Maths", "Geography", "History", "Physics", "Chemistry", "Biology" };

**string**[] arrSubjectShorts = { "ELIT", "ELANG", "MATH", "GEOG", "HIST", "PHY", "CHEM", "BIO" };

**string**[] arrStudents = { "Rahul Gandhi", "Vijay Mallya", "Lady Gaga", "Arun Jaitley", "MS Dhoni", "Kim Kardashian" };

List<Student> listStudents = **new** List<Student>();

List<Subject> listSubjects = **new** List<Subject>();

**for** (**int** i =0; i < arrSubjects.Length; i++)

{

Subject subject = **new** Subject();

subject.Name = arrSubjects[i];

subject.ShortName = arrSubjectShorts[i];

listSubjects.Add(subject);

}

**for** (**int** i = 0; i < arrStudents.Length; i++)

{

Student student= **new** Student();

student.Name = arrStudents[i];

student.Address = "Some address";

student.ContactNumber = "124567";

student.GuardianName = "My Guardian";

listStudents.Add(student);

}

Teacher classTeacher = **new** Teacher();

classTeacher.Name = "Mr.RadheShyam";

classTeacher.DateOfJoining = Convert.ToDateTime("2010-09-11 00:00:00");

classTeacher.ContactAddress = "Some address";

CClass class1A = **new** CClass();

class1A.Name = "1A";

class1A.Students = listStudents;

class1A.Subjects = listSubjects;

class1A.ClassTeacher = classTeacher;

}

}

}

**Step 4.2.7:** Building the project

* From the top menu, choose **Build->Build Solution.**
* If any compile errors are shown, fix them as required.

**Step 4.2.8:** Publishing and running the project

* From the top menu, select **Debug->Start Without Debugging.**
* This will execute the program in a console window. Since there is no output, nothing will be displayed.

**Step 4.2.9:** Pushing the code to your GitHub repositories

Open your command prompt and navigate to the folder where you have created your files.

cd <folder path>

Initialize your repository using the following command:

git init

Add all the files to your git repository using the following command:

git add .

Commit the changes using the following command:

git commit -m “Changes have been committed.”

Push the files to the folder you created initially using the following command:

git push -u origin master