Ejercicio: Control de notas de alumnos. 3 evaluaciones + 1 actitudinal. Mostrar la condición del alumno. Cuadro estadístico que muestre la nota más alta, más baja, total de aprobados, total de desaprobados y la suma de promedios.

Implementar una clase “Promedio” (atributos: nombre, sus 3 evaluaciones y su actitudinal)

Alumno de pre grado ➔ Con porcentaje de cada nota

Alumno regular ➔ promedio ponderado

Clase Promedio:

```
class Promedio
{
    0 references
    public string estudiante { get; set; }

    1 reference
    public int evaluacion1 { get; set; }

    1 reference
    public int evaluacion2 { get; set; }

    1 reference
    public int evaluacion3 { get; set; }

    1 reference
    public int actitudinal { get; set; }

    // Cálculo de promedio para alumno regular
    1 reference
    public virtual double calculaPromedio()
    {
        return (evaluacion1 + evaluacion2 + evaluacion3 + actitudinal) / 4;
    }
```

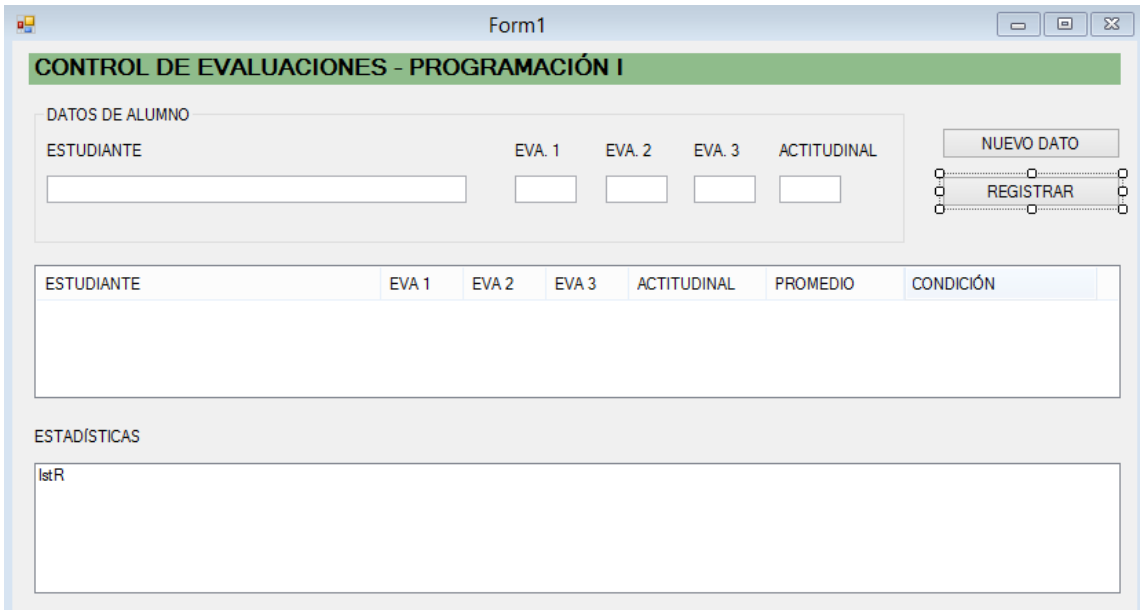
```
// Método para la condición del estudiante
0 references
public string determinarCondicion()
{
    return (calculaPromedio() < 12.5) ? "DESAPROBADO" : "APROBADO";
}
```

Clase derivada PromedioProgramacion:

```
class Promedioprogramacion : Promedio // Hereda de "Promedio"
{
    // Método para el cálculo de promedio de programación (Sobreescrito)
    2 references
    public override double calculaPromedio()
    {
        return evaluacion1 * 0.15 + evaluacion2 * 0.3 + evaluacion3 * 0.5 + actitudinal * 0.05;
    }

}
```

Formulario:




```
private void btnRegistrar_Click(object sender, EventArgs e)
{
    // Objeto de la clase PromedioProgramacion
    Promedioprogramacion objP = new Promedioprogramacion();

    // En viando valores a la clase
    objP.estudiante = txtEstudiante.Text;
    objP.evaluacion1 = int.Parse(txtEva1.Text);
    objP.evaluacion2 = int.Parse(txtEva2.Text);
    objP.evaluacion3 = int.Parse(txtEva3.Text);
    objP.actitudinal= int.Parse(txtActitudinal.Text);

    // Imprimiendo a la lista
    ListViewItem fila = new ListViewItem(objP.estudiante);
    fila.SubItems.Add(objP.evaluacion1.ToString("0.00"));
    fila.SubItems.Add(objP.evaluacion2.ToString("0.00"));
    fila.SubItems.Add(objP.evaluacion3.ToString("0.00"));
    fila.SubItems.Add(objP.actitudinal.ToString("0.00"));
    fila.SubItems.Add(objP.calculaPromedio().ToString("0.00"));
    fila.SubItems.Add(objP.determinarCondicion());
    lvEvaluaciones.Items.Add(fila);
```

```
    // Invocar al método para imprimir las estadísticas
    estadisticas();
}
```

```
// Método para la suma de promedios
1 reference
public double sumaPromedios()
{
    double suma = 0;
    for (int i = 0; i < lvEvaluaciones.Items.Count; i++)
    {
        suma += double.Parse(lvEvaluaciones.Items[i].SubItems[5].Text);
    }
    return suma;
}

// Método para el promedio más alto
1 reference
public double promedioMasAlto()
{
    double mayor = 0;
    for (int i = 0; i < lvEvaluaciones.Items.Count; i++)
    {
        double promedio = double.Parse(lvEvaluaciones.Items[i].SubItems[5].Text);
        mayor = (promedio > mayor) ? promedio : mayor;
    }
    return mayor;
}
```

```
// Método para el promedio más bajo
1 reference
public double promedioMasBajo()
{
    double menor = int.MaxValue; // Un máximo, para hacer la compración
    for (int i = 0; i < lvEvaluaciones.Items.Count; i++)
    {
        double promedio = double.Parse(lvEvaluaciones.Items[i].SubItems[5].Text);
        menor = (promedio < menor) ? promedio : menor;
    }
    return menor;
}
```

```
// Método para hallar la cantidad de aprobados
1 reference
public int totalAprobados()
{
    int cantidadAprob = 0;

    for (int i = 0; i < lvEvaluaciones.Items.Count; i++)
    {
        if (double.Parse(lvEvaluaciones.Items[i].SubItems[5].Text) > 12.5)
        {
            cantidadAprob += 1;
        }
    }
    return cantidadAprob;
}
```

```
// Método para hallar la cantidad de desaprobados
1 reference
public int totalDesaprobados()
{
    int cantidadDesap = 0;

    for (int i = 0; i < lvEvaluaciones.Items.Count; i++)
    {
        if (double.Parse(lvEvaluaciones.Items[i].SubItems[5].Text) <= 12.5)
        {
            cantidadDesap += 1;
        }
    }
    return cantidadDesap;
}
```

```
// Método para imprimir las estadísticas
1 reference
private void estadisticas()
{
    lstR.Items.Clear();
    lstR.Items.Add("Suma de promedios: " + sumaPromedios().ToString("0.00"));
    lstR.Items.Add("Promedio más alto: " + promedioMasAlto().ToString("0.00"));
    lstR.Items.Add("Promedios ás bajo: " + promedioMasBajo().ToString("0.00"));
    lstR.Items.Add("Total de aprobados: " + totalAprobados().ToString("0.00"));
    lstR.Items.Add("Total de desaprobados: " + totalDesaprobados().ToString("0.00"));
}
```