Universidad Israel

DEPARTAMENTO: Ciencias de la Ingeniería
CARRERA: Sistemas de Información
CURSO: Séptimo **PARALELO:** "A"
ASIGNATURA: Plataformas de Desarrollo 1

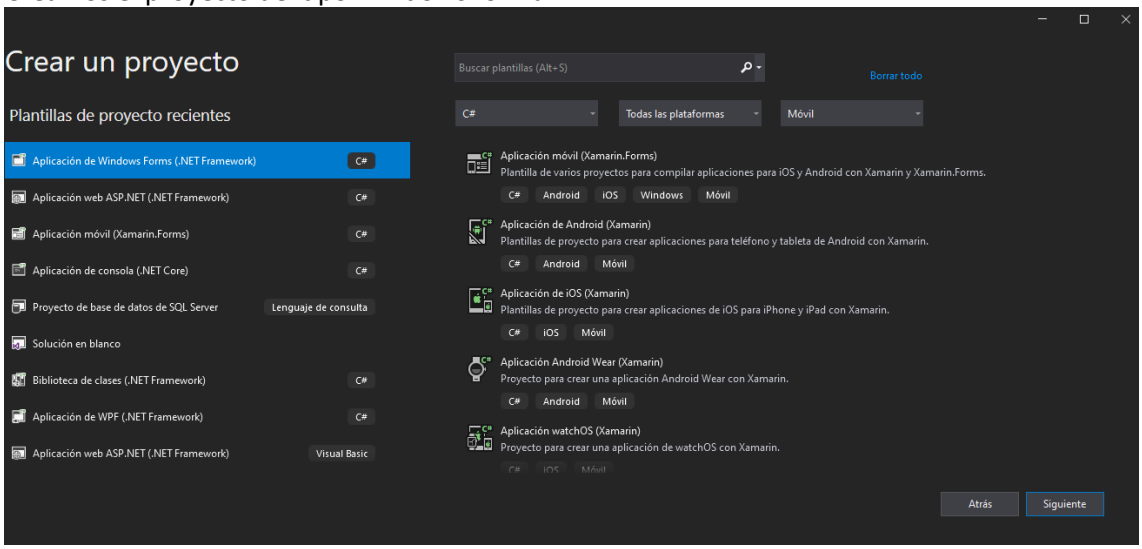
PROFESOR: Mg. Luis Fernando Aguas B.
ESTUDIANTE: (Marco Antonio Ayala Lituma)
DESCRIPCIÓN: Deber 1-S6

TEMA:

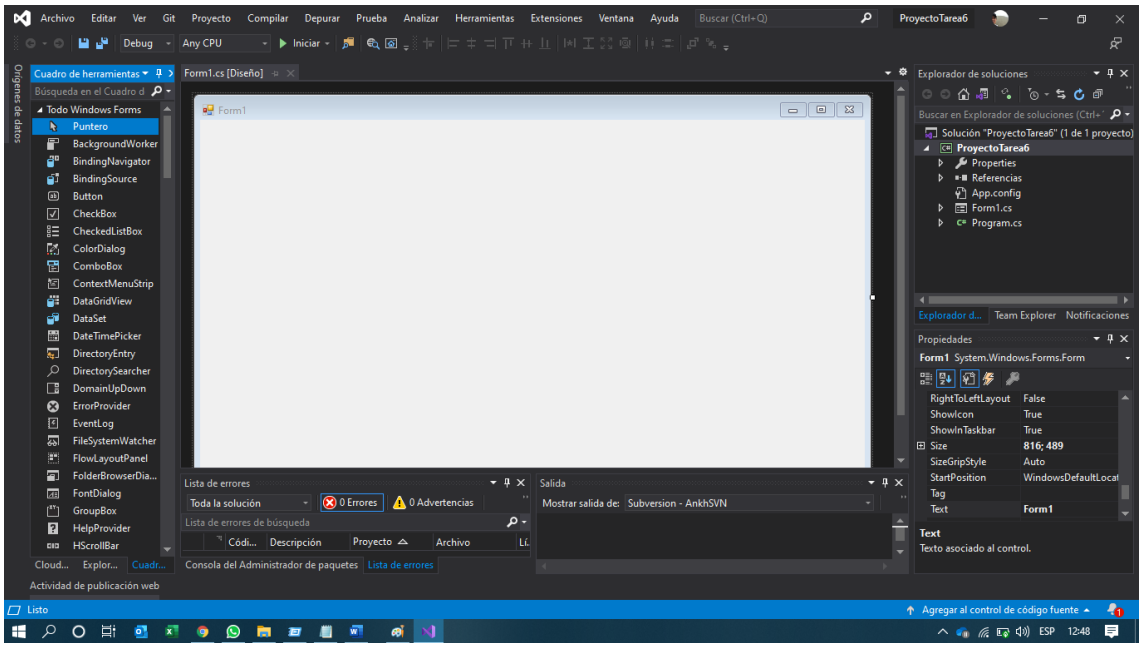
Realice un programa en el cual implemente un sistema de registro de asistencias para los ayudantes y becarios del LTIC, el sistema debe presentar un resumen de asistencias semanal, mensual de los estudiantes, adicionalmente se debe ingresar disponibilidad de los estudiantes, se debe presentar el horario global de todos los ayudantes y becarios, debe realizar gráficos estadísticos de la asistencia por estudiante.

DESARROLLO:

Creamos el proyecto del tipo Windows forma

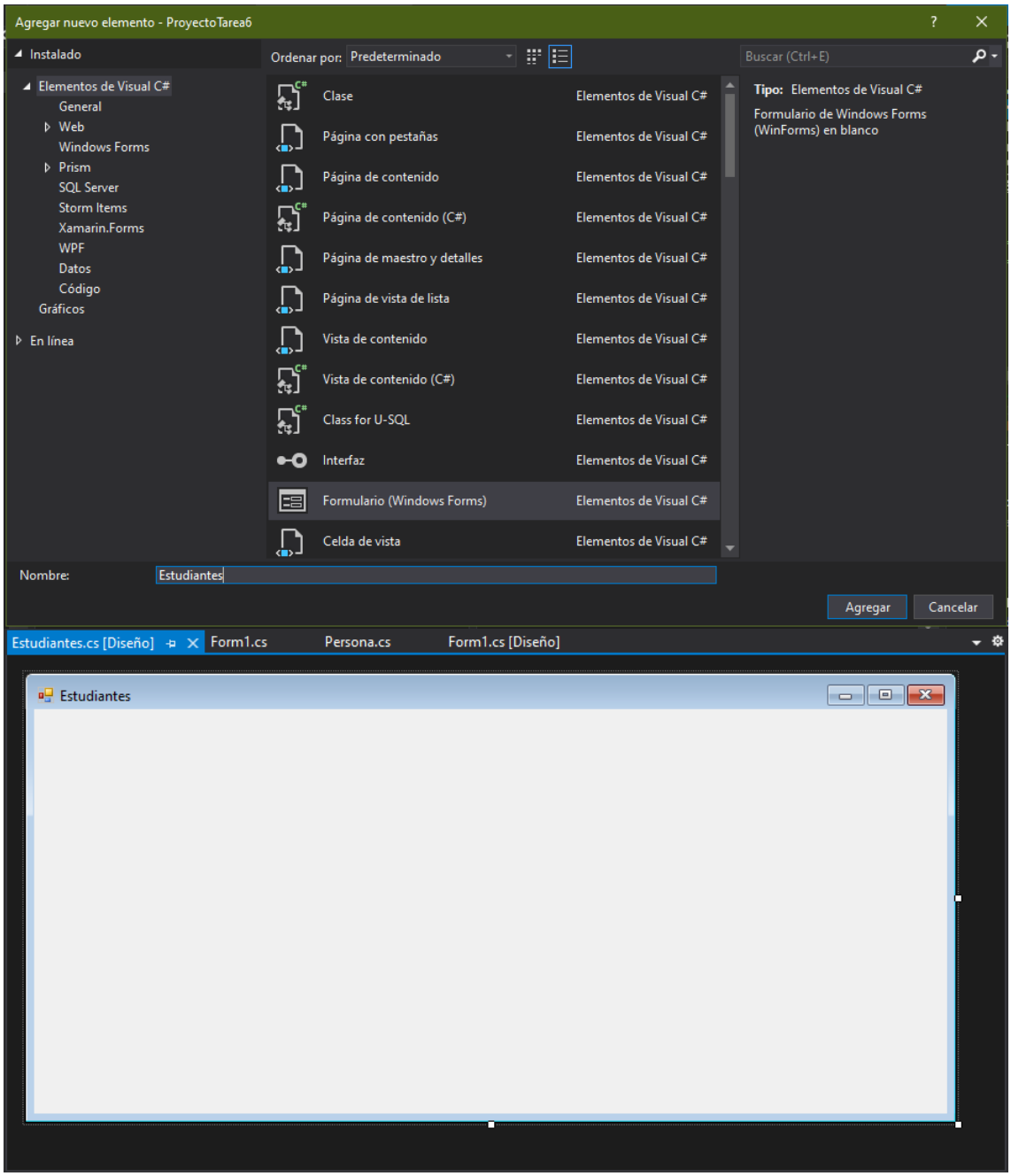


El formulario Inicial se visualizara de esta manera

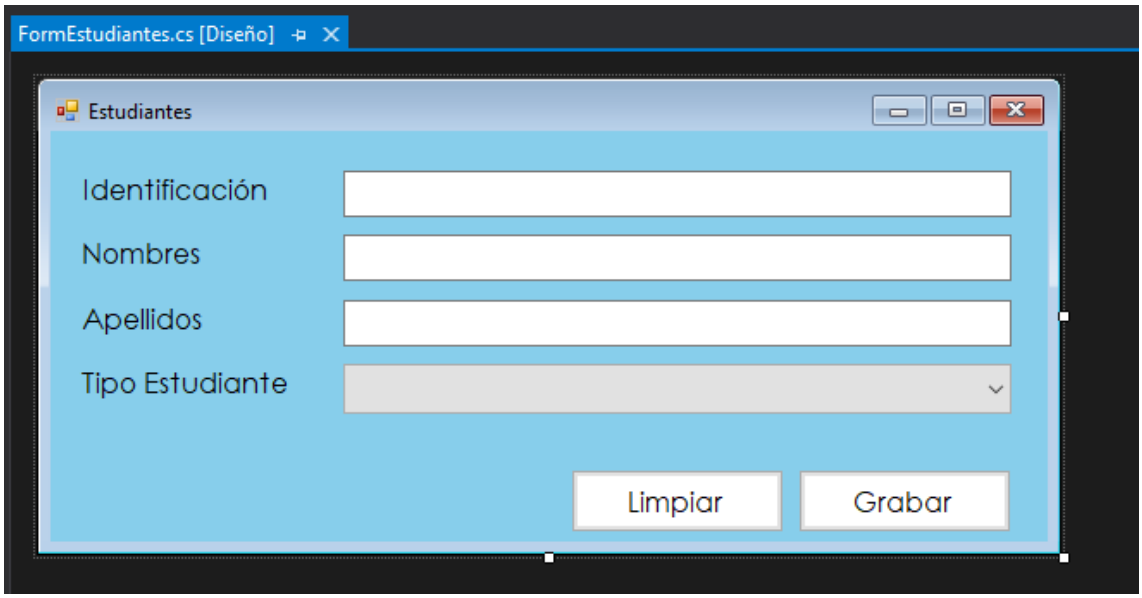


Creamos el siguiente menú

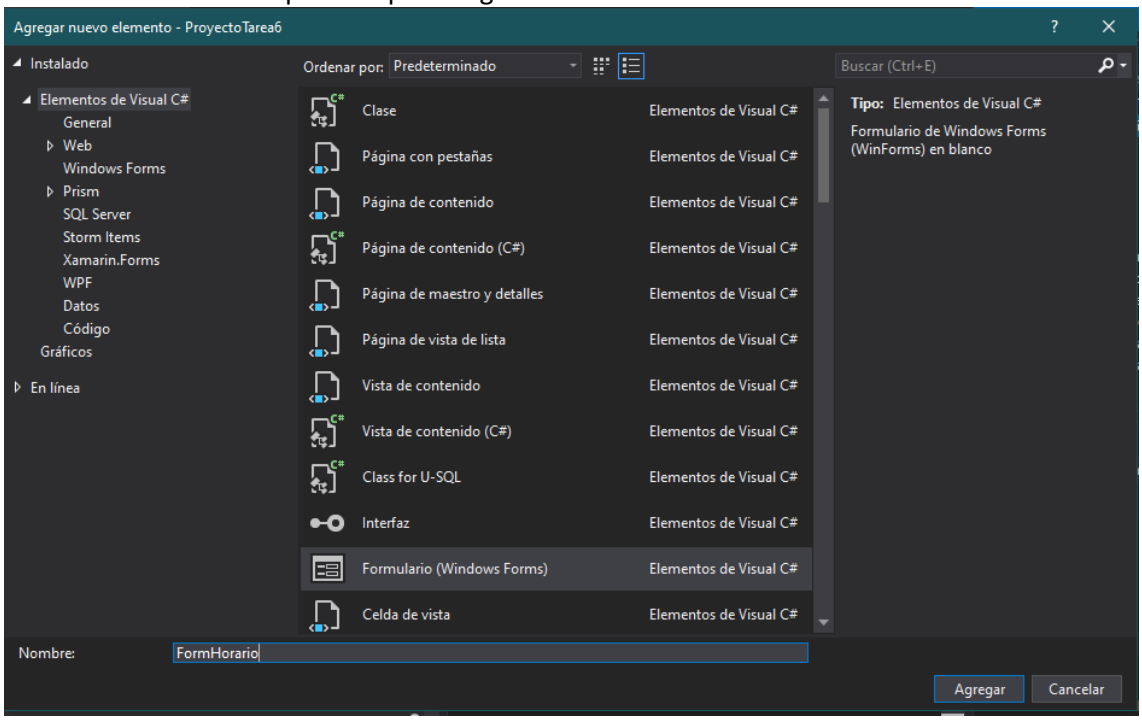
Creamos el Formulario Estudiante



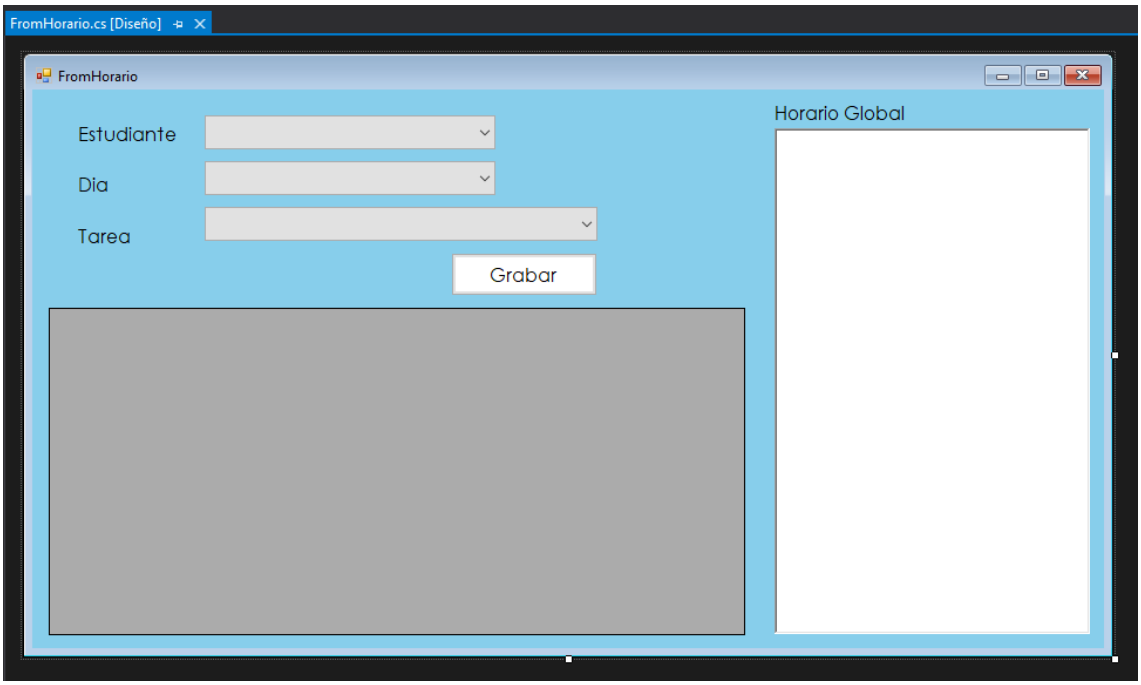
Vamos a diseñar una estructira similiar a esta



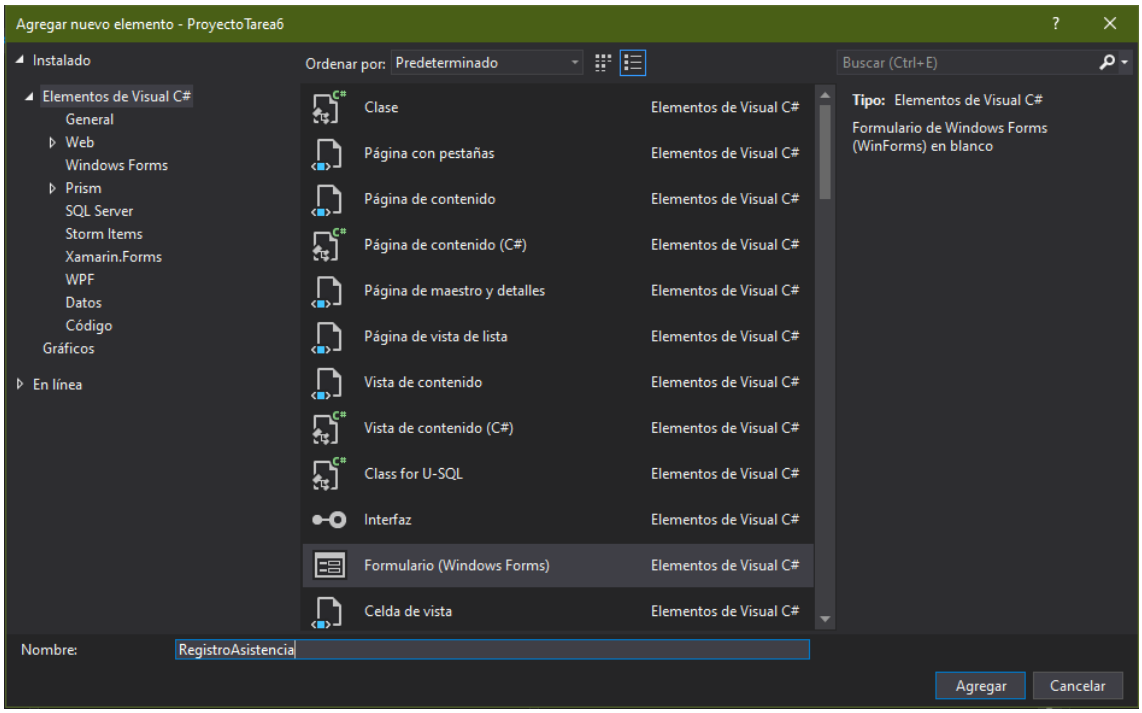
Ahora vamos a crear la pantalla para asignar el horario

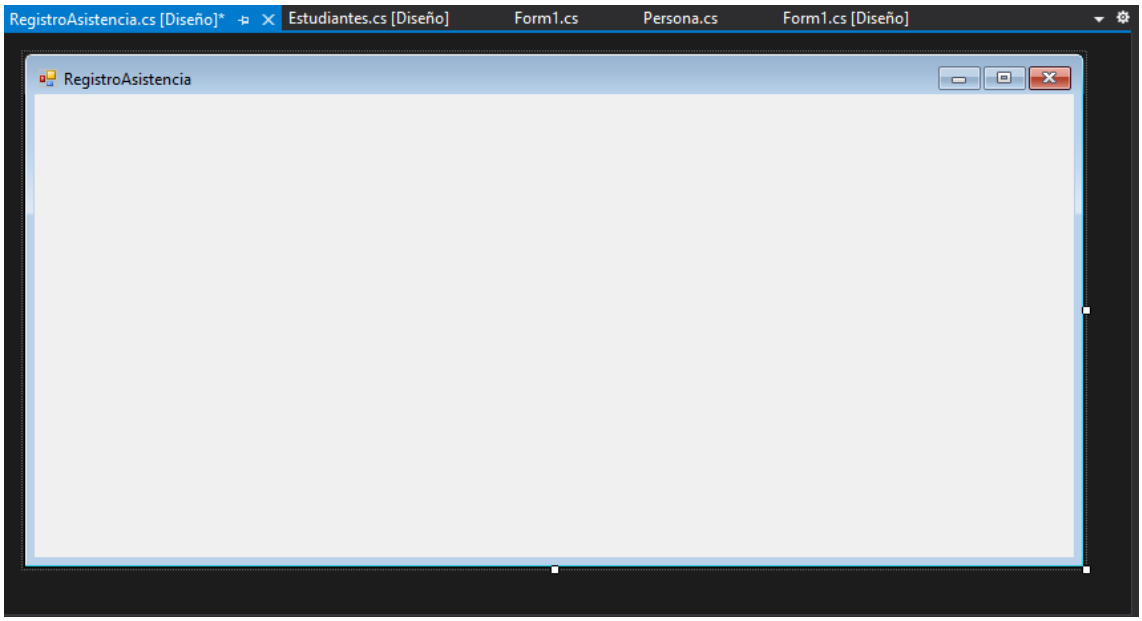


Diseñamos los siguientes controles

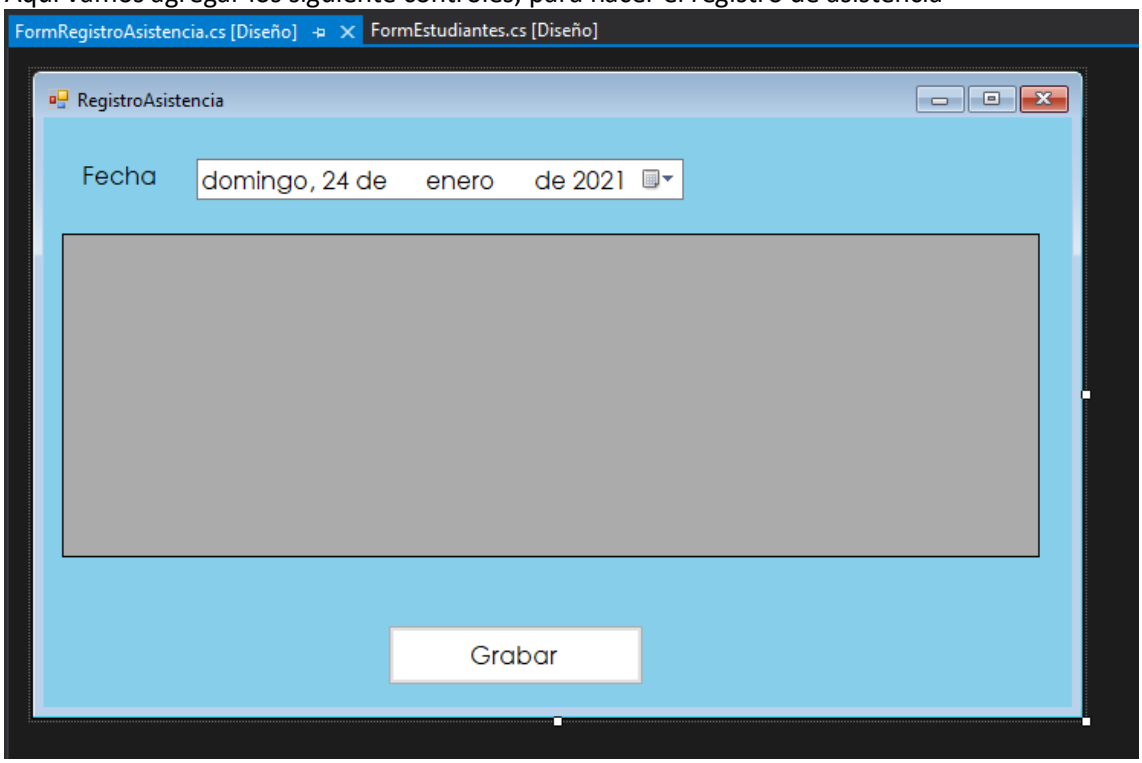


Ahora creamos la pantalla para registro de asistencia

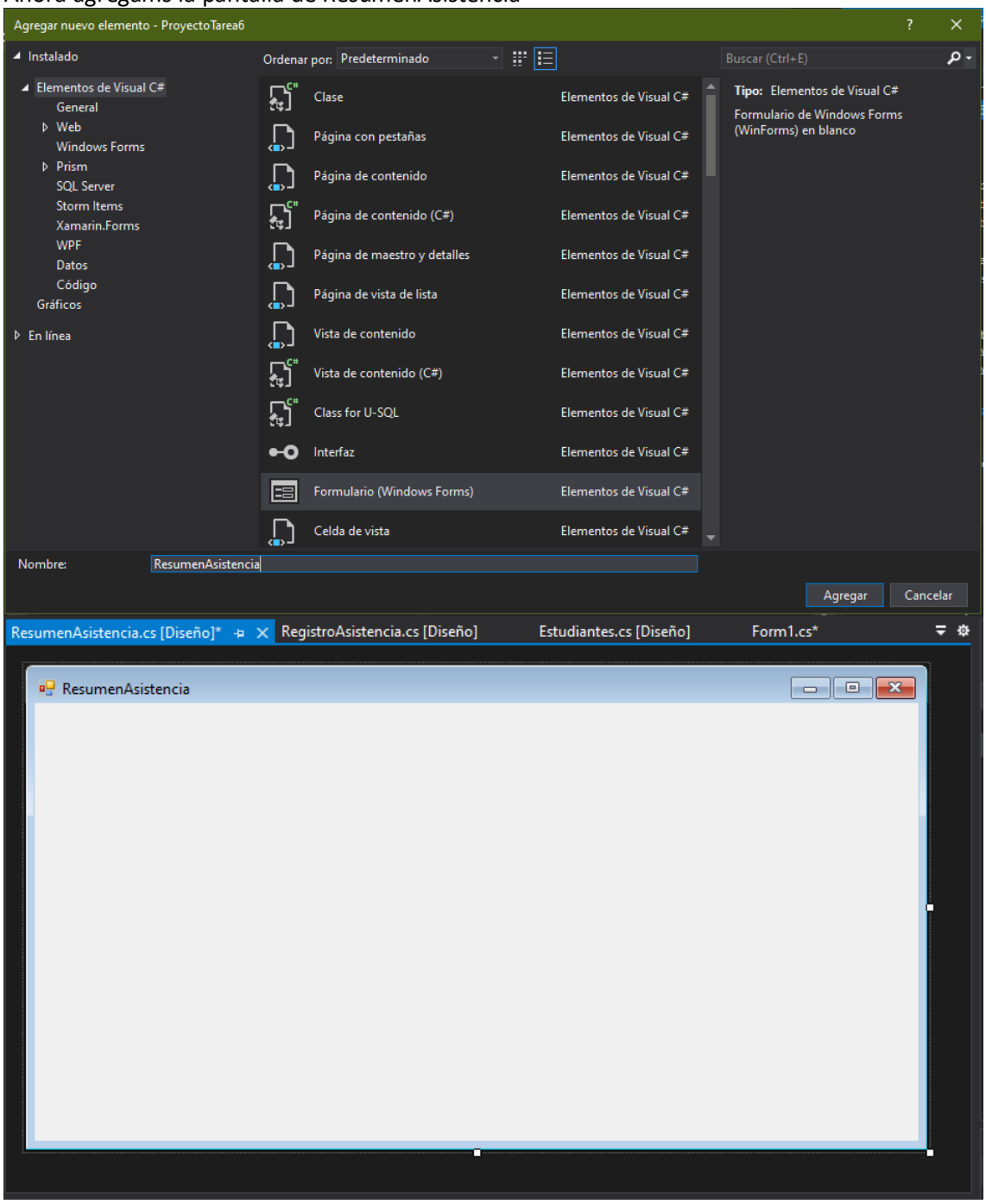


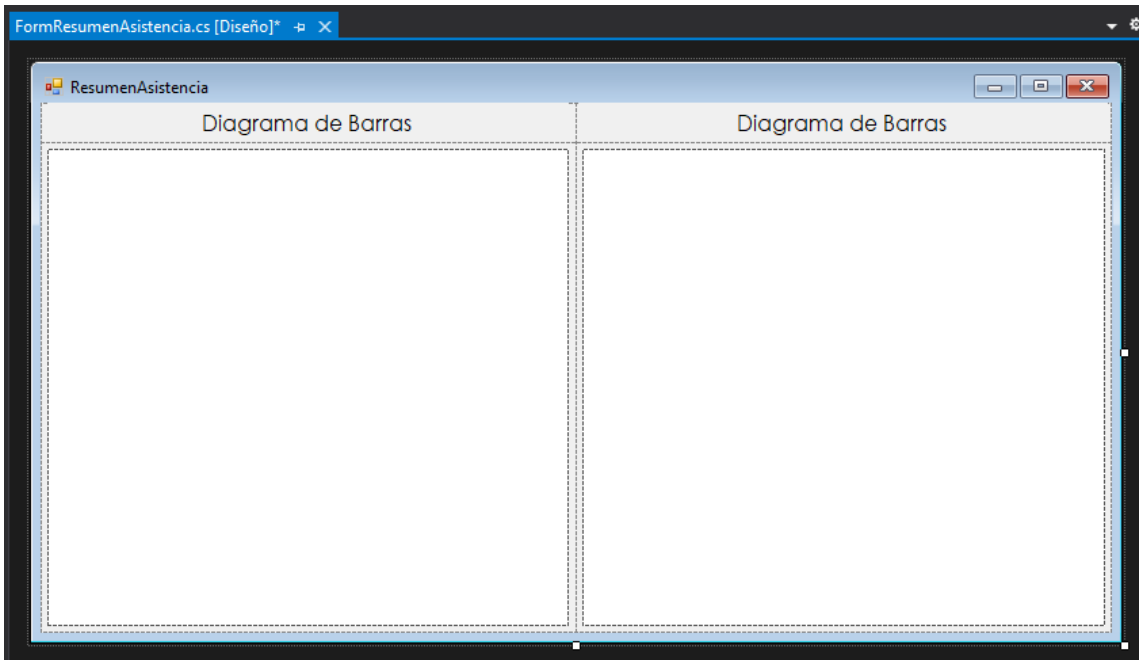


Aquí vamos agregar los siguiente controles, para hacer el registro de asistencia



Ahora agregams la pantalla de ResumenAsistencia





Ahora vamos a agregar el siguiente código
En el Programa vamos a implementar nuevas variables globales

```
15 referencias | 0 cambios | 0 autores, 0 cambios
static class Program
{
    public static List<Horario> ListaHorarios = new List<Horario>();
    public static List<Asistencia> ListaAsistencia = new List<Asistencia>();
    public static List<Estudiante> ListaEstudiantes = new List<Estudiante>();
```

Ahora vamos agregar las siguientes clases

```
3 referencias | 0 cambios | 0 autores, 0 cambios
public class Asistencia
{
    3 referencias | 0 cambios | 0 autores, 0 cambios
    public string fecha { get; set; }
    1 referencia | 0 cambios | 0 autores, 0 cambios
    public string tarea { get; set; }
    2 referencias | 0 cambios | 0 autores, 0 cambios
    public string identificacion { get; set; }
    5 referencias | 0 cambios | 0 autores, 0 cambios
    public bool estadoAsistencia { get; set; }
}
```

```
2 referencias | 0 cambios | 0 autores, 0 cambios
public class Estudiante
{
    1 referencia | 0 cambios | 0 autores, 0 cambios
    public string identificacion { get; set; }
    1 referencia | 0 cambios | 0 autores, 0 cambios
    public string nombres { get; set; }
    1 referencia | 0 cambios | 0 autores, 0 cambios
    public string Apellidos { get; set; }
    1 referencia | 0 cambios | 0 autores, 0 cambios
    public string tipoEstudiante { get; set; }
}
```

```
public class Horario
{

  public string identificacionEstudiante { get; set; }

  public string nombreEstudiante { get; set; }
  3 referencias | 0 cambios | 0 autores, 0 cambios
  public string dia { get; set; }
  3 referencias | 0 cambios | 0 autores, 0 cambios
  public string tarea { get; set; }



}
```

```
public class TipoActividad
{
  6 referencias | 0 cambios | 0 autores, 0 cambios
  public string nombre { get; set; }

  5 referencias | 0 cambios | 0 autores, 0 cambios
  public TipoActividad()
  {
    nombre = string.Empty;
  }
  1 referencia | 0 cambios | 0 autores, 0 cambios
  public static List<TipoActividad> getTipos()
  {
    List<TipoActividad> lista = new List<TipoActividad>();
    lista.Add(new TipoActividad { nombre = "Seleccione" });
    lista.Add(new TipoActividad { nombre = "Asesorias" });
    lista.Add(new TipoActividad { nombre = "Curso de Base de Datos" });
    lista.Add(new TipoActividad { nombre = "Desarrollo de Aplicaciones Moviles" });
    lista.Add(new TipoActividad { nombre = "Formateo de Computadores" });

    return lista;
  }
}
```

No se encontraron problemas.

```
public class TipoEstudiante
{
  4 referencias | 0 cambios | 0 autores, 0 cambios
  public string nombre { get; set; }

  3 referencias | 0 cambios | 0 autores, 0 cambios
  public TipoEstudiante()
  {
    nombre = string.Empty;
  }
  1 referencia | 0 cambios | 0 autores, 0 cambios
  public static List<TipoEstudiante> getTipos()
  {
    List<TipoEstudiante> lista = new List<TipoEstudiante>();
    lista.Add(new TipoEstudiante { nombre = "Seleccione" });
    lista.Add(new TipoEstudiante { nombre = "Ayudante" });
    lista.Add(new TipoEstudiante { nombre = "Becario" });
    return lista;
  }
}
```

Agregar el siguiente código en la pantalla estudiantes

```
public FormEstudiantes()
  {
```

```
    InitializeComponent();
    BindingSource bs = new BindingSource();
    cmbTipoEstudiante.DataSource = TipoEstudiante.getTipos();
  }

  private void button2_Click(object sender, EventArgs e)
  {
    txtIdentificacion.Text = string.Empty;
    txtNombres.Text = string.Empty;
    txtApelidos.Text = string.Empty;
    cmbTipoEstudiante.SelectedIndex=0;
  }

  private bool Valida()
  {
    if (txtIdentificacion.Text == string.Empty)
    {
      MessageBox.Show(this, "La identificacion esta vacia", "Advertencia",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
      txtIdentificacion.Focus();
      return false;
    }
    else if (txtNombres.Text == string.Empty)
    {
      MessageBox.Show(this, "El nombre esta vacia", "Advertencia", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
      txtNombres.Focus();
      return false;
    }
    else if (txtApelidos.Text == string.Empty)
    {
      MessageBox.Show(this, "El Apellido esta vacia", "Advertencia", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
      txtApelidos.Focus();
      return false;
    }
    else if (cmbTipoEstudiante.Text == "Seleccione")
    {
      MessageBox.Show(this, "Seleccione el tipo de estudiante", "Advertencia",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
      cmbTipoEstudiante.Focus();
      return false;
    }
    else
    {
      return true;
    }

  }
  private void button1_Click(object sender, EventArgs e)
  {
```

```
    if (!Valida())
    {
      return;
    }
    int Id = Program.ListaEstudiantes.Count() + 1;
    Program.ListaEstudiantes.Add(new Estudiante
    {
      identificacion = txtIdentificacion.Text.Trim(),
      nombres = txtNombres.Text.Trim(),
      Apellidos = txtApelidos.Text.Trim(),
      tipoEstudiante = cmbTipoEstudiante.SelectedText
    });

    MessageBox.Show(this,"Estudiante
Crado","Mensaje",MessageBoxButtons.OK,MessageBoxIcon.Information);
    button2_Click(sender,e);
  }
```

Ahora el codigo de pantalla de asignación de horarios

```
public FromHorario()
  {
    InitializeComponent();
    cmbEstudiante.DataSource = Program.ListaEstudiantes;
    cmbEstudiante.DisplayMember = "Nombres";
    cmbEstudiante.ValueMember = "Identificacion";
    cmbTarea.DataSource = Modelos.TipoActividad.getTipos();
    cmbTarea.DisplayMember = "nombre";
    cmbTarea.ValueMember = "nombre";
  }
  public void CargarInformacion(string Identificacion)
  {
    richTextBox1.Clear();
    foreach (var item in Program.ListaHorarios)
    {
      richTextBox1.AppendText("Estudiante:" + item.nombreEstudiante + " Día:" + item.dia + "
Tarea:" + item.tarea + "\n");
    }
    dataGridView2.DataSource = Program.ListaHorarios.Where(x =>
x.identificacionEstudiante.Equals(Identificacion)).ToList();
    dataGridView2.Refresh();
  }

  private void cmbEstudiante_SelectionChangeCommitted(object sender, EventArgs e)
  {
    CargarInformacion(cmbEstudiante.SelectedValue.ToString());
  }

  private void btnGrabar_Click(object sender, EventArgs e)
  {
    Program.ListaHorarios.Add(new Modelos.Horario
    {
      dia = cmbDia.Text,
```

```
      identificacionEstudiante = cmbEstudiante.SelectedValue.ToString(),
      nombreEstudiante = cmbEstudiante.Text,
      tarea = cmbTarea.Text
    });
    CargarInformacion(cmbEstudiante.SelectedValue.ToString());
  }
```

Este codigo es para registro de asistencia

```
private string diaSeleccionado = string.Empty;
  public FormRegistroAsistencia()
  {
    InitializeComponent();

  }

  private void dateTimePicker1_ValueChanged(object sender, EventArgs e)
  {
    diaSeleccionado = getNombreDia(dateTimePicker1.Value.DayOfWeek.ToString());// ==
DayOfWeek.Monday
    foreach (var item in Program.ListaHorarios)
    {
      if (item.dia.Equals(diaSeleccionado))
      {
        if (Program.ListaAsistencia.Where(x =>
x.fecha.Equals(dateTimePicker1.Value.ToString("dd-MM-yyyy")) &&
x.identificacion.Equals(item.identificacionEstudiante)).Count() == 0)
        {
          Program.ListaAsistencia.Add(new Modelos.Asistencia
          {
            estadoAsistencia = false,
            fecha = dateTimePicker1.Value.ToString("dd-MM-yyyy"),
            identificacion = item.identificacionEstudiante,
            tarea = item.tarea
          });
        }

      }
    }
    dataGridView1.DataSource =
Program.ListaAsistencia.Where(x=>x.fecha.Equals(dateTimePicker1.Value.ToString("dd-MM-
yyyy"))).ToList();
    dataGridView1.Refresh();
  }


  public string getNombreDia(string diaIngles)
  {
    switch (diaIngles)
    {
      case "Monday":
        return "Lunes";
      case "Tuesday":
```

```
        return "Martes";
      case "Wednesday":
        return "Miercoles";
      case "Thursday":
        return "Jueves";
      case "Friday":
        return "Viernes";
      case "Saturday":
        return "Sabado";
      case "Sunday":
        return "Donigo";
      default:
        return diaIngles;
    }
  }
```

Finalmente el codigo de resumen de asistencia

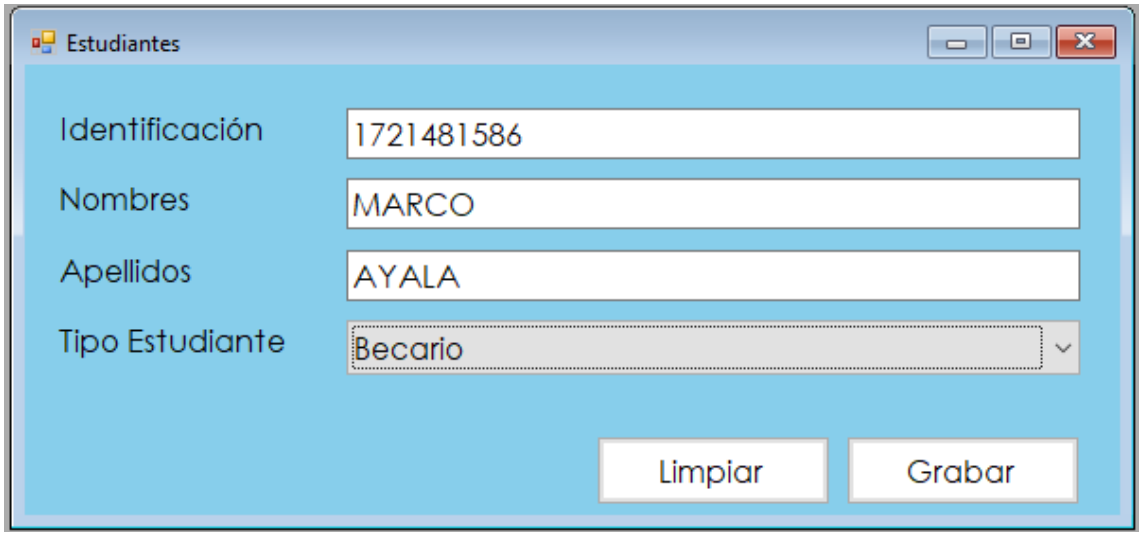
```
int i = 0;
  Pen f = new Pen(Color.Red);
  Pen f1 = new Pen(Color.Blue);
  Pen f2 = new Pen(Color.Beige);
  Font tipo1 = new Font("Arial", 10);
  Font tipo2 = new Font("Comic Sans MS", 9);
  bool band = false;
  public FormResumenAsistencia()
  {
    InitializeComponent();
  }

  private void panel1_Paint(object sender, PaintEventArgs e)
  {
    if (band)
    {
      graficoBarras(panel1.CreateGraphics(), Program.ListaAsistencia, 100, 300);
    }
  }

  private void panel2_Paint(object sender, PaintEventArgs e)
  {
    if (band)
    {
      graficoPastel(panel2.CreateGraphics(), 50, 125);
    }
  }

  private void FormResumenAsistencia_Load(object sender, EventArgs e)
  {
    band = true;
    this.Refresh();
  }
  public void graficoBarras(Graphics t, List<Asistencia> A, int x, int y)
```

```
  {
    int contS = 0, contC = 0, a = 20, ancho = 40;
    for (int i = 0; i < A.Count; i++)
    {
      if (A[i].estadoAsistencia.Equals(true)) contS++;
      if (!A[i].estadoAsistencia.Equals(true)) contC++;
    }
    t.DrawLine(f1, x, y, x + 300, y); t.DrawLine(f1, x, y, x, y - 250);
    t.DrawString("Registro Asistencia", tipo2, Brushes.Black, x + 300, y);
    t.DrawString("Contador", tipo2, Brushes.Black, x - 100, y - 250);
    t.DrawRectangle(f1, x + 250, y - 190, 160, 100); t.DrawString("Leyenda", tipo2,
Brushes.Black, x + 270, y - 170);
    t.FillRectangle(Brushes.SkyBlue, x + 270, y - 150, ancho, ancho / 2);
    t.FillRectangle(Brushes.Purple, x + 270, y - 120, ancho, ancho / 2);
    t.DrawRectangle(f1, x + 270, y - 150, ancho, ancho / 2);
    t.DrawRectangle(f1, x + 270, y - 120, ancho, ancho / 2);
    t.DrawString("Asistencia", tipo2, Brushes.Black, x + 320, y - 150);
    t.DrawString("Falta", tipo2, Brushes.Black, x + 320, y - 120);
    t.DrawRectangle(f1, x + 270, y - 150, ancho, ancho / 2);
    t.DrawRectangle(f1, x + 270, y - 120, ancho, ancho / 2);
    t.DrawString("Asistencia", tipo2, Brushes.Black, x + 320, y - 150);
    t.DrawString("Falta", tipo2, Brushes.Black, x + 320, y - 120);

    t.FillRectangle(Brushes.SkyBlue, x + 50, y - (contS * a), ancho, contS * a);
    t.DrawRectangle(f1, x + 50, y - (contS * a), ancho, contS * a);
    t.DrawLine(f1, x, y - (contS * a), x + 50 + ancho, y - (contS * a));

    t.DrawString("Asistencia", tipo2, Brushes.Black, x + 50, y + 30);
    t.DrawString("" + contS, tipo2, Brushes.Black, x - 20, y - (contS * a));
    t.FillRectangle(Brushes.Purple, x + 140, y - (contC * a), ancho, contC * a);
    t.DrawRectangle(f1, x + 140, y - (contC * a), ancho, contC * a);
    t.DrawLine(f1, x + 50 + ancho, y - (contC * a), x + 140 + ancho, y - (contC * a));
    t.DrawString("Falta", tipo2, Brushes.Black, x + 140, y + 30);
    t.DrawString("" + contC, tipo2, Brushes.Black, x - 20, y - (contC * a));
  }
  public void graficoPastel(Graphics g, int x, int y)
  {
    int ptot = 0; int s = 0, ns = 0;
    for (int i = 0; i < Program.ListaAsistencia.Count; i++)
    {
      if (Program.ListaAsistencia[i].estadoAsistencia.Equals(true)) s++;
      if (!Program.ListaAsistencia[i].estadoAsistencia.Equals(true)) ns++;
    }
    ptot = Program.ListaAsistencia.Count; if (ptot != 0)
    {
      g.DrawRectangle(f1, x + 200, y + 20, 200, 100);
      g.DrawString("Leyenda:", tipo2, Brushes.Black, x + 210, y + 40);
      g.DrawString("Asistencia " + (double)(s * 100 / ptot) + " %", tipo2, Brushes.Black, x + 230, y
+ 60);
      g.DrawString("Falta " + (double)(ns * 100 / ptot) + " %", tipo2, Brushes.Black, x + 230, y +
80);
```

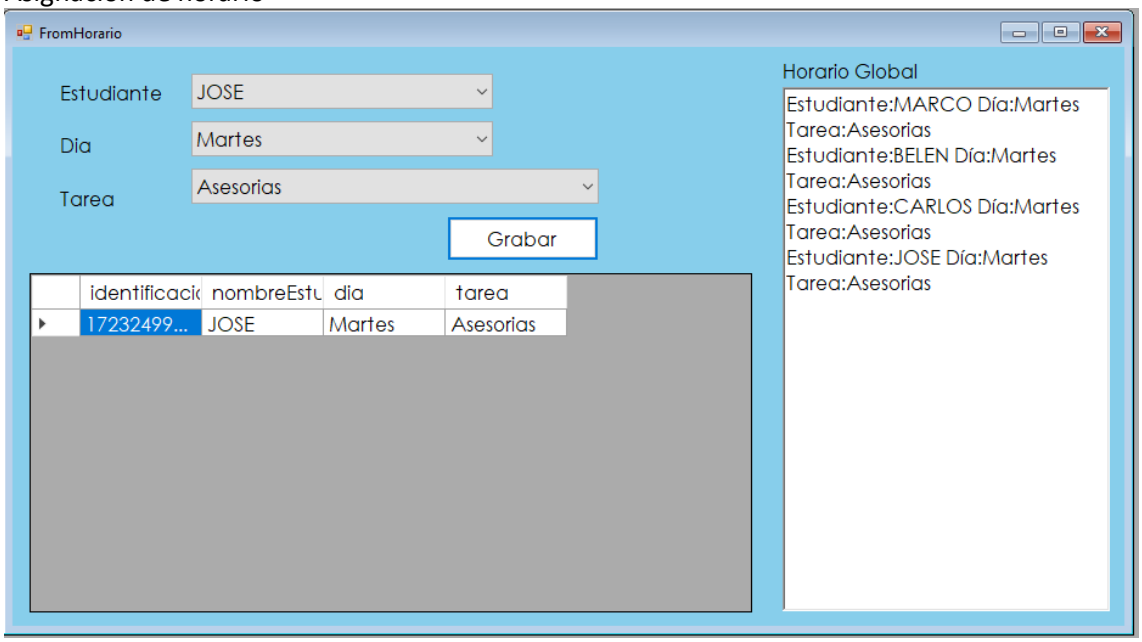
```
        g.DrawRectangle(f1, x + 210, y + 60, 10, 10);
        g.DrawRectangle(f1, x + 210, y + 80, 10, 10);
        g.FillEllipse(Brushes.Blue, x - 3, y - 3, 156, 156);

        g.FillPie(Brushes.SkyBlue, x, y, 150, 150, 0, (int)(s * 360 / ptot));
        g.FillRectangle(Brushes.SkyBlue, x + 210, y + 60, 10, 10);
        g.FillPie(Brushes.Purple, x, y, 150, 150, (int)(s * 360 / ptot), (int)(ns * 360 / ptot));
        g.FillRectangle(Brushes.Purple, x + 210, y + 80, 10, 10);
    }
}
```

Ejecución del Programa

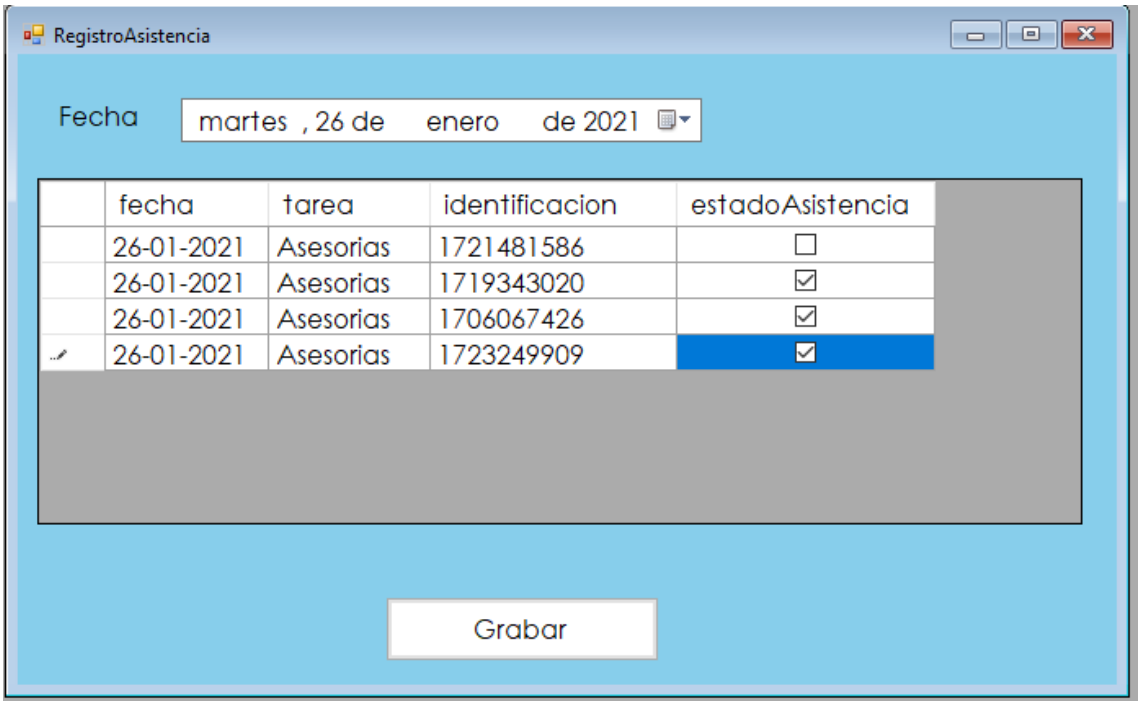
Creación de Estudiantes



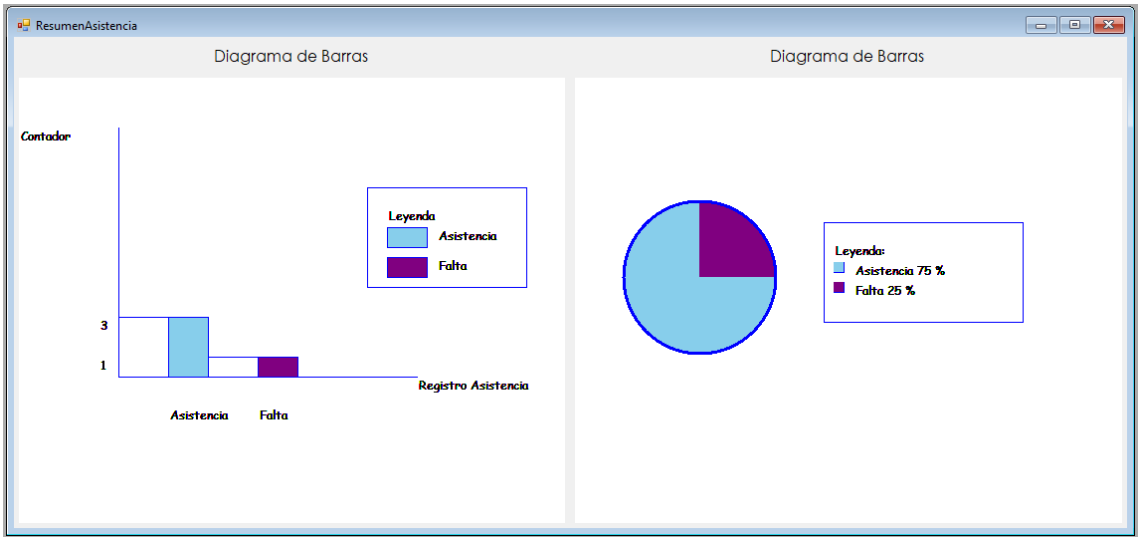
Asignacion de horario



Registro de asistencia



Resumen Estadístico



BIBLIOGRAFÍA:

Titulo	Autor	Año	Editorial	URL/Observacion
Java 2 Lenguaje y Aplicaciones	Ceballos Sierra F.J.	2015	RA-MA-Editorial.	https://elibro.net/es/lc/uisrael/titulos/624
Empezar a programar usando Java (3ra. Ed)	Prieto Saez, N y Casanova Faus A.	2016	Edittorial de la Univerdad Politecnica de Valencia	https://elibro.net/es/lc/uisrael/titulos/574