

Programación II

Guía 3: Modelado de clases en UML: Asociación.

Objetivos

- Conocer la relación de asociación entre clases.
- Codificar clases utilizando diagramas UML con relación de asociación.

¿Qué es Asociación?:

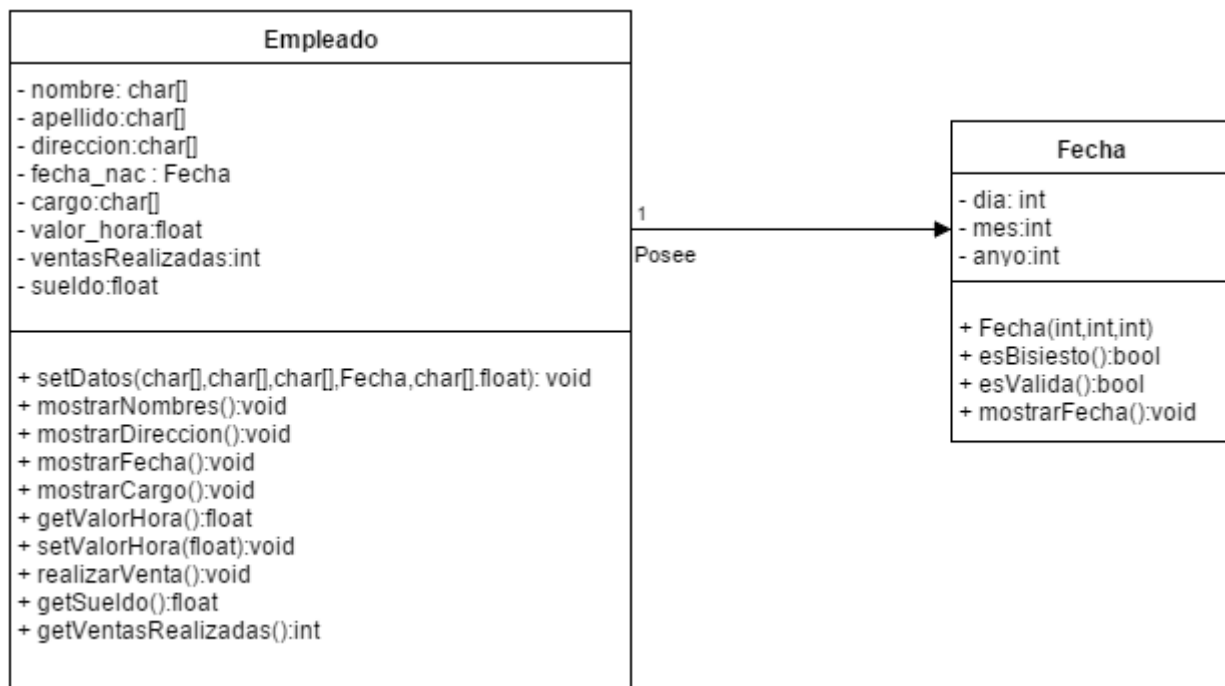
Es una relación de estructura entre clases, es decir, una entidad se construye a partir de otra u otras.

Aunque este tipo de relación es más fuerte que la **Dependencia** es más débil que la **Agregación**, ya que el tiempo de vida de un objeto no depende de otro.

Representación UML:

Se representa con una flecha continua que parte desde una clase y apunta a otra. El sentido de la flecha nos indica la clase que se compone (base de la flecha) y sus componentes (punta de la flecha).

Ejemplo:



Programación II

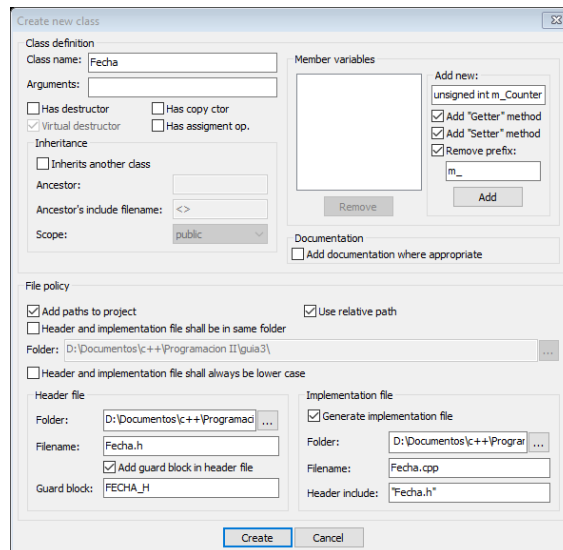
Del cual se puede identificar:

1. Tenemos una clase Empleado que posee un atributo Fecha
2. Tenemos una clase Fecha que tiene atributos día, mes, año(año), y ciertos métodos.
3. El empleado necesita una fecha la cual es su fecha de nacimiento.
4. Es necesario validar la fecha.

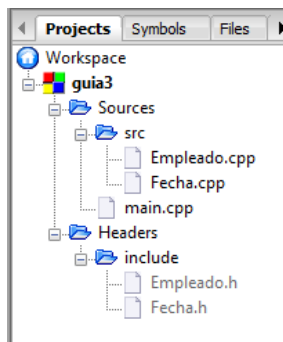
Se puede observar que la clase Fecha es independiente de la clase Empleado pero **todo cambio en la clase Fecha se ve afectado por la clase Empleado**.

Código:

1. Crear un nuevo proyecto en consola usando c++ bajo el nombre “guia3”.
2. Crear dos clases la primera será “Fecha” y la segunda “Empleado” y agregarlas al proyecto.
File->New->Class...



Tendríamos un resultado así:



Programación II

3. Trasladar el contenido del diagrama UML presentado en la página 1 a sus respectivas clases.

Contenido Fecha.h:

```
4  class Fecha
5  {
6      private:
7          int dia;
8          int mes;
9          int anyo;
10     public:
11         Fecha();
12         Fecha(int,int,int);
13         bool esValida();
14         bool esBisiesto();
15         void mostrarFecha();
16     };
```

Contenido Fecha.cpp:

```
#include <iostream>
#include "Fecha.h"

using namespace std;

Fecha::Fecha(){
    //constructor vacio
}

Fecha::Fecha(int d,int m,int a){
    dia = d;
    mes = m;
    anyo = a;
}

bool Fecha::esValida(){
    bool valida=true;
    //completar funcion
    return valida;
}

bool Fecha::esBisiesto(){
    bool bisiesto=false;
    //completar funcion
    return bisiesto;
}
```

Programación II

```

}

void Fecha::mostrarFecha(){
    if(esValida()==true){
        cout << "Fecha: "<<this->dia<<"/"<<this->mes<<"/"<<this->anyo<<endl;
    }else{
        cout << "La fecha proporcionada es invalida ";
    }
}
}

```

Contenido Empleado.h:

```

4  #include "Fecha.h"
5  #define TAM 20
6  class Empleado
7  {
8      private:
9          char nombre[TAM];
10         char apellido[TAM];
11         char direccion[TAM];
12         Fecha fecha_nac;
13         char cargo[TAM];
14         float valor_hora;
15         int ventasRealizadas;
16         float sueldo;
17     public:
18         void setDatos(char[],char[],char[],Fecha,char[]);
19         void mostrarNombres();
20         void mostrarDireccion();
21         void mostrarFecha();
22         void mostrarCargo();
23         float getValorHora();
24         void setValorHora(float);
25         void realizarVenta();
26         float getSuelo();
27         int getVentasRealizadas();
28 };

```

Nota: digitar el nombre de los atributos y de los métodos exactamente igual.

Contenido Empleado.cpp:

```

#include <string.h>
#include <iostream>
#include "Empleado.h"

using namespace std;

void Empleado::setDatos(char nom[],char ape[],char dir[],Fecha f,char ca[]){

```

Programación II

```
strcpy(this->nombre,nom);
strcpy(this->apellido,ape);
strcpy(this->direccion,dir);
this->fecha_nac = f;
strcpy(this->cargo,ca);
this->ventasRealizadas=0;
}
void Empleado::mostrarNombres(){
    cout << "Empleado : "<< this->nombre << " " << this->apellido<<endl;
}
void Empleado::mostrarDireccion(){
    cout << "Direccion : "<< this->direccion<<endl;
}
void Empleado::mostrarFecha(){
    fecha_nac.mostrarFecha();
}
void Empleado::mostrarCargo(){
    cout << "Cargo " << this->cargo<<endl;
}
float Empleado::getValorHora(){
    return this->valor_hora;
}
void Empleado::setValorHora(float valor){
    this->valor_hora = valor;
}
void Empleado::realizarVenta(){
    cout << "Venta realizada con exito!"<<endl;
    this->ventasRealizadas++;
}
float Empleado::getSueldo(){
    sueldo = (this->valor_hora*this->ventasRealizadas);
    return this->sueldo;
}
int Empleado::getVentasRealizadas(){
    return ventasRealizadas;
}
```

Dado el contenido de las clases finalizado procedemos a codificar nuestro main.cpp

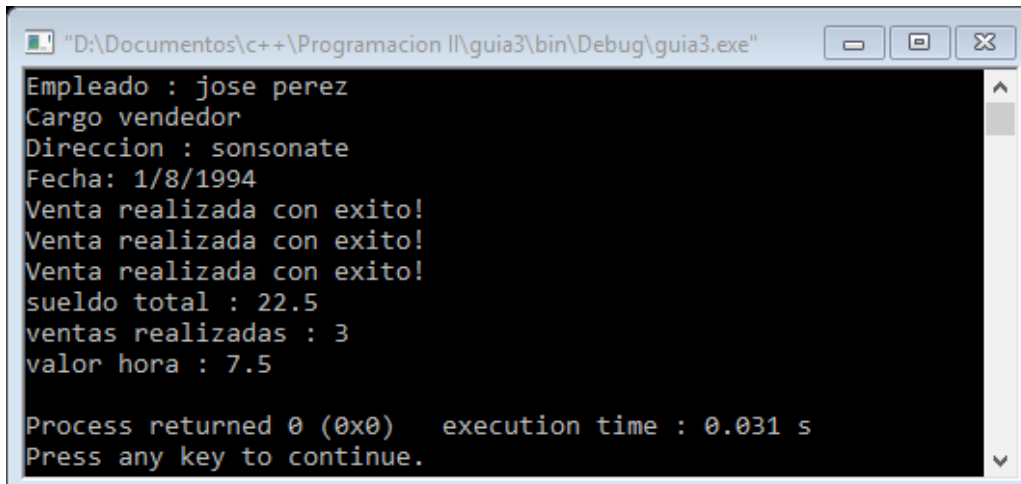
Programación II

4. Contenido del archivo main.cpp y luego compilar.

```
#include <iostream>

using namespace std;
#include "Fecha.h"
#include "Empleado.h"
int main()
{
    Fecha fecha_nac = Fecha(1,8,1994);
    Empleado emp = Empleado();
    emp.setDatos("jose","perez","sonsonate",fecha_nac,"vendedor");
    emp.mostrarNombres();
    emp.mostrarCargo();
    emp.mostrarDireccion();
    emp.mostrarFecha();
    emp.realizarVenta();
    emp.realizarVenta();
    emp.realizarVenta();
    emp.setValorHora(7.5);
    cout << "sueldo total : "<< emp.getSueldo()<<endl;
    cout << "ventas realizadas : "<<emp.getVentasRealizadas()<<endl;
    cout << "valor hora : "<<emp.getValorHora()<<endl;
    return 0;
}
```

Resultado:



```
"D:\Documentos\c++\Programacion II\guia3\bin\Debug\guia3.exe"
Empleado : jose perez
Cargo vendedor
Direccion : sonsonate
Fecha: 1/8/1994
Venta realizada con exito!
Venta realizada con exito!
Venta realizada con exito!
sueldo total : 22.5
ventas realizadas : 3
valor hora : 7.5

Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
```

Programación II

Ejercicios:

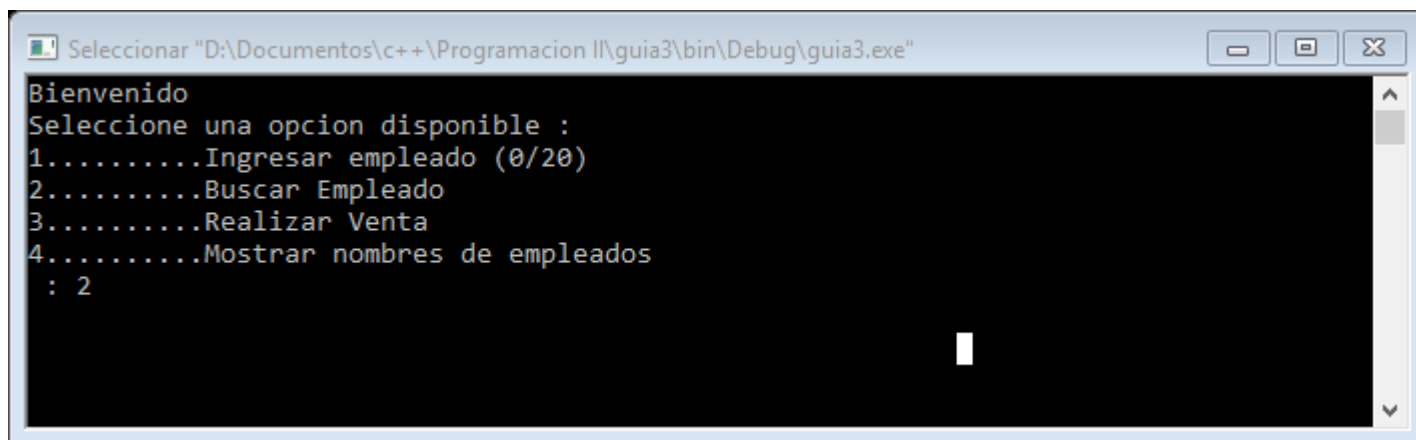
1. Completar la función si el año es bisiesto o no, esta función se usara en la función de validación de fecha. (20%)
2. Completar las funciones si una fecha es válida, para esto verificar si el día se excede del número de mes correspondiente (teniendo en cuenta febrero de 29 o 28 días), y además si el número de mes no sobrepasa de 12, también verificar si son números positivos. (30%)
3. Crear un menú donde se pida insertar un nuevo empleado de un total de 20 disponibles(mostrar la cantidad disponible), el menú contara con la opción de buscar empleado mediante el nombre y mostrar sus datos, así como realizar ventas mediante la búsqueda por el nombre, y la opción de mostrar el nombre de todos los empleados insertados. (50%)

Nota: para el ejercicio 3 puede usar la estructura de dato arreglo del tipo Empleado:

```
#define TAM 20
```

```
Empleado empleados[TAM];
```

Se recomienda que sea una variable global, así como usar estructuras repetitivas para su llenado.



```
Seleccionar "D:\Documentos\c++\Programacion II\guia3\bin\Debug\guia3.exe"
Bienvenido
Seleccione una opcion disponible :
1.....Ingresar empleado (0/20)
2.....Buscar Empleado
3.....Realizar Venta
4.....Mostrar nombres de empleados
: 2
```