

Conceptos de programación orientada a objetos



Ejemplo herencia



Ejemplo herencia

```

1  #include <iostream>
2  #include <stdlib.h>
3  using namespace std;
4
5  class Persona {
6      private:
7          string nombre;
8          int edad;
9      public:
10         Persona(string,int);
11         void verPersona();
12     };
13
14     class Empleado : public Persona{
15     private:
16         string cargo;
17         float salario;
18     public:
19         Empleado(string,int,string,float);
20         void verEmpleado();
21     };
22
23     Persona::Persona(string elNombre,int laEdad){
24         nombre = elNombre;
25         edad = laEdad;
26     }
27
28     Empleado::Empleado(string elNombre,int laEdad,string elCargo,float elSalario) : Persona(elNombre,laEdad){
29         cargo = elCargo;
30         salario = elSalario;
31     }
32
33     void Empleado::verEmpleado(){
34         verPersona();
35         cout<<"Cargo: "<<cargo<<endl;
36         cout<<"Salario: "<<salario<<endl;
37     }
38
39     void Persona::verPersona(){
40         cout<<"Nombre: "<<nombre<<endl;
41         cout<<"Edad: "<<edad<<endl;
42     }
43
44     int main(){
45         Empleado empleado1 = Empleado("Camila Soto",19,"Recepcionista",890000);
46
47         empleado1.verEmpleado();
48
49         system("pause");
50         return 0;
51     }

```

En el siguiente ejemplo se observa la aplicación de la herencia en C++ a través de la creación de un programa con una superclase llamada Persona, que tiene una clase hija llamada Empleado. Se debe digitar el código en el Dev-C++ para continuar con la práctica del lenguaje.

A continuación, se explican los apartes del código donde se realiza la creación de la clase padre (Persona) y el proceso de herencia en la clase hija (Empleado).

Ejemplo herencia

```

1  #include <iostream>
2  #include <stdlib.h>
3  using namespace std;
4
5  class Persona {
6      private:
7          string nombre;
8          int edad;
9      public:
10         Persona(string,int);
11         void verPersona();
12     };
13
14     class Empleado : public Persona{
15         private:
16             string cargo;
17             float salario;
18         public:
19             Empleado(string,int,string,float);
20             void verEmpleado();
21     };
22
23     Persona::Persona(string elNombre,int laEdad){
24         nombre = elNombre;
25         edad = laEdad;
26     }
27
28     Empleado::Empleado(string elNombre,int laEdad,string elCargo,float elSalario) : Persona(elNombre,laEdad){
29         cargo = elCargo;
30         salario = elSalario;
31     }
32
33     void Empleado::verEmpleado(){
34         verPersona();
35         cout<<"Cargo: "<<cargo<<endl;
36         cout<<"Salario: "<<salario<<endl;
37     }
38
39     void Persona::verPersona(){
40         cout<<"Nombre: "<<nombre<<endl;
41         cout<<"Edad: "<<edad<<endl;
42     }
43
44     int main(){
45         Empleado empleado1 = Empleado("Camila Soto",19,"Recepcionista",890000);
46
47         empleado1.verEmpleado();
48
49         system("pause");
50         return 0;
51     }

```

Clase padre "Persona"

- Entre las líneas 5 y 12 se crea la clase Persona con sus respectivos atributos y métodos, dicha creación se realiza del mismo modo en que fue explicado para la figura Inicialización del constructor en el ejemplo 1.

Ejemplo herencia

```

1  #include <iostream>
2  #include <stdlib.h>
3  using namespace std;
4
5  class Persona {
6      private:
7          string nombre;
8          int edad;
9      public:
10         Persona(string,int);
11         void verPersona();
12     };
13
14     class Empleado : public Persona{
15     private:
16         string cargo;
17         float salario;
18     public:
19         Empleado(string,int,string,float);
20         void verEmpleado();
21     };
22
23     Persona::Persona(string elNombre,int laEdad){
24         nombre = elNombre;
25         edad = laEdad;
26     }
27
28     Empleado::Empleado(string elNombre,int laEdad,string elCargo,float elSalario) : Persona(elNombre,laEdad){
29         cargo = elCargo;
30         salario = elSalario;
31     }
32
33     void Empleado::verEmpleado(){
34         verPersona();
35         cout<<"Cargo: "<<cargo<<endl;
36         cout<<"Salario: "<<salario<<endl;
37     }
38
39     void Persona::verPersona(){
40         cout<<"Nombre: "<<nombre<<endl;
41         cout<<"Edad: "<<edad<<endl;
42     }
43
44     int main(){
45         Empleado empleado1 = Empleado("Camila Soto",19,"Recepcionista",890000);
46
47         empleado1.verEmpleado();
48
49         system("pause");
50         return 0;
51     }

```

Clase hija "Empleado"

- En la línea 14 se define la clase Empleado y para indicar que es una clase hija; es decir, que hereda de la clase Persona, se escriben los dos puntos, la palabra *public* y el nombre de la clase padre. En este caso el *public* significa que la clase Empleado puede manipular todos los elementos públicos de la clase Persona.
- En las líneas 16 y 17 se declaran los atributos propios de la clase Empleado.
- En la línea 19 se declara el constructor de la clase Empleado donde se inicializan tanto los atributos heredados (nombre y edad) como los atributos propios (cargo y salario). Se debe conservar el orden de ocurrencia de dichos atributos; es decir, incluir primero los de la clase padre y luego los de la clase hija.
- En la línea 20 se incluye un método llamado verEmpleado() propio de la clase Empleado; por lo tanto, dicha clase tiene un método propio llamado verEmpleado() y un método heredado llamado verPersona();

Ejemplo herencia

```

1  #include <iostream>
2  #include <stdlib.h>
3  using namespace std;
4
5  class Persona {
6      private:
7          string nombre;
8          int edad;
9      public:
10         Persona(string,int);
11         void verPersona();
12     };
13
14     class Empleado : public Persona{
15         private:
16             string cargo;
17             float salario;
18         public:
19             Empleado(string,int,string,float);
20             void verEmpleado();
21     };
22
23     Persona::Persona(string elNombre,int laEdad){
24         nombre = elNombre;
25         edad = laEdad;
26     }
27
28     Empleado::Empleado(string elNombre,int laEdad,string elCargo,float elSalario) : Persona(elNombre,laEdad){
29         cargo = elCargo;
30         salario = elSalario;
31     }
32
33     void Empleado::verEmpleado(){
34         verPersona();
35         cout<<"Cargo: "<<cargo<<endl;
36         cout<<"Salario: "<<salario<<endl;
37     }
38
39     void Persona::verPersona(){
40         cout<<"Nombre: "<<nombre<<endl;
41         cout<<"Edad: "<<edad<<endl;
42     }
43
44     int main(){
45         Empleado empleado1 = Empleado("Camila Soto",19,"Recepcionista",890000);
46
47         empleado1.verEmpleado();
48
49         system("pause");
50         return 0;
51     }

```

Inicialización del constructor de la clase "Persona"

- Entre las líneas 23 y 26 se inicializa el constructor de la clase Persona; lo cual se realiza del mismo modo en que fue explicado para la figura Inicialización del constructor en el ejemplo 1.

Ejemplo herencia

```

1  #include <iostream>
2  #include <stdlib.h>
3  using namespace std;
4
5  class Persona {
6      private:
7          string nombre;
8          int edad;
9      public:
10         Persona(string,int);
11         void verPersona();
12     };
13
14     class Empleado : public Persona{
15         private:
16             string cargo;
17             float salario;
18         public:
19             Empleado(string,int,string,float);
20             void verEmpleado();
21     };
22
23     Persona::Persona(string elNombre,int laEdad){
24         nombre = elNombre;
25         edad = laEdad;
26     }
27
28     Empleado::Empleado(string elNombre,int laEdad,string elCargo,float elSalario) : Persona(elNombre,laEdad){
29         cargo = elCargo;
30         salario = elSalario;
31     }
32
33     void Empleado::verEmpleado(){
34         verPersona();
35         cout<<"Cargo: "<<cargo<<endl;
36         cout<<"Salario: "<<salario<<endl;
37     }
38
39     void Persona::verPersona(){
40         cout<<"Nombre: "<<nombre<<endl;
41         cout<<"Edad: "<<edad<<endl;
42     }
43
44     int main(){
45         Empleado empleado1 = Empleado("Camila Soto",19,"Recepcionista",890000);
46
47         empleado1.verEmpleado();
48
49         system("pause");
50         return 0;
51     }

```

Inicialización del constructor de la clase "Empleado"

- En la línea 28 se inicializa el constructor, para lo cual debe indicarse el nombre (Empleado) y de qué tipo es (Empleado) y a continuación entre paréntesis los datos que recibe con su respectivo tipo. Teniendo en cuenta que esta es la clase hija, deben incluirse tanto los atributos heredados como los propios; sin embargo, los datos de los atributos heredados ya fueron indicados, razón por la cual deben ponerse al final los dos puntos (:), el nombre de la clase padre y los atributos inicializados en el constructor de dicha clase.
- En las líneas 29 y 30 se indica que el cargo y el salario (atributos propios de la clase hija *Empleado*) deben ser iguales a las nuevas variables que serán ingresadas cuando se cree el objeto.

Ejemplo herencia

```

1  #include <iostream>
2  #include <stdlib.h>
3  using namespace std;
4
5  class Persona {
6      private:
7          string nombre;
8          int edad;
9      public:
10         Persona(string,int);
11         void verPersona();
12     };
13
14     class Empleado : public Persona{
15     private:
16         string cargo;
17         float salario;
18     public:
19         Empleado(string,int,string,float);
20         void verEmpleado();
21     };
22
23     Persona::Persona(string elNombre,int laEdad){
24         nombre = elNombre;
25         edad = laEdad;
26     }
27
28     Empleado::Empleado(string elNombre,int laEdad,string elCargo,float elSalario) : Persona(elNombre,laEdad){
29         cargo = elCargo;
30         salario = elSalario;
31     }
32
33     void Empleado::verEmpleado(){
34         verPersona();
35         cout<<"Cargo: "<<cargo<<endl;
36         cout<<"Salario: "<<salario<<endl;
37     }
38
39     void Persona::verPersona(){
40         cout<<"Nombre: "<<nombre<<endl;
41         cout<<"Edad: "<<edad<<endl;
42     }
43
44     int main(){
45         Empleado empleado1 = Empleado("Camila Soto",19,"Recepcionista",890000);
46
47         empleado1.verEmpleado();
48
49         system("pause");
50         return 0;
51     }

```

Creación de métodos clases "Empleado y Persona"

- Entre las líneas 33 y 37 se crea el método verEmpleado(), cuyo objetivo consiste en mostrar toda la información del empleado y teniendo en cuenta que Empleado es la clase hija, hereda el método verPersona() de la clase padre; por lo tanto, la primera instrucción de dicho método consiste en invocar el método verPersona, lo cual mostrará el nombre y la edad, de modo que solo resta incluir las instrucciones que muestren el cargo y el salario, lo cual se realiza en las líneas 35 y 36.
- Entre las líneas 39 y 41 se crea el método verPersona() que se encarga de mostrar el nombre y la edad.

Ejemplo herencia

```

1  #include <iostream>
2  #include <stdlib.h>
3  using namespace std;
4
5  class Persona {
6      private:
7          string nombre;
8          int edad;
9      public:
10         Persona(string,int);
11         void verPersona();
12     };
13
14     class Empleado : public Persona{
15     private:
16         string cargo;
17         float salario;
18     public:
19         Empleado(string,int,string,float);
20         void verEmpleado();
21     };
22
23     Persona::Persona(string elNombre,int laEdad){
24         nombre = elNombre;
25         edad = laEdad;
26     }
27
28     Empleado::Empleado(string elNombre,int laEdad,string elCargo,float elSalario) : Persona(elNombre,laEdad){
29         cargo = elCargo;
30         salario = elSalario;
31     }
32
33     void Empleado::verEmpleado(){
34         verPersona();
35         cout<<"Cargo: "<<cargo<<endl;
36         cout<<"Salario: "<<salario<<endl;
37     }
38
39     void Persona::verPersona(){
40         cout<<"Nombre: "<<nombre<<endl;
41         cout<<"Edad: "<<edad<<endl;
42     }
43
44     int main(){
45         Empleado empleado1 = Empleado("Camila Soto",19,"Recepcionista",890000);
46
47         empleado1.verEmpleado();
48
49         system("pause");
50         return 0;
51     }

```

Creación de objeto como instancia de la clase "Empleado"

- En la línea 45 se crea el objeto llamado empleado1 de la clase Empleado inicializando tanto los atributos heredados como los propios con los valores nombre Camila Soto, edad 19, cargo Recepcionista y salario 890000.
- En la línea 47 se indica al objeto empleado1 que ejecute el método verEmpleado().

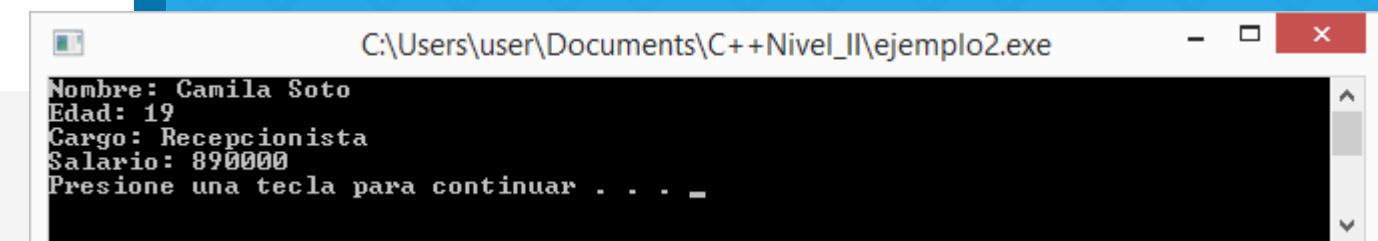
Ejemplo herencia

```

1  #include <iostream>
2  #include <stdlib.h>
3  using namespace std;
4
5  class Persona {
6      private:
7          string nombre;
8          int edad;
9      public:
10         Persona(string,int);
11         void verPersona();
12     };
13
14     class Empleado : public Persona{
15     private:
16         string cargo;
17         float salario;
18     public:
19         Empleado(string,int,string,float);
20         void verEmpleado();
21     };
22
23     Persona::Persona(string elNombre,int laEdad){
24         nombre = elNombre;
25         edad = laEdad;
26     }
27
28     Empleado::Empleado(string elNombre,int laEdad,string elCargo,float elSalario) : Persona(elNombre,laEdad){
29         cargo = elCargo;
30         salario = elSalario;
31     }
32
33     void Empleado::verEmpleado(){
34         verPersona();
35         cout<<"Cargo: "<<cargo<<endl;
36         cout<<"Salario: "<<salario<<endl;
37     }
38
39     void Persona::verPersona(){
40         cout<<"Nombre: "<<nombre<<endl;
41         cout<<"Edad: "<<edad<<endl;
42     }
43
44     int main(){
45         Empleado empleado1 = Empleado("Camila Soto",19,"Recepcionista",890000);
46
47         empleado1.verEmpleado();
48
49         system("pause");
50         return 0;
51     }

```

La ejecución del programa anterior devuelve como resultado lo que se observa a continuación:



```

C:\Users\user\Documents\C++Nivel_II\ejemplo2.exe
Nombre: Camila Soto
Edad: 19
Cargo: Recepcionista
Salario: 890000
Presione una tecla para continuar . . . _

```