Programación Orientada a Objetos - Referencias

Pedro Fernando Flores Palmeros
ESIME - ZACATENCO

Abril 2020

1. ¿Qué es una referencia?

Una referencia es un alias o sinónimo; cuando crea una referencia, la inicializa conel nombre de otro objeto que viene siendo el destino. A partir de ese momento, la referencia acua como un nombre alternativo para el destino y cualquier cosa que le haga a la referencia, en realidad se la hace al destino.

PUede crear una referencia escribiendo el tipo del objeto de destino, seguido deloperador de referencia `&` y el nombre de la referencia.

A continuación se muestra un programa en el que se observa la implementación de una referencia, al comienzo se modifica el valor de la variable y ese cambio se ve reflejado en la referencia y después se modifica el valor de la referencia y se ve el cambio en la variable original.

```
#include <iostream>

using namespace std;

int main(){
   int miEntero;
   int  & ref_miEntero = miEntero;

   //Se modifica miEntero
   miEntero = 7;

   cout << "miEntero: " << miEntero << endl;
   cout << "ref_miEntero: " << ref_miEntero << endl;

   // Se modifica la referencia
   ref_miEntero = -100;

   cout << "miEntero: " << miEntero << endl;
   cout << "ref_miEntero: " << ref_miEntero << endl;

   return 0;
```

```
}
```

2. Uso del operador de dirección &

Si le pide la dirección de la referencia, ésta regresa el valor de la variable destino. Ésta es la naturaleza de las referencias: Son un alias para el destino, tal como se muestra en el siguiente código.

```
#include <iostream>

using namespace std;

int main(){
   int miEntero;
   int & ref_miEntero = miEntero;

   miEntero = -100;

   cout << "miEntero: " << miEntero << endl;
   cout << "ref_miEntero" << ref_miEntero << endl;

   cout << "La direccion de miEntero es: " << &miEntero << endl;
   cout << "La direccion de ref_miEntero es: " << &miEntero << endl;
}
```

Las referrencias se inicializan al crearse, y siempre actúan como sinónimo para su destino, incluso al aplicar el operador de dirección.

3. Las referencias NO SE PUEDEN REASIGNAR

Como se ha mencionado anteriormente e incluso se puede ver en los programas anteriores, cuando se crea una referencia, ésta se debe de inicializar, de lo contrario marcaría error.

Otro aspecto importante de las referencias es que una vez que se han sido asignadas no pueden cambiar.

Observe el siguiente código, que de manera general hace lo siguiente: declara un `Numero1` y le asigna el valor de `1027`, declara una referencia y la inicializa `rNumero1 = Numero1`, después simplemente se mandan a imprimir los valores del número y su referencia y deben de ser los mismos, también se imprimen las direcciones de ambos y como es de esperarse, coinciden, hasta aquí es el uso común de una referencia.

En la siguiente línea se trata delcara un `Numero2` y se le asigna el valor de `--50` y después se hace la siguiente asignación `rNumero1 = Numero2` y se imprimen los valores y direcciones

de las dos variables y la referencia, a continuación se muestra el código y después la salida en terminal del programa.

```
#include <iostream>

using namespace std;

int main(){
   int Numero1 = 1027;
   int & rNumero1 = Numero1;

   cout << "El valor de Numero1 es: " << Numero1 << endl;
   cout << "El valor de rNumero1 es: " << rNumero1 << endl;

   cout << "La direccion de Numero1 es: " << &Numero1 << endl;
   cout << "La direccion de rNumero1 es: " << &rNumero1 << endl;

   int Numero2 = -50;

   rNumero1 = Numero2;

   cout << "El valor de Numero1 es: " << Numero1 << endl;
   cout << "El valor de rNumero1 es: " << rNumero1 << endl;
   cout << "El valor de Numero2 es: " << Numero2 << endl;

   cout << "La direccion de Numero1 es: " << &Numero1 << endl;
   cout << "La direccion de rNumero1 es: " << &rNumero1 << endl;
   cout << "La direccion de Numero2 es: " << &Numero2 << endl;

}
```

```
El valor de Numero1 es: 1027
El valor de rNumero1 es: 1027
La direccion de Numero1 es: 0x7ffede05f868
La direccion de rNumero1 es: 0x7ffede05f868
El valor de Numero1 es: -50
El valor de rNumero1 es: -50
El valor de Numero2 es: -50
La direccion de Numero1 es: 0x7ffede05f868
La direccion de rNumero1 es: 0x7ffede05f868
La direccion de Numero2 es: 0x7ffede05f86c
```

Observe que cuando se imprimen los valores todos son iguales, cuando se imprimen los valores de las direcciones observe que la referencia sigue teniendo la dirección de la variable `Numero1`, entonces no se pudo hacer el cambio de referencia. **las referencias no se pueden asginar y son simepre UN alias para su destino**.

4. Referencias a objetos

Cualquier objeto se puede refernciar, inlusive los objetos definidos por el usuario. Para ejemplificar la referencia a un objeto, supongamos que ya se ha desarrollado la clase `robot` entonces se puede declarar un objeto tipo `robot` pero el objeto será `R2D2`, y la referencia sería `refR2D2` en código se debería ver algo similar

```
#include <iostream>
#include <string>

using namespace std;

class Robot{
   private:
   string Name;
   string Procesador = "Arduino";

   public:
   Robot(string Nombre){
      this->Name = Nombre;
   }

   void Saludar(){
      cout << "Hola mi nombre es " << this->Name << endl;
   }

   void Descripcion(){
      cout << "Tengo un@ " << this->Procesador << " como procesador" << endl;
   }
};

int main(){
   Robot R2D2("R2D2");
   Robot & refR2D2 = R2D2;

   R2D2.Saludar();
   refR2D2.Saludar();
   refR2D2.Descripcion();
}
```

Observe que la referencia se hace al objeto no a la clase