Informática I para Bachillerato

C/C++
Programación Orientada a Objetos(POO)

José Luis Alonzo Velázquez

CIMAT

Sesión 16

Sintaxis

En C/C++ se forma una estructura utilizando la palabra reservada **struct**, seguida por un campo etiqueta opcional conocida como rótulo de la estructura, y luego una lista de miembros dentro de la estructura. La etiqueta opcional se utiliza para crear otras variables del tipo particular de la estructura:

```
1 struct [ <nombre tipo de estructura > ] {
2     [<tipo> <nombre-variable[,nombre-variable,...] >];
3     [<tipo> <nombre-variable[,nombre-variable,...] >];
4     ...
5     } [ <variables de estructura> ] ;
```

Un punto y coma finaliza la definición de una estructura puesto que ésta es realmente una sentencia C/C++.

Programación Orientada a Objetos

La programación orientada a objetos o POO (OOP según sus siglas en inglés) es un paradigma de programación que usa objetos y sus interacciones, para diseñar aplicaciones y programas informáticos. Está basado en varias técnicas, incluyendo herencia, abstracción, polimorfismo y encapsulamiento.

Objetos

1. Objetos son estructuras de datos en memoria.
2. Un objeto lleva dos tipos de cosa : datos (o atributos) y código ejecutable (operaciones que este objeto puede hacer, como funciones C).
3. Los objetos tienen una interfaz y una implementación; solo la interfaz esta visible por los otros objetos del entorno.

Datos:

- Son guardados en variables propias al objeto.

Operaciones:

- Las operaciones que puede hacer el objeto se llaman métodos.
- La interfaz precisa el prototipo de cada método : nombre, valor de regreso, nombre y tipos de los argumentos.
- Los métodos pueden cumplir tareas diversas : crear objetos, enviar mensajes, hacer manipulación sobre las variables internas del objeto.

Clase

Definiciones de las propiedades y comportamiento de un tipo de objeto concreto. La instanciación es la lectura de estas definiciones y la creación de un objeto a partir de ellas.

Herencia

Es la facilidad mediante la cual la clase D hereda en ella cada uno de los atributos y operaciones de C, como si esos atributos y operaciones hubiesen sido definidos por la misma D. Por lo tanto, puede usar los mismos métodos y variables publicas declaradas en C. Los componentes registrados como "privados" (private) también se heredan, pero como no pertenecen a la clase, se mantienen escondidos al programador y sólo pueden ser accedidos a través de otros métodos públicos. Esto es así para mantener hegemónico el ideal de OOP.

Objeto

Entidad provista de un conjunto de propiedades o atributos (datos) y de comportamiento o funcionalidad (métodos) los mismos que consecuentemente reaccionan a eventos. Se corresponde con los objetos reales del mundo que nos rodea, o a objetos internos del sistema (del programa). Es una instancia a una clase.

Método

Algoritmo asociado a un objeto (o a una clase de objetos), cuya ejecución se desencadena tras la recepción de un "mensaje". Desde el punto de vista del comportamiento, es lo que el objeto puede hacer. Un método puede producir un cambio en las propiedades del objeto, o la generación de un .^evento con un nuevo mensaje para otro objeto del sistema.

```
1 class Rectangulo{
2     int x;
3     int y;
4     int ancho;
5     int alto;
6     //faltan las funciones miembro
7 }
```

Método

```
1 class Rectangulo{
2     int x;
3     int y;
4     int ancho;
5     int alto;
6    int calcularArea(){
7     return (ancho*alto);
8    }
9 }
```

SET

```
1  class Rectangulo{
2      int x;
3      int y;
4      int ancho;
5      int alto;
6     void desplazar(int dx, int dy){
7      x+=dx;
8      y+=dy;
9     }
10 }
```

Constructores

```
1  class Rectangulo{
2      int x;
3      int y;
4      int ancho;
5      int alto;
6     Rectangulo(int x1, int y1, int w, int h){
7      x=x1;
8      y=y1;
9      ancho=w;
10     alto=h;
11    }
12 }
```

Public, Private

```
1  public class Rectangulo {
2      int x;
3      int y;
4      int ancho;
5      int alto;
6      public Rectangulo() {
7          x=0;
8          y=0;
9          ancho=0;
10         alto=0;
11     }
```

Public, Private

```
1     public Rectangulo(int x1, int y1, int w, int h) {
2         x=x1;
3         y=y1;
4         ancho=w;
5         alto=h;
6     }
7     public Rectangulo(int w, int h) {
8         x=0;
9         y=0;
10        ancho=w;
11        alto=h;
12    }
```

Public, Private

```
1      int calcularArea(){
2      return (ancho*alto);
3     }
4     void desplazar(int dx, int dy){
5      x+=dx;
6      y+=dy;
7     }
8     boolean estaDentro(int x1, int y1){
9      if((x1>x)&&(x1<x+ancho)&&(y1>y)&&(y1<y+ancho)){
10          return true;
11     }
12     return false;
13    }
14 }
```

 Como Programar en C/C++, Deitel (Prentice Hall), 2da Edición.

 Programming Principles and Practice Using C++, Bjarne Stroustrup.

 http://www.codeblocks.org

 http://www.wxwidgets.org