#	Fecha	Alcance
1	2/9	*- Orientación a Objetos.* *- Clase, objeto/instancia, atributo, método, mensaje.* *- Objetos que conocen a objetos.* *- Variables como punteros.* *- Creación de objetos, Inicialización.* *- El sistema orientado a objetos: UI/Servicio + modelo (nuestros objetos nunca hacen I/O).* *- Modularización. ¿Por qué lo hacemos en objetos? self/this como estrategia para evitar duplicación, aprovechar comportamiento heredado, acortar métodos.*
2	16/9	***Herencia:*** *- Heredar, redefinir, y extender comportamiento heredado.* *- El keyword super.* *- Binding dinámico en el contexto de herencia.* *- Las colecciones como objetos.* *- Polimorfismo.* *- Delegación como estrategia de diseño.* *- La importancia del encapsulamiento.* *- Introducción a diagramas de clase de UML.* *- Primer caso de estudio.* *- Interfaces, su rol en el chequeo de tipos y su relación con polimorfismo.*
3	30/9	*- Los Tests Unitarios como mecanismo de aseguramiento de calidad y como guía al proceso de desarrollo.* *Automatización de test de unidad con herramientas de la familia xUnit .* *- Pruebas de particiones equivalentes y valores de borde.* *- Colecciones heterogéneas de objetos; polimorfismo entre colecciones y polimorfismo entre los elementos en una colección. Iteradores de colecciones (y objetos que abstraen y/o soportan el comportamiento de iteración).* *- Clausuras léxicas en objetos.*
4	14/10	*- Identificar clases y responsabilidades a partir de una especificación.* *- Modelo conceptual (clases candidatas).* *- Pre y post contratos (que hay antes y que queda después) para los casos de uso.* *- Derivar métodos a partir de los pre y post contratos.* *- Discutimos Herencia vs. Composición.* *- Editores UML: Editores gráficos vs. Editores UML.* *- Incorporamos herencia, sinónimos, y objetos no tan obvios.* *- Discutimos Herencia vs. Composición.*
5	11/11	*- Arquitectura de la aplicación / servicio.* *- Separación en capas.* *- La importancia del modelo de dominio.* *- Particularidades de otros lenguajes de programación OO.* *- Lenguajes basados en clases y basados en instancias.* *- El rol del tipado Tipado estático y dinámico.*