Ejercicio 1

a) Clase Dios: la clase “Producto” representa a los productos y a los servicios por ende almacena v.i. que en ocasiones quedan obsoletas.
Envidia de atributos: en la clase “Pedido” se calcula el costo de los items con un método privado donde se utiliza implementación interna de “Producto”.
Switch statement: se usa una estructura de control if en la clase “Pedido” para determinar si un item es un servicio o producto.
Rompe encapsulamiento: las v.i. de la clase “Producto” son públicas lo que puede llegar a tener problemas de obtener valores indeseados.

Ejercicio 2

a) getCosto() de Servicio usa particiones equivalentes.
getCosto() de Producto usa particiones equivalentes.
getCostoTotal() de Pedido usa particiones equivalentes.
b) Tabla

Descripción del caso	Método a testear	Valores usados
test itemCosto	getCosto() de la clase Servicio	horasTrabajadas = 5 valorHora = 300
test itemCosto	getCosto() de la clase Producto	costo = 200 costoEnvio = 50 peso = 3
test sinItems	getCostoTotal() de la clase Pedido	no se agregan items
test conItems	getCostoTotal() de la clase Pedido	servicio1 servicio2 producto