CASE PROGRAM – ENUM

import case.lang.System

namespace HelloWorld {

//String->Object->Main is a type constructor

String->Object->Main

#public class Program

//during compilation this function is included in the

//codebase whether it is used or not

[@required [public Program(String [] args)

System.out.println(@texts:“Hello World”)

Print sizeOf(“Size of args structure” + sizeof(args))

listClasses = cons ClassA Class B

listClasses2 = cons ClassA ClassB Class C

pairClasses = pair ClassA ClassB //Class, ClassB

ClassA = car listClasses

ClassesEnd = cdr listClasses

]

**Bridge –** Decouple an abstraction from its implementation so that the two can vary independently.

Platform depenent code and abstractions should be kept apart. In the Bridge pattern, the implementation and its abstraction are connected only by a single connection, known as the Bridge.

//left of bridge

String->Object->Window

#public class Window

[DrawText()]

[DrawRect()]

Imp.DevDrawLine()

Imp.DevDrawLine()

Imp.DevDrawLine()

Imp.DevDrawLine()

endclass

String->Object->Window->IconWindow

#public class IconWindow

[DrawBorder()

[DrawRect()]

[DrawText()]

]

endclass

String->Object->Window->TransientWindow

#public class TransientWindow

[DrawCloseBox()

[DrawRect()]

]

endclass

String->Object->WindowImp

#public class WindowImp

[DevDrawText()]

[DevDrawLine()]

endclass WindowImp

String->Object->WindowIMP->XWindowIMP

XWindowIMP  
 [DevDrawText()]

[DevDrawLine()

[XDrawLine()]

[XdrawString()]

]

String->Object->PMWindowIMP->PMWindowIMP

#public class PMWindowIMP

[DevDrawText]

[DevDrawLine]

*Window* exposes an interface to its subclass heiarchy by exposing drawTex() and drawRect(). WindowIMP (the other side of the bridge) is the implementation of the abstract/interface of Window.