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

]

**Adapter –** Convert the interface of a class into another interface clients would expect. An adapter takes two classes and lets them work together when they typically wouldn’t be able to. Acts as the essential “middle-man.”

String->Object->Client //entry point

#class Client

#endclass Client

//one side of the adapter

String->Object->Target

#public class Target

[Request()] ///called by adapter

#endclass Target

//adapter takes request from target, converts it to specific request for the adaptee //and to send back to adapter

//to send out new format

String\_>Object->Adapter

#public classs Adapter

//takes out adaptee implement results and sends out specific request

[Request()]

#endclass Adapter

(implemetation of the adapter)

String->Object->Adaptee

#public class Adaptee

[SpecificRequest()] //

#endclass Adaptee

Essentially Target wants to execute some commands but doesn’t know how to communicate with Adaptee. Adapter serves as a middle man. Target communicates its request to adapter, and adapter convert that into a request that adaptee can understand. This process occurs vice versa.