CASE – SYNOPERATOR

package myentitites

import case.lang.System

namespace EntitiesNameSpace {

String->Object->Main

#public class Program

//constant field

const float PI = 3.14

[public Program(String [] args)

[EntityPool Pool = EntityPool.getEntityPool]

assert(Pool) //asserts that Pool exists and has a value

Stream (n) String

Int MyInt = EntityPool.getStreamMemory() //retrieve mem from pool

Int GetInt = EntityPool.get(“MyInt”) //pointer to MyInt using pool get

//get pointer to CurrentLocationInList from the pool

Int ListStatus = n.get(“CurrentLocationInList”)

//output

//Futures are read-only placeholder for a variable

//Promise is a one time writable placeholder sets the value of the //future

//Suppose we call an API, then we will be actually blocking. If we use //a Future we establishe a promise, removes blocking? Gets a portion //back.

//future – read only promise

//promise – read/write

**future** atom List myList

myList.populate()

functionThatTakesAWhile()

**promise** WriteLine(myList.text())

]

#endclass