public class Approval{

//////////////////////////////

// known variables from UML //

//////////////////////////////

-populationMean : double STATIC

-mean : double

-variance : double

-numTimesToRun : int

public Approval(int numTimesToRun){

this.numTimesToRun = numTimesToRun;

}

//////////////////////

// methods from UML //

//////////////////////

//------------------------------------------------------

+doPoll(samples : int) : void

{

Gets poll’s raw variance and divides it by (samples – 1) to obtain an estimate

of the variance of the measured average, relative to the underlying true

population mean.

It prints the standard deviation of variation in the measured average,

as a percentage.

Finally, it reveals the “true” underlying population mean, as a percentage,

for comparison.

The doPoll method loops for the specified number of samples, accumulating value

and value squared.

After the looping, it divides each accumulation by the specified number of samples

to obtain the respective averages. Then it sets mean, and:

variance <- avgOfValueSquared – avgValue \* avgValue.

}

//------------------------------------------------------

+getMean() : double

{

return this.mean;

}

//------------------------------------------------------

+getVariance() : double

{

return this.variance;

}

//------------------------------------------------------

+setPopulationMean() : void STATIC

{

The setPopulationMean method just makes populationMean

equal to whatever value is returned by Math.random.

this.populationMean = Math.random();

}

//------------------------------------------------------

+getPopulationMean() : double STATIC

{

return this.populationMean;

}

//------------------------------------------------------

+getSample : double STATIC

{

The getSample method implements the following algorithm,

which provides a simple representation of the population’s

distribution of approval values:

sample <- Math.random

where <- Math.random

if where < populationMean

{

sample <- populationMean + (1 – populationMean) \* sample

}

else

{

sample <- populationMean – populationMean \* sample

}

}

//------------------------------------------------------

+verifyModel() : void STATIC

{

The verifyModel method establishes an independent population mean,

and loops through one million samples, to confirm that the average

of all of them approximately equals the established population mean,

and to confirm that all samples are within the allowed range.

}

} // end class

public class ApprovalDriver{

main(){

int samples

The main method calls verifyModel to prove that the getSample algorithm is OK.

Approval.verifyModel();

Then it calls setPopulationMean to set the population mean approval at an

unknown random value between 0.0 and 1.0.

Approval.setPopulationMean();

Then main asks the user for a desired number of samples, and it instantiates

an Approval object, poll.

System.out.print("Enter number of samples in poll: ");

samples = sysIn.next();

Approval poll = new Approval();

Then it asks poll to call doPoll and prints poll’s mean value, as a percentage.

poll.doPoll(samples);

} // end main

} // end class