**Question 1, a:**

**import** java.util.\*;

**public** **class** question1A {

**private** Random random = **new** Random();

**public** **class** Pipe {

**int** gamma;

**int** spike\_num;

**public** Pipe() {

**this**.gamma=random.nextInt(1000);//Each spike carries a

//fixed amount of charge Gamma.

**this**.spike\_num=random.nextInt(1000);//particular amounts

//of water (i.e. charge) to enter at a time

}

}

**class** Bucket{

**int** c\_old ; //capacity "bucket" having a maximal capacity

**int** l; //hole in the bucket with a small leak of a certain

//constant rate

Pipe p; //no. of pipes that enters the bucket at a time

**public** Bucket(){

c\_old = random.nextInt(1000);

l = random.nextInt(1000);

p = **new** Pipe();

}

**public** **void** action() {

**int** c\_new=0;

**int** sum=0; // #spikes from pipe i \* gamma pipe i

Thread thread = **new** Thread();

thread.start();

**while** (c\_new<=c\_old) { //If at the new time C is bigger

//than threshold it will be print "spike"

Pipe p=**new** Pipe();

**this**.p=p;

sum=sum+ (p.gamma\*p.spike\_num); //sum ( w\_i .times. // Gamma times the

//number of spikes entering on dendrite i )

c\_new =sum-l;

**try** {

Thread.*sleep*(1);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

thread.interrupt();

System.***out***.println("Spike: " + p.spike\_num);

}

**public** **void** main(String[] argv) {

Bucket bucket= **new** Bucket();

bucket.action();

}

}

}

**Question 1, b+c:**

**import** java.util.\*;

**public** **class** question1B {

**private** Random random = **new** Random();

**public** **class** Pipe {

**final** **int** gamma=1;

**int** spike\_num;

**int** stop; // Time allotted for each pipe

**public** Pipe() {

**this**.spike\_num=random.nextInt(1000);//particular amounts

//of water (i.e. charge) to enter at a time

}

**public** **int** getStop() {

**return** stop;

}

**public** **void** setStop(**int** stop) {

**this**.stop = stop;

}

}

**class** Bucket{

**double** c\_old ; //capacity "bucket" having a maximal capacity

**double** l; //hole in the bucket with a small leak of a certain   
 //constant rate

Pipe p; //no. of pipes that enters the bucket at a time

**public** Bucket(){

c\_old = random.nextInt(1000);

l = (1/50);

p = **new** Pipe();

}

**public** **void** action() {

**double** c\_new=0;

**int** sum=0; // #spikes from pipe i \* gamma pipe i

Thread D\_1 = **new** Thread(); // E1->D\_1

Thread D\_2 = **new** Thread(); // E2->D\_2

D\_1.start();

D\_2.start();

**int** count=0;

Pipe p=**new** Pipe();

**this**.p=p;

p.setStop(random.nextInt(1)+10);

l=l\*(p.spike\_num);

**while** (c\_new<=c\_old) {

**if** (count> p.stop){ // coincidence detector beween //two events,E1 E2

System.***out***.println("Last spike: "+p.spike\_num);

**// solution of section 1.C**

/\*

הסבר- כאשר הצינור הגיע לזמן המוקצב לו, נוסיף 5 ספייקים לדנדיריט הראשון ואם מדובר על הדנדריט השני נוסיף לו 3 ספייקים שיצאו אחד אחרי השני

\*/

**if**(Thread.*currentThread*().getName().matches("D\_1")){

p.spike\_num+=1;

p.spike\_num+=1;

p.spike\_num+=1;

p.spike\_num+=1;

p.spike\_num+=1;

System.***out***.println("Last spike: "+p.spike\_num);

}

**else**{

p.spike\_num+=1;

p.spike\_num+=1;

p.spike\_num+=1;

System.***out***.println("Last spike: "+p.spike\_num);

}

**// end of solution of section 1.C**

Thread.*currentThread*().interrupt();

}

sum=sum+ (p.gamma\*p.spike\_num); //sum (w\_i .times.

//Gamma times the number of spikes entering on dendrite i)

c\_new =sum-l;

**try** {

Thread.*currentThread*().*sleep*(1);

count++;

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

D\_1.interrupt();

D\_2.interrupt();

System.***out***.println("Spike"+ p.spike\_num);

}

}

}