Concurrent programming lab03 - simon.barras

!!!WARNING!!! - C7: Long lines detected

```
            BabyBird.LOGGER.info("Parent hunted " + preNbWorm + " food, " + (lostWorm > 0 ?
        throw new IllegalArgumentException(e.getMessage() + "
```

Please follow usage bellow java babybird [chicks [baby_iter [max_food_size [hunting_success_rate]]]]“);

```java
// Copyright 2021, School of Engineering and Architecture of Fribourg
//
// Licensed under the Apache License, Version 2.0 (the "License");
// you may not use this file except in compliance with the License.
// You may obtain a copy of the License at
//
//     http://www.apache.org/licenses/LICENSE-2.0
//
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.

// author = "Simon Barras"
// date = "2021-11-23"
// version = "0.0.1"
// email = "simon.barras@edu.hefr.ch"
// userid = "simon.barras"
//


import java.util.Random;
import java.util.logging.Logger;

public class BabyBird {

    public static final int MAX_ARGS = 4; // maximum number of arguments
    public static final int SUCCESS_RATE_INDEX = 3; // index of success rate in arguments
    public static final Random RND = new Random(); // random number generator
    public static final int WAIT_BIRD = 10; // max wait time attributed to a bird
    public static final int WAIT_MAX = 50; // total max wait time
    public static final int NB_PARENTS = 2; // number of parents
    public static final int DEFAULT_NB_CHICKS = 17; // default number of baby bird
    public static final int DEFAULT_LIFE_CYCLE = 53; // default life cycle before leaving the nest
    public static final int DEFAULT_MAX_FOOD = 7; // default max food in the nest
    public static final int DEFAULT_HUNTING_RATE = 50; // default change of hunting something
    public static final int MAX_HUNTING_RATE = 100; // max hunting rate (100%)
    public static final Logger LOGGER = Logger.getLogger(BabyBird.class.getName()); // log thread safe

    static {
        System.setProperty("java.util.logging.SimpleFormatter.format", "%5$s %n"); // set log format
    }

    /**
     * Simulate a bird's nest with threads and semaphores.
     *
     * @param args usage: java babybird [chicks [baby_iter [max_food_size [hunting_success_rate]]]]
     */
    public static void main(String[] args) {
        System.out.println("Lab3 - BabyBird simulation");
        Ctrl ctrl = new Ctrl(Logger.getLogger(BabyBird.class.getName()));

        ctrl.run(ctrl.parseArgs(args));

    }
}
```

what's the purpose of this file? & class

```java
// Copyright 2021, School of Engineering and Architecture of Fribourg
//
// Licensed under the Apache License, Version 2.0 (the "License");
// you may not use this file except in compliance with the License.
// You may obtain a copy of the License at
//
//     http://www.apache.org/licenses/LICENSE-2.0
//
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.

// author = "Simon Barras"
// date = "2021-11-23"
// version = "0.0.1"
// email = "simon.barras@edu.hefr.ch"
// userid = "simon.barras"
//

import java.util.concurrent.Semaphore;

/**
 * Object for simulating a baby bird.
 * Use a thread with a routine to live.
 */
public class Chick {
    private Ctrl ctrl; // controller of the application

    private String name; // name of the bird
    private Thread spirit; // thread of the bird
    private long speed; // wait time attributed to the bird
    private Semaphore live; // release when the child leave the nest

    public Chick(Ctrl ctrl, int nbDayToAdult) {
        this.ctrl = ctrl;
        speed = BabyBird.RND.nextInt(BabyBird.WAIT_BIRD + 1);
        spirit = new Thread(() -> {
            // Routine of the bird's life
            for (int i = 0; i < nbDayToAdult; i++) {
                sleep();
                getFood();
                eat();
                digestAnd_();
            }
            leaveNest();
        });
    }

    /**
     * Start the thread and the routine
     *
     * @param name the name of the bird
     */
    public void born(String name) {
        this.name = "Chick " + name;
        live = new Semaphore(0);
        spirit.start();
    }

    /**
     * Call by the main thread for waiting until the bird left the nest
     *
     * @throws InterruptedException
     */
    public void farewellParty() throws InterruptedException {
        live.acquire();
    }

    /**
     * Close the thread
     *
     * @throws InterruptedException
     */
    public void funeral() throws InterruptedException {
        spirit.join();
        BabyBird.LOGGER.info(name + " is buried");
    }

    /**
     * Wait a random time
     */
    private void sleep() {
        BabyBird.LOGGER.info(name + " is sleeping");
        ctrl.nap(spirit, speed);
    }

    /**
     * Take food only if the tank isn't empty and nobody is using the variable for the stock
     */
    private void getFood() {
        BabyBird.LOGGER.info(name + " is getting food");
        ctrl.getFood();
    }

```

```
97:     /**
98:      * Random timer
99:      */
100:    private void eat() {
101:        BabyBird.LOGGER.info(name + " is eating");
102:        ctrl.nap(spirit, speed);
103:    }
104:
105:    /**
106:     * Random timer
107:     */
108:    private void digestAnd_() {
109:        BabyBird.LOGGER.info(name + " is digesting");
110:        ctrl.nap(spirit, speed);
111:    }
112:
113:    /**
114:     * Last method of the life cycle.
115:     * Release the semaphore
116:     */
117:    private void leaveNest() {
118:        BabyBird.LOGGER.info(name + " is leaving the nest");
119:        live.release();
120:    }
121:
122: }
```

```java
// Copyright 2021, School of Engineering and Architecture of Fribourg
//
// Licensed under the Apache License, Version 2.0 (the "License");
// you may not use this file except in compliance with the License.
// You may obtain a copy of the License at
//
//     http://www.apache.org/licenses/LICENSE-2.0
//
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.

// author = "Simon Barras"
// date = "2021-11-23"
// version = "0.0.1"
// email = "simon.barras@edu.hefr.ch"
// userid = "simon.barras"
//

import java.util.concurrent.BrokenBarrierException;
import java.util.concurrent.CyclicBarrier;
import java.util.concurrent.Semaphore;
import java.util.logging.Logger;

/**
 * This class is the controller of the application. It wraps all the methods which aren't in a class int he Lab02.
 */
public class Ctrl {
    private Logger logger;
    private Chick[] chicks; // array of the chicks
    private Parent[] parents; // array of the parents
    private int nbMaxWorm; // number of max worms that can be put in the nest
    private volatile Integer preNbWorm; // number of worms fund by the 2 parents
    private volatile int nbWorm; // number of worms in the nest
    private Semaphore waitEat; // semaphore to wait to get food of the nest
    private Semaphore waitHunt; // semaphore to wait to put food in the nest
    private CyclicBarrier barrier; // the first parent wait the second to put some food in the nest
    private boolean hasChild; // ture if steal at least one child in the nest

    /**
     * Constructor of the class.
     *
     * @param logger logger that will be used to log
     */
    public Ctrl(Logger logger) {
        this.logger = logger;
    }

    /**
     * Start the application.
     *
     * @param args parsed arguments
     */
    public void run(int[] args) {
        System.out.println("Lab03 - BabyBird simulation");
        building_nest(args[0], args[1], args[2], args[3]);
        simulating();
        destroy_nest();
        System.out.println("Lab03 - finish simulation");
    }

    /**
     * Initialize the nest.
     *
     * @param nbChicks     number of chicks
     * @param babyItr      number of life cycle
     * @param foodCapacity max number of worms in the nest
     * @param huntingRate  chance to find something
     */
    private void building_nest(int nbChicks, int babyItr, int foodCapacity, int huntingRate) {
        chicks = new Chick[nbChicks];
        parents = new Parent[BabyBird.NB_PARENTS];
        hasChild = true;
        nbMaxWorm = foodCapacity;
        preNbWorm = 0;
        waitEat = new Semaphore(0);
        waitHunt = new Semaphore(1);
        barrier = new CyclicBarrier(BabyBird.NB_PARENTS, () -> {
            try {
                // food put in the nest by the barrier
                waitHunt.acquire();
                int lostWorm = preNbWorm - nbWorm;
                BabyBird.LOGGER.info("Parent hunted " + preNbWorm + " food, " + (lostWorm > 0 ? lostWorm : 0) + " will b
e lost");
                wormTank(preNbWorm > nbMaxWorm ? nbMaxWorm : preNbWorm);
                preNbWorm = 0;
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        });

        for (int i = 0; i < nbChicks; i++) {
            chicks[i] = new Chick(this, babyItr);
        }
```

```
96:         for (int i = 0; i < BabyBird.NB_PARENTS; i++) {
97:             parents[i] = new Parent(this, foodCapacity, huntingRate);
98:         }
99:     }
100:
101:    /**
102:     * Run the nest life
103:     */
104:    private void simulating() {
105:        int i = 1;
106:        for (Chick chick : chicks) {
107:            chick.born(i++ + "");
108:        }
109:        i = 1;
110:        for (Parent parent : parents) {
111:            parent.born(i++ + "");
112:        }
113:        for (Chick chick : chicks) {
114:            try {
115:                chick.farewellParty();
116:            } catch (InterruptedException e) {
117:                e.printStackTrace();
118:            }
119:        }
120:        hasChild = false;
121:        for (Parent parent : parents) {
122:            waitHunt.release();
123:        }
124:
125:    }
126:
127:    /**
128:     * Clos all threads.
129:     */
130:    private void destroy_nest() {
131:        for (Chick chick : chicks) {
132:            try {
133:                chick.funeral();
134:            } catch (InterruptedException e) {
135:                e.printStackTrace();
136:            }
137:        }
138:        for (Parent parent : parents) {
139:            try {
140:                barrier.reset();
141:                parent.funeral();
142:            } catch (InterruptedException e) {
143:                e.printStackTrace();
144:            }
145:        }
146:    }
147:
148:    /**
149:     * Call by the parent's thread routine
150:     *
151:     * @return true if at least one chick is alive
152:     */
153:    public boolean hasChild() {
154:        return hasChild;
155:    }
156:
157:    /**
158:     * Call to eat some food
159:     */
160:    public void getFood() {
161:        try {
162:            waitEat.acquire();
163:            wormTank(-1);
164:        } catch (InterruptedException e) {
165:            e.printStackTrace();
166:        }
167:    }
168:
169:    /**
170:     * Class to add some food
171:     *
172:     * @param worm number of worms hunted
173:     */
174:    public void putFood(int worm) {
175:        if (hasChild()) {
176:            synchronized (preNbWorm) {
177:                preNbWorm += worm;
178:            }
179:            try {
180:                barrier.await();
181:            } catch (InterruptedException e) {
182:                e.printStackTrace();
183:            } catch (BrokenBarrierException e) {
184:                BabyBird.LOGGER.info("Barrier broken");
185:            }
186:        }
187:
188:    }
189:
190:    /**
191:     * Call when we want to modify the number of worms in the tank
```

give some more info about your exceptions! CJ

like this

```
192:      *
193:      * @param nbWorm worms to add
194:      */
195:     private synchronized void wormTank(int nbWorm) {
196:         this.nbWorm += nbWorm;
197:         if (nbWorm > 0) {
198:             waitEat.release(nbWorm);
199:         } else if (this.nbWorm == 0) {
200:             waitHunt.release();
201:         }
202:     }
203:
204:     /**
205:      * Pause thread
206:      *
207:      * @param t    thread to pause
208:      * @param time at least time to wait
209:      */
210:     public void nap(Thread t, long time) {
211:         try {
212:             t.sleep(time + BabyBird.RND.nextInt(BabyBird.WAIT_MAX - BabyBird.WAIT_BIRD + 1));
213:         } catch (InterruptedException e) {
214:             e.printStackTrace();
215:         }
216:     }
217:
218:     /**
219:      * Format arguments
220:      *
221:      * @param args original arguments
222:      * @return formatted arguments
223:      */
224:     public int[] parseArgs(String[] args) {
225:         int[] result = new int[BabyBird.MAX_ARGS];
226:         try {
227:             for (int i = 0; i < args.length; i++) {
228:                 result[i] = checkArgs(i, args[i]);
229:             }
230:             for (int i = args.length; i < BabyBird.MAX_ARGS; i++) {
231:                 switch (i) {
232:                     case 0:
233:                         result[i] = BabyBird.DEFAULT_NB_CHICKS;
234:                         break;
235:                     case 1:
236:                         result[i] = BabyBird.DEFAULT_LIFE_CYCLE;
237:                         break;
238:                     case 2:
239:                         result[i] = BabyBird.DEFAULT_MAX_FOOD;
240:                         break;
241:                     case 3:
242:                         result[i] = BabyBird.DEFAULT_HUNTING_RATE;
243:                         break;
244:                 }
245:             }
246:             return result;
247:         } catch (Exception e) {
248:             throw new IllegalArgumentException(e.getMessage() + "\n Please follow usage bellow\n java babybird [chicks [
baby_iter [max_food_size [hunting_success_rate]]]]");
249:         }
250:     }
251:
252:     /**
253:      * Check type and value of arguments
254:      * Also check the number of args
255:      *
256:      * @param index index of the argument
257:      * @param arg   argument to check
258:      * @return value of the argument in int
259:      * @throws Exception if the argument is not correct
260:      */
261:     private int checkArgs(int index, String arg) throws IllegalArgumentException {
262:         if (index >= BabyBird.MAX_ARGS) {
263:             throw new IllegalArgumentException("To many arguments. Max: 4.");
264:         } else {
265:             int value = Integer.parseInt(arg);
266:             if (index == BabyBird.SUCCESS_RATE_INDEX) {
267:                 if (0 > value || value > BabyBird.MAX_HUNTING_RATE)
268:                     throw new IllegalArgumentException("Success rate must be between 0 and 100.");
269:             } else {
270:                 if (0 >= value) throw new IllegalArgumentException("Argument must be bigger than 0.");
271:             }
272:             return value;
273:         }
274:     }
275:
276: }
```

```java
// Copyright 2021, School of Engineering and Architecture of Fribourg
//
// Licensed under the Apache License, Version 2.0 (the "License");
// you may not use this file except in compliance with the License.
// You may obtain a copy of the License at
//
//     http://www.apache.org/licenses/LICENSE-2.0
//
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.

// author = "Simon Barras"
// date = "2021-11-23"
// version = "0.0.1"
// email = "simon.barras@edu.hefr.ch"
// userid = "simon.barras"
//


/**
 * Object for simlating a prent bird.
 * Use a thread with a routine to live.
 */
public class Parent {
    private Ctrl ctrl; // controller of the application

    private String name; // name of the parent
    private Thread spirit; // thread of the parent
    private long speed; // wait time attributed to this parent
    private int huntingRate; // change of funding something
    private int maxHunt; // number max of the hunting's result
    private int huntingResult; // result of the hunting

    public Parent(Ctrl ctrl, int maxHunt, int huntingRate) {
        this.ctrl = ctrl;
        this.maxHunt = maxHunt;
        this.huntingRate = huntingRate;
        this.speed = BabyBird.RND.nextInt(BabyBird.WAIT_BIRD + 1);

        spirit = new Thread(() -> {
            // Routine of the bird's life
            while (ctrl.hasChild()) {
                hunt();
                depositFood();
                rest();
            }
            BabyBird.LOGGER.info(name + " is dying of sorrow");
        });
    }

    /**
     * Start the thread and the routine
     *
     * @param name name of the bird
     */
    public void born(String name) {
        this.name = "Parent " + name;
        huntingResult = 0;
        spirit.start();
    }

    /**
     * Close the thread
     *
     * @throws InterruptedException
     */
    public void funeral() throws InterruptedException {
        spirit.join();
        BabyBird.LOGGER.info(name + " is buried");
    }

    /**
     * Wait a random time and the try to hunt.
     * The probability for the parent to find something is defined in the parameters.
     * The number of food fin is a random number: 0 <= food <= maxHunt.
     */
    private void hunt() {
        BabyBird.LOGGER.info(name + " is hunting");
        ctrl.nap(spirit, speed);
        if (BabyBird.RND.nextInt(BabyBird.MAX_HUNTING_RATE + 1) < huntingRate) {
            huntingResult = BabyBird.RND.nextInt(maxHunt + 1);
            BabyBird.LOGGER.info(name + " found " + huntingResult + " worms");
        }
    }

    /**
     * Put food only if the tank is empty and nobody is using the variable of the tank.
     */
    private void depositFood() {
        if (huntingResult > 0) {
            ctrl.putFood(huntingResult);
            BabyBird.LOGGER.info(name + " is throwing in the nest");
        }
```

```
 97:     }
 98:
 99:     /**
100:      * Random timer
101:      */
102:     private void rest() {
103:         ctrl.nap(spirit, speed);
104:         BabyBird.LOGGER.info(name + " is taking a coffee");
105:     }
106:
107: }
```

Git Logs

```
commit 8f30f07f1bb1c03d7b11f9b650517cdffc69f492 (HEAD -> refs/heads/main,
refs/remotes/origin/main, refs/remotes/origin/HEAD)
Author: Simon Barras <simon.barras02@gmail.com>
Date:   Fri, 3 Dec 2021 17:43:28 +0100

    Reformat code
```

```
commit b7a2e2492cdde15a222e1a5285ff5d0373c02c9f
Author: Simon Barras <simon.barras02@gmail.com>
Date:   Fri, 3 Dec 2021 17:41:52 +0100

    add documentation
```

```
commit 1fa6bdd4199b0212563349f3efbfa31925ae7486
Author: Simon Barras <simon.barras02@gmail.com>
Date:   Fri, 3 Dec 2021 17:20:46 +0100

    Modify log system
```

```
commit a4cdcdced2ce019a323c4ab80da6ae4b3389b116
Author: Simon Barras <simon.barras02@gmail.com>
Date:   Fri, 3 Dec 2021 15:14:52 +0100

    Fix some magics numbers and comments
```

```
commit 89ae6190de860c06681610e37922ee894b22fc25
Author: Simon Barras <simon.barras02@gmail.com>
Date:   Fri, 3 Dec 2021 14:55:39 +0100

    Add some comments
```

```
commit 3ba71893dec19fbfe347515170377b8ce5216e30
Author: Simon Barras <simon.barras02@gmail.com>
Date:   Fri, 3 Dec 2021 14:47:43 +0100

    fix bugs
```

-> what kind of bug ?

```
commit e4aa1370d5a494d210f85fa26ce0446e6e81c5e8
Author: Simon Barras <simon.barras02@gmail.com>
Date:   Fri, 3 Dec 2021 13:15:06 +0100

    add tests

commit e0b3c32006ecda753ebd6e4502c52a52fd4f9ddb
Author: Simon Barras <simon.barras02@gmail.com>
Date:   Fri, 3 Dec 2021 13:14:44 +0100

    Finish application

commit cf305ee3fb3cc3203d15a7bfbdb9d671a3a85799
Author: Simon Barras <66463606+simbarras@users.noreply.github.com>
Date:   Tue, 23 Nov 2021 16:10:37 +0100

    Comment classes

commit 783295a19fbe2a748bf6bc0e79636fc48f4de841
Author: Simon Barras <66463606+simbarras@users.noreply.github.com>
Date:   Tue, 23 Nov 2021 15:33:40 +0100

    Convert lab02 in java

commit d747b66087a24118cc7c1eb6b35df8f793fe993b
Merge: 99eb960 5089076
Author: Simon Barras <66463606+simbarras@users.noreply.github.com>
Date:   Tue, 23 Nov 2021 13:14:37 +0100

    Merge branch 'main' of https://gitlab.forge.hefr.ch/concurp/2021-
2022/concurp-student-labs
```

Run params: 1-1-1-0

```
Lab3 - BabyBird simulation
Lab03 - BabyBird simulation
Parent 2 is hunting
Chick 1 is sleeping
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Chick 1 is getting food
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
```

```
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee

#######
##########################..... 4287 lines skipped ......##########################
#######

Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
```

```
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
gtimeout: sending signal TERM to command ‘java’
```

Run params: 1-1-1-50.2

```
Lab3 - BabyBird simulation
Exception in thread "main" java.lang.IllegalArgumentException: For input string: "50.2"
 Please follow usage bellow
 java babybird [chicks [baby_iter [max_food_size [hunting_success_rate]]]]
at Ctrl.parseArgs(Ctrl.java:248)
at BabyBird.main(BabyBird.java:54)
```

Run params: 1-50-5000-100

```
Lab3 - BabyBird simulation
Lab03 - BabyBird simulation
Lab03 - finish simulation
Chick 1 is sleeping
Parent 1 is hunting
Parent 2 is hunting
Parent 2 found 107 worms
Parent 1 found 2519 worms
Chick 1 is getting food
Parent hunted 2626 food, 2626 will be lost
Parent 1 is throwing in the nest
Parent 2 is throwing in the nest
Chick 1 is eating
Parent 2 is taking a coffee
Parent 2 is hunting
Chick 1 is digesting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 found 775 worms
Parent 1 found 2611 worms
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
```

2626

→ ? I don't understand. Is this just a wrong text?

would be good to print the remaining nbr of worms

```
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
```

```
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
```

```
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is sleeping
Chick 1 is getting food
Chick 1 is eating
Chick 1 is digesting
Chick 1 is leaving the nest
Parent hunted 3386 food, 810 will be lost
Parent 1 is throwing in the nest
Chick 1 is buried
Parent 2 is throwing in the nest
Parent 1 is taking a coffee
Parent 1 is dying of sorrow
Parent 1 is buried
Parent 2 is taking a coffee
Parent 2 is dying of sorrow
Parent 2 is buried
```

Run params: default

```
Lab3 - BabyBird simulation
Lab03 - BabyBird simulation
Lab03 - finish simulation
Chick 10 is sleeping
Parent 1 is hunting
Parent 2 is hunting
Chick 8 is sleeping
Chick 14 is sleeping
Chick 11 is sleeping
Chick 13 is sleeping
Chick 15 is sleeping
Chick 3 is sleeping
Chick 12 is sleeping
Chick 5 is sleeping
Chick 2 is sleeping
Chick 17 is sleeping
Chick 6 is sleeping
Chick 1 is sleeping
Chick 16 is sleeping
Chick 7 is sleeping
Chick 9 is sleeping
Chick 4 is sleeping
Chick 16 is getting food
Chick 9 is getting food
Chick 12 is getting food
Chick 10 is getting food
Chick 15 is getting food
Chick 14 is getting food
Chick 2 is getting food
Chick 3 is getting food
Chick 11 is getting food
Chick 13 is getting food
Chick 7 is getting food
Chick 17 is getting food
Chick 4 is getting food
Chick 5 is getting food
Chick 8 is getting food
Chick 1 is getting food
Chick 6 is getting food
Parent 2 found 0 worms
Parent 1 found 4 worms
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 2 found 0 worms
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 2 found 4 worms
Parent hunted 8 food, 8 will be lost
Parent 1 is throwing in the nest
Chick 2 is eating
Chick 14 is eating
Chick 9 is eating
Chick 15 is eating
```

```
Chick 12 is eating
Chick 16 is eating
Chick 10 is eating
Parent 2 is throwing in the nest
Parent 2 is taking a coffee
Parent 2 is hunting
Chick 2 is digesting
Chick 2 is sleeping
Chick 16 is digesting
Parent 1 is taking a coffee
Parent 1 is hunting
Chick 9 is digesting
Chick 15 is digesting
Chick 15 is sleeping
Parent hunted 8 food, 8 will be lost
Parent 1 is throwing in the nest
Chick 13 is eating
Chick 5 is eating
Chick 4 is eating
Chick 17 is eating
Chick 7 is eating
Chick 11 is eating
Parent 2 is throwing in the nest
Chick 3 is eating
Chick 10 is digesting
Chick 14 is digesting
Chick 16 is sleeping
Chick 12 is digesting
Chick 2 is getting food
Chick 5 is digesting
Chick 13 is digesting
Chick 9 is sleeping
Chick 4 is digesting
Chick 13 is sleeping
Chick 17 is digesting
Chick 7 is digesting
Chick 12 is sleeping
Chick 14 is sleeping
Parent 1 is taking a coffee
Parent 1 is hunting
Chick 15 is getting food
Parent 2 is taking a coffee
Chick 10 is sleeping
Chick 5 is sleeping
Parent 2 is hunting
Chick 3 is digesting
Chick 11 is digesting
Chick 14 is getting food
Chick 13 is getting food
Chick 7 is sleeping
Chick 16 is getting food
Chick 9 is getting food
Chick 4 is sleeping
Parent hunted 8 food, 8 will be lost
Chick 10 is getting food
Parent 1 is throwing in the nest
Chick 13 is eating
Chick 11 is sleeping
Chick 14 is eating
```

```
Chick 15 is eating
Chick 2 is eating
Chick 6 is eating
Chick 1 is eating
Chick 8 is eating
Parent 2 is throwing in the nest
Chick 15 is digesting
Chick 12 is getting food
Chick 17 is sleeping
Chick 3 is sleeping
Chick 6 is digesting
Chick 4 is getting food
Chick 8 is digesting
Chick 5 is getting food
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Chick 2 is digesting
Chick 17 is getting food
Chick 7 is getting food
Chick 8 is sleeping
Chick 6 is sleeping
Chick 15 is sleeping
Chick 13 is digesting
Chick 1 is digesting
Chick 14 is digesting
Chick 11 is getting food
Chick 3 is getting food
Chick 2 is sleeping
Parent hunted 8 food, 8 will be lost
Parent 1 is throwing in the nest
Chick 12 is eating
Chick 1 is sleeping
Chick 5 is eating
Chick 17 is eating
Chick 10 is eating
Chick 4 is eating
Chick 9 is eating
Parent 2 is throwing in the nest
Chick 16 is eating
Parent 1 is taking a coffee
Parent 1 is hunting
Chick 4 is digesting
Chick 13 is sleeping
Chick 1 is getting food
Chick 14 is sleeping
Chick 6 is getting food
Chick 15 is getting food
Chick 9 is digesting
Chick 8 is getting food
Chick 16 is digesting
Chick 17 is digesting
Chick 14 is getting food
Chick 12 is digesting
Chick 13 is getting food
Chick 9 is sleeping
Chick 5 is digesting
Parent 2 is taking a coffee
```

```
Parent 2 is hunting
Chick 4 is sleeping
Chick 2 is getting food
Chick 10 is digesting
Chick 12 is sleeping
Chick 17 is sleeping
Chick 16 is sleeping
Chick 12 is getting food
Chick 16 is getting food
Chick 4 is getting food
Chick 9 is getting food
Chick 17 is getting food
Chick 10 is sleeping
Parent hunted 8 food, 8 will be lost
Parent 2 is throwing in the nest
Chick 7 is eating
Chick 6 is eating
Chick 5 is sleeping
Chick 8 is eating
Chick 15 is eating
Chick 1 is eating
Chick 3 is eating
Chick 11 is eating
Parent 1 is throwing in the nest
Chick 1 is digesting

#######
##########################..... 4654 lines skipped ......##########################
#######

Parent 2 is taking a coffee
Parent 2 is hunting
Chick 14 is sleeping
Chick 12 is digesting
Chick 10 is getting food
Chick 9 is digesting
Chick 17 is digesting
Chick 12 is sleeping
Chick 5 is digesting
Chick 13 is getting food
Chick 8 is digesting
Parent 2 found 4 worms
Chick 6 is getting food
Parent hunted 11 food, 11 will be lost
Parent 2 is throwing in the nest
Chick 11 is eating
Chick 10 is eating
Chick 6 is eating
Chick 13 is eating
Parent 1 is throwing in the nest
Chick 3 is eating
Chick 17 is leaving the nest
Chick 9 is sleeping
Chick 5 is sleeping
Chick 8 is sleeping
Chick 3 is digesting
Chick 9 is getting food
Chick 9 is eating
Chick 11 is digesting
```

```
Chick 14 is getting food
Chick 14 is eating
Parent 2 is taking a coffee
Chick 12 is getting food
Chick 6 is digesting
Parent 2 is hunting
Chick 3 is sleeping
Chick 5 is getting food
Chick 8 is getting food
Chick 11 is sleeping
Chick 13 is digesting
Parent 1 is taking a coffee
Parent 1 is hunting
Chick 14 is digesting
Chick 10 is digesting
Parent 2 found 7 worms
Chick 3 is getting food
Chick 13 is sleeping
Chick 9 is digesting
Chick 6 is leaving the nest
Parent hunted 14 food, 14 will be lost
Parent 1 is throwing in the nest
Chick 5 is eating
Chick 8 is eating
Parent 2 is throwing in the nest
Chick 12 is eating
Chick 3 is eating
Chick 10 is sleeping
Chick 11 is getting food
Chick 9 is leaving the nest
Chick 14 is sleeping
Chick 11 is eating
Chick 3 is digesting
Parent 2 is taking a coffee
Parent 2 is hunting
Chick 8 is digesting
Chick 11 is digesting
Parent 1 is taking a coffee
Parent 1 is hunting
Parent 2 found 6 worms
Chick 3 is sleeping
Chick 14 is getting food
Chick 13 is getting food
Chick 13 is eating
Chick 14 is eating
Chick 10 is getting food
Chick 12 is digesting
Chick 14 is digesting
Chick 3 is getting food
Chick 8 is sleeping
Chick 14 is sleeping
Parent hunted 13 food, 13 will be lost
Parent 1 is throwing in the nest
Chick 11 is leaving the nest
Chick 3 is eating
Chick 5 is digesting
Parent 2 is throwing in the nest
Chick 10 is eating
Chick 13 is digesting
```

```
Chick 14 is getting food
Chick 14 is eating
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Chick 8 is getting food
Chick 8 is eating
Chick 3 is digesting
Chick 12 is sleeping
Chick 10 is digesting
Parent 1 found 4 worms
Chick 5 is sleeping
Chick 13 is leaving the nest
Chick 14 is digesting
Parent 2 found 3 worms
Chick 10 is sleeping
Chick 12 is getting food
Chick 12 is eating
Chick 3 is leaving the nest
Chick 8 is digesting
Chick 5 is getting food
Chick 5 is eating
Chick 14 is leaving the nest
Chick 10 is getting food
Chick 10 is eating
Parent hunted 7 food, 7 will be lost
Parent 2 is throwing in the nest
Parent 1 is throwing in the nest
Chick 8 is sleeping
Chick 5 is digesting
Chick 12 is digesting
Chick 5 is sleeping
Chick 12 is sleeping
Chick 10 is digesting
Parent 2 is taking a coffee
Parent 2 is hunting
Parent 1 is taking a coffee
Parent 1 is hunting
Chick 12 is getting food
Chick 12 is eating
Chick 5 is getting food
Chick 5 is eating
Chick 8 is getting food
Chick 8 is eating
Parent 2 found 7 worms
Parent 1 found 6 worms
Chick 12 is digesting
Chick 10 is sleeping
Chick 5 is digesting
Chick 10 is getting food
Chick 10 is eating
Chick 8 is digesting
Chick 12 is sleeping
Chick 5 is sleeping
Chick 10 is digesting
Chick 5 is getting food
Chick 5 is eating
Chick 10 is sleeping
```

```
Chick 8 is sleeping
Chick 12 is getting food
Chick 12 is eating
Chick 5 is digesting
Chick 10 is getting food
Chick 10 is eating
Parent hunted 13 food, 13 will be lost
Parent 1 is throwing in the nest
Parent 2 is throwing in the nest
Chick 10 is digesting
Parent 1 is taking a coffee
Parent 1 is hunting
Chick 10 is leaving the nest
Chick 8 is getting food
Chick 8 is eating
Chick 12 is digesting
Parent 2 is taking a coffee
Parent 2 is hunting
Chick 5 is leaving the nest
Parent 1 found 6 worms
Chick 8 is digesting
Chick 12 is leaving the nest
Chick 8 is sleeping
Chick 8 is getting food
Chick 8 is eating
Chick 8 is digesting
Chick 8 is leaving the nest
Parent hunted 13 food, 8 will be lost
Parent 1 is throwing in the nest
Parent 2 is throwing in the nest
Chick 1 is buried
Chick 2 is buried
Chick 3 is buried
Chick 4 is buried
Chick 5 is buried
Chick 6 is buried
Chick 7 is buried
Chick 8 is buried
Chick 9 is buried
Chick 10 is buried
Chick 11 is buried
Chick 12 is buried
Chick 13 is buried
Chick 14 is buried
Chick 15 is buried
Chick 16 is buried
Chick 17 is buried
Parent 2 is taking a coffee
Parent 2 is dying of sorrow
Parent 1 is taking a coffee
Parent 1 is dying of sorrow
Parent 1 is buried
Parent 2 is buried
```

Baby bird java

For this new lab, we need to reuse the code from the previous lab (documentation here). The main job is to translate the code from python to java but the tricky part was to implement the barrier.

Barier

We need to implement this barrier to complete this fact:

> *One parent waits until the other parent comes with or without (hunting chance!) food. They must deposit their food together in the nest. Of course, the sum of the food of both parents must not excite the maximum food capacity `C`. Possible excess food is disposed by the parents (this food-waste must be indicated in the simulation logs!)*

To modify my code, I didn't touch to the class `Chick` and `Parent`. I only need to change the method `putFodd` to add an `await()` statement:

```
/**
 * Class to add some food
 *
 * @param worm number of worms hunted
 */
public void putFood(int worm) {
    if (hasChild()) {
        synchronized (preNbWorm) {
            preNbWorm += worm;
        }
        try {
            barrier.await();
        } catch (InterruptedException e) {
            e.printStackTrace();
        } catch (BrokenBarrierException e) {
            BabyBird.LOGGER.info("Barrier broken");
        }
    }

}
```

The real modification of the tank's value is in the release of the barrier.

```
/**
 * Initialize the nest.
 *
 * @param nbChicks     number of chicks
 * @param babyItr      number of life cycle
 * @param foodCapacity max number of worms in the nest
```

```
 * @param huntingRate  chance to find something
 */
private void building_nest(int nbChicks, int babyItr, int foodCapacity, int huntingRate) {
    [...]
    barrier = new CyclicBarrier(BabyBird.NB_PARENTS, () -> {
        try {
            // food put in the nest by the barrier
            waitHunt.acquire();
            int lostWorm = preNbWorm - nbWorm;
            BabyBird.LOGGER.info("Parent hunted " + preNbWorm + " food, " + (lostWorm > 0 ?
            wormTank(preNbWorm > nbMaxWorm ? nbMaxWorm : preNbWorm);
            preNbWorm = 0;
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    });

   [...]
}
```

Conclusion

This is not my favorite lab, maybe because I don't like to do 2 times the same job. But I think it's a very good lab and I think he will be very useful. The last lab I expected the 6 but I got 5,5, I try to learn of my fault and I think I patch all the mistakes.