

CNT 4714 – Project 2 – Spring 2021

Title: “Project 2: An Application Employing Synchronized/Cooperating Multiple Threads In Java Using Locks – A Banking Simulator”

Points: 100 points

Due Date: Sunday February 14, 2021 by 11:59 pm (WebCourses time)

Objectives: To practice programming cooperating, synchronized multiple threads of execution.

Description: In this programming assignment you will simulate the deposits and withdrawals made to a fictitious bank account (I’ll let you use my real bank account if you promise to make only deposits! ☺). In this case the deposits and withdrawals will be made by synchronized threads. Synchronization is required for two reasons – (1) mutual exclusion (updates cannot be lost) and (2) because a withdrawal cannot occur if the amount of the withdrawal request is greater than the current balance in the account. This means that access to the account (the shared object) must be synchronized. This application requires cooperation and communication amongst the various threads (cooperating synchronized threads). (In other words, this problem is similar to the producer/consumer problem where there is more than one producer and more than one consumer process active simultaneously.) If a withdrawal thread attempts to withdraw an amount greater than the current balance in the account – then it must block itself and wait until a deposit has occurred before it can try again. As we covered in the lecture notes, this will require that the depositor threads signal all waiting withdrawal threads whenever a deposit is completed.

1. You should have **five depositor** threads and **nine withdrawal** threads simultaneously executing.
2. To keep things relatively simple, as well as to see immediate results from a series of transactions (deposits and withdrawals), assume that deposits are made in amounts ranging from \$1 to \$250 (whole dollars only) and withdrawals are made in amounts ranging from \$1 to \$50 (again, whole dollars only). Since we have more withdrawal threads than depositor threads, the account balance should constantly decrease over time.
3. Once a depositor thread has executed, put it to sleep for few milliseconds (randomly generate this number – don’t use a constant sleep time) or so (depends a little bit on the speed of your system as to how long you will want

to sleep the depositor threads - basically we want to ensure a lot more withdrawals than deposits) to allow other threads to execute. This is the only situation in which a deposit thread will block.

4. For withdrawal threads, things will be a bit different depending on whether you are working on a single or multi-core processor.
 - a. For single core processors, once a withdrawal thread has executed, have it yield to another thread. Since the thread is giving up the processor voluntarily, it will be unlikely to run again (attempt a second withdrawal in a row), before another thread runs. Note however, that it does not prevent it from running again, if all other withdrawal threads are blocked and all depositors are sleeping, it will run again. So occasional back-to-back runs of withdrawal threads might occur.
 - b. For multi-core processors, once a withdrawal thread has executed, have it sleep for some random period of time (again, a few milliseconds should be fine). Depending on which core a thread is executing, yielding the CPU won't ensure that the same thread will not run again immediately. While, sleeping the thread will also not ensure that it will not run two or more times in succession, it is less likely to do so in the multi-core environment.
 - c. What we don't want to happen is a single withdrawal thread gaining the CPU and then executing a long sequence of withdrawal operations. Recall though that withdrawal threads block if they attempt to withdraw more than the current balance in the account.
 - d. Similarly, we don't want depositor threads monopolizing the CPU either and causing the balance in the account to grow continuously. This would most likely occur when the withdrawal threads are sleeping too long in comparison to the average sleep time of the deposit threads. See page 7 for an illustration of this.
5. Assume all threads have the same priority. Do not give different priority to depositor and withdrawal threads.
6. The output from your program must look reasonably similar to the sample output shown below.
7. **Do not put the threads into a counted loop for your simulation.** In other words, the `run()` method should be an infinite loop. Just stop the simulation from your IDE after a few seconds.

8. **Do not use the Java synchronized statement.** I want you to handle the locking and signaling yourself. No monitors!
9. You must utilize a reentrant lock from the `java.util.concurrent.locks` package for implementing your locking protocols. We will specify no fairness policy for this application. **Do not create your own lock using a Boolean or any other type of variable.**

References:

Notes: Lecture Notes for Multithreaded Applications.

Restrictions:

Your source files shall begin with comments containing the following information:

```
/* Name:  
   Course: CNT 4714 Spring 2021  
   Assignment title: Project 2 – Synchronized, Cooperating Threads Under Locking  
   Due Date: February 14, 2021  
*/
```

Input Specification: Internal to the program.

Output Specification: Console based. Your output should appear reasonably similar to the output shown below.

Deliverables:

- (1) Zip up your entire project folder from your IDE and submit it via WebCourses no later than 11:59pm Sunday February 14, 2021.
- (2) Include at least one screen shot which illustrates the execution of your synchronized threaded application. See below for some representative examples. You can either do a screen shot of the console window like I did below or redirect your output to a file and take a screen shot from an editor.

Additional Information:

Shown below are three example screen shots of the output from this program to help illustrate how your application is to operate and display the results. The last page illustrates execution runs that you do not want to produce.

```

eclipse-workspace-2020-12 - CNT 4714 - Project 2 - Spring 2021/src/ABankAccount.java

Console
<terminated> ABankingSimulator [Java Application] /Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/java (Jan 30, 2021, 7:38:50 PM - 7:38:53 PM)

Deposit Threads      Withdrawal Threads      Balance
-----
Thread D1 deposits $119
Thread D2 deposits $12

Thread D3 deposits $124
Thread D4 deposits $3
Thread D5 deposits $96

Thread D5 deposits $222
Thread D2 deposits $22
Thread D1 deposits $142

Thread W3 withdraws $12
Thread W4 withdraws $9
Thread W5 withdraws $37
Thread W6 withdraws $31
Thread W2 withdraws $19
Thread W1 withdraws $4
Thread W8 withdraws $33
Thread W7 withdraws $48
Thread W9 withdraws $33
Thread W8 withdraws $33
Thread W6 withdraws $33
Thread W7 withdraws $6
Thread W6 withdraws $5
Thread W8 withdraws $45
Thread W2 withdraws $21
Thread W5 withdraws $42
Thread W9 withdraws $1
Thread W4 withdraws $31
Thread W1 withdraws $42
Thread W3 withdraws $34
Thread W9 withdraws $11
Thread W6 withdraws $40
Thread W7 withdraws $4
Thread W8 withdraws $19
Thread W7 withdraws $22
Thread W9 withdraws $15
Thread W3 withdraws $34
Thread W4 withdraws $3
Thread W6 withdraws $30
Thread W8 withdraws $40
Thread W5 withdraws $44
Thread W6 withdraws $21
Thread W5 withdraws $13
Thread W2 withdraws $13
Thread W9 withdraws $12
Thread W1 withdraws $30
Thread W7 withdraws $19
Thread W4 withdraws $2
Thread W9 withdraws $29
Thread W6 withdraws $19
Thread W8 withdraws $3
Thread W1 withdraws $1
Thread W4 withdraws $16
Thread W3 withdraws $35
Thread W2 withdraws $20
Thread W5 withdraws $33
Thread W1 withdraws $43
Thread W8 withdraws $14
Thread W6 withdraws $44
Thread W1 withdraws $14

(+) Balance is $119
(+) Balance is $131
(-) Balance is $119
(-) Balance is $110
(-) Balance is $73
(-) Balance is $42
(+) Balance is $166
(-) Balance is $147
(-) Balance is $143
(+) Balance is $146
(-) Balance is $113
(-) Balance is $65
(+) Balance is $161
(-) Balance is $128
(-) Balance is $95
(-) Balance is $62
(-) Balance is $56
(-) Balance is $51
(-) Balance is $6
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(-) Balance is $5
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(-) Balance is $1
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(+) Balance is $223
(-) Balance is $208
(-) Balance is $174
(-) Balance is $171
(-) Balance is $141
(-) Balance is $101
(-) Balance is $57
(-) Balance is $36
(-) Balance is $23
(-) Balance is $10
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(+) Balance is $32
(-) Balance is $2
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(-) Balance is $0
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
(+) Balance is $142
(-) Balance is $99
(-) Balance is $85
(-) Balance is $41
(-) Balance is $27

```

Two withdrawal threads blocked due to insufficient funds.

```
eclipse-workspace-2020-12 - CNT 4714 - Project 2 - Spring 2021/src/ABankAccount.java

<terminated> ABankingSimulator [Java Application] /Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/java
Thread D3 deposits $240      (+) Balance is $230
Thread W3 withdraws $6      (-) Balance is $250
Thread W2 withdraws $24      (-) Balance is $226
Thread W6 withdraws $33      (-) Balance is $193
Thread W5 withdraws $40      (-) Balance is $153
Thread W7 withdraws $50      (-) Balance is $103
Thread W2 withdraws $48      (-) Balance is $55
Thread W8 withdraws $33      (-) Balance is $22
Thread W1 withdraws $10      (-) Balance is $12
Thread D2 deposits $114      (+) Balance is $126
Thread W3 withdraws $31      (-) Balance is $95
Thread W3 withdraws $30      (-) Balance is $65
Thread W3 withdraws $44      (-) Balance is $21
Thread W4 withdraws $25      (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W9 withdraws $32      (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W6 withdraws $37      (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W3 withdraws $31      (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W8 withdraws $39      (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W5 withdraws $14      (-) Balance is $7
Thread W2 withdraws $10      (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W1 withdraws $15      (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W7 withdraws $11      (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W5 withdraws $31      (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread D4 deposits $202      (+) Balance is $209
Thread W1 withdraws $8        (-) Balance is $201
Thread W2 withdraws $5        (-) Balance is $196
Thread W9 withdraws $30      (-) Balance is $166
Thread W6 withdraws $39      (-) Balance is $127
Thread W4 withdraws $33      (-) Balance is $94
Thread W9 withdraws $12      (-) Balance is $82
Thread W8 withdraws $24      (-) Balance is $58
Thread W7 withdraws $30      (-) Balance is $28
Thread W1 withdraws $4        (-) Balance is $24
Thread D5 deposits $143      (+) Balance is $167
Thread W9 withdraws $19      (-) Balance is $148
Thread W5 withdraws $49      (-) Balance is $99
```

Withdrawal thread W3 runs three times in a row. This is ok. It may happen from time to time, just not too frequently.

```
eclipse-workspace-2020-12 - CNT 4714 - Project 2 - Spring 2021/src/ABankAccount.java - Eo

Console Open a Terminal (^⌘T)

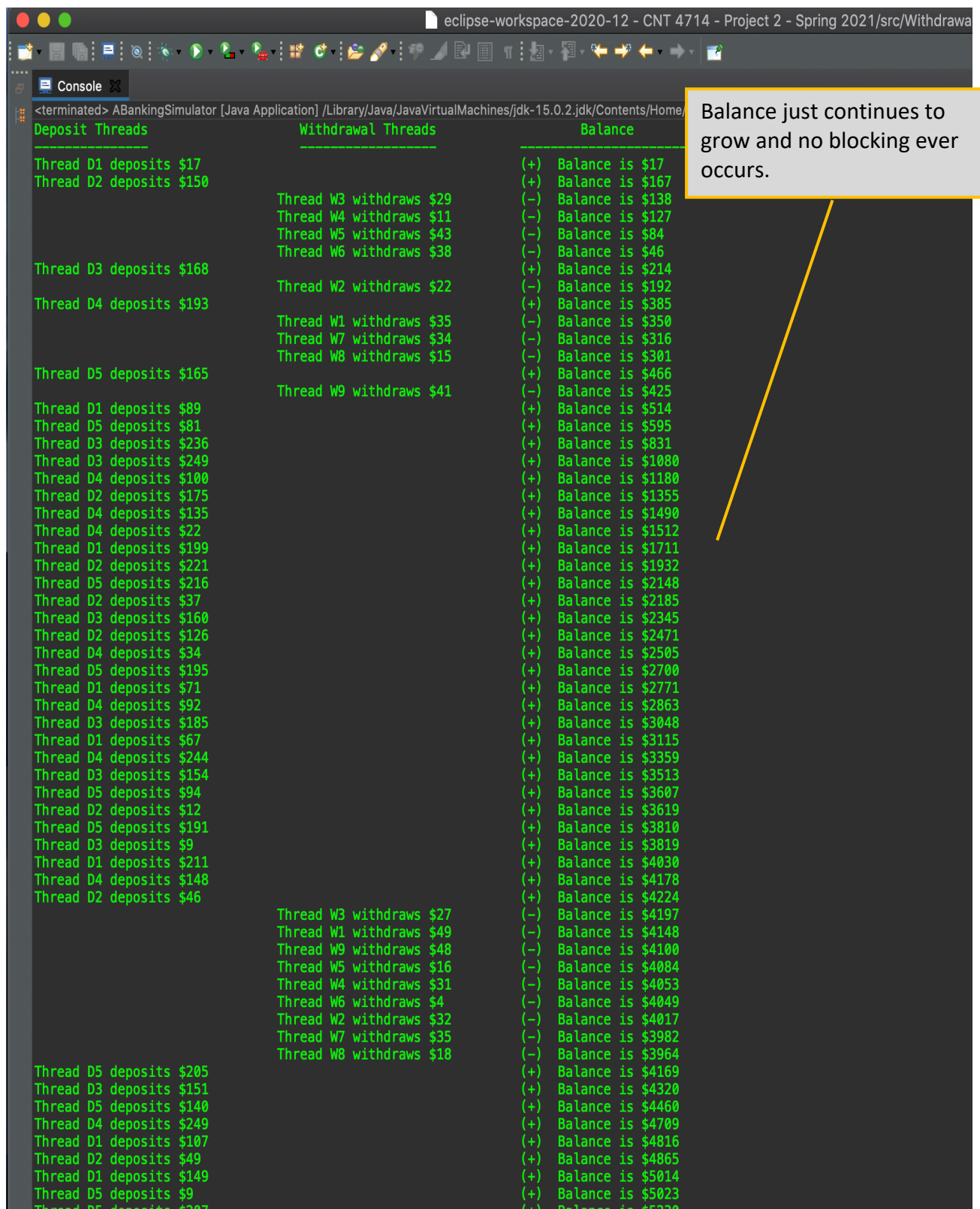
<terminated> ABankingSimulator [Java Application] /Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/java (Jan 30, 2021, 7:44:38 PM - 7:44:53 PM)

Thread W4 withdraws $1      (-) Balance is $13
Thread W4 withdraws $44     (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W3 withdraws $11     (-) Balance is $2
Thread D5 deposits $192     (+) Balance is $194
Thread W8 withdraws $26     (-) Balance is $168
Thread D3 deposits $92     (+) Balance is $260
Thread W5 withdraws $27     (-) Balance is $233
Thread W5 withdraws $9      (-) Balance is $224
Thread W4 withdraws $43     (-) Balance is $181
Thread W1 withdraws $39     (-) Balance is $142
Thread W9 withdraws $18     (-) Balance is $124
Thread W2 withdraws $30     (-) Balance is $94
Thread W9 withdraws $11     (-) Balance is $83
Thread W6 withdraws $50     (-) Balance is $33
Thread W3 withdraws $26     (-) Balance is $7
Thread W8 withdraws $11     (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W1 withdraws $44     (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W2 withdraws $33     (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W7 withdraws $21     (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W3 withdraws $40     (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W5 withdraws $32     (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W4 withdraws $32     (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W6 withdraws $19     (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W9 withdraws $6      (-) Balance is $1
Thread W9 withdraws $15     (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread D5 deposits $173     (+) Balance is $174
Thread W6 withdraws $30     (-) Balance is $144
Thread W8 withdraws $14     (-) Balance is $130
Thread W6 withdraws $50     (-) Balance is $80
Thread D4 deposits $192     (+) Balance is $272
Thread W9 withdraws $24     (-) Balance is $248
Thread W9 withdraws $16     (-) Balance is $232
Thread W5 withdraws $26     (-) Balance is $206
Thread W4 withdraws $42     (-) Balance is $164
Thread W2 withdraws $24     (-) Balance is $140
Thread W1 withdraws $26     (-) Balance is $114
Thread W3 withdraws $4      (-) Balance is $110
Thread W7 withdraws $25     (-) Balance is $85
```

All withdrawal threads are blocked. No withdrawal thread can run until a depositor thread runs.

Thread W8 withdraws \$11 (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W1 withdraws \$44 (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W2 withdraws \$33 (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W7 withdraws \$21 (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W3 withdraws \$40 (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W5 withdraws \$32 (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W4 withdraws \$32 (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W6 withdraws \$19 (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!
Thread W9 withdraws \$6 (-) Balance is \$1
Thread W9 withdraws \$15 (*****) WITHDRAWAL BLOCKED - INSUFFICIENT FUNDS!!!

We don't want to see this sort of scenario where the depositors are monopolizing the account. Indication is the depositor threads aren't sleeping long enough or the withdrawal threads are sleeping too long.



```

eclipse-workspace-2020-12 - CNT 4714 - Project 2 - Spring 2021/src/Withdrawa
Console
<terminated> ABankingSimulator [Java Application] /Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home
Deposit Threads      Withdrawal Threads      Balance
Thread D1 deposits $17
Thread D2 deposits $150
Thread D3 deposits $168
Thread D4 deposits $193
Thread D5 deposits $165
Thread D1 deposits $89
Thread D5 deposits $81
Thread D3 deposits $236
Thread D3 deposits $249
Thread D4 deposits $100
Thread D2 deposits $175
Thread D4 deposits $135
Thread D4 deposits $22
Thread D1 deposits $199
Thread D2 deposits $221
Thread D5 deposits $216
Thread D2 deposits $37
Thread D3 deposits $160
Thread D2 deposits $126
Thread D4 deposits $34
Thread D5 deposits $195
Thread D1 deposits $71
Thread D4 deposits $92
Thread D3 deposits $185
Thread D1 deposits $67
Thread D4 deposits $244
Thread D3 deposits $154
Thread D5 deposits $94
Thread D2 deposits $12
Thread D5 deposits $191
Thread D3 deposits $9
Thread D1 deposits $211
Thread D4 deposits $148
Thread D2 deposits $46
Thread D5 deposits $205
Thread D3 deposits $151
Thread D5 deposits $140
Thread D4 deposits $249
Thread D1 deposits $107
Thread D2 deposits $49
Thread D1 deposits $149
Thread D5 deposits $9
Thread D5 deposits $207
Thread W3 withdraws $29
Thread W4 withdraws $11
Thread W5 withdraws $43
Thread W6 withdraws $38
Thread W2 withdraws $22
Thread W1 withdraws $35
Thread W7 withdraws $34
Thread W8 withdraws $15
Thread W9 withdraws $41
Thread W3 withdraws $27
Thread W1 withdraws $49
Thread W9 withdraws $48
Thread W5 withdraws $16
Thread W4 withdraws $31
Thread W6 withdraws $4
Thread W2 withdraws $32
Thread W7 withdraws $35
Thread W8 withdraws $18
(+) Balance is $17
(+) Balance is $167
(-) Balance is $138
(-) Balance is $127
(-) Balance is $84
(-) Balance is $46
(+) Balance is $214
(-) Balance is $192
(+) Balance is $385
(-) Balance is $350
(-) Balance is $316
(-) Balance is $301
(+) Balance is $466
(-) Balance is $425
(+) Balance is $514
(+) Balance is $595
(+) Balance is $831
(+) Balance is $1080
(+) Balance is $1180
(+) Balance is $1355
(+) Balance is $1490
(+) Balance is $1512
(+) Balance is $1711
(+) Balance is $1932
(+) Balance is $2148
(+) Balance is $2185
(+) Balance is $2345
(+) Balance is $2471
(+) Balance is $2505
(+) Balance is $2700
(+) Balance is $2771
(+) Balance is $2863
(+) Balance is $3048
(+) Balance is $3115
(+) Balance is $3359
(+) Balance is $3513
(+) Balance is $3607
(+) Balance is $3619
(+) Balance is $3810
(+) Balance is $3819
(+) Balance is $4030
(+) Balance is $4178
(+) Balance is $4224
(-) Balance is $4197
(-) Balance is $4148
(-) Balance is $4100
(-) Balance is $4084
(-) Balance is $4053
(-) Balance is $4049
(-) Balance is $4017
(-) Balance is $3982
(-) Balance is $3964
(+) Balance is $4169
(+) Balance is $4320
(+) Balance is $4460
(+) Balance is $4709
(+) Balance is $4816
(+) Balance is $4865
(+) Balance is $5014
(+) Balance is $5023
(+) Balance is $5230

```

Balance just continues to grow and no blocking ever occurs.