Atomic Operations Education Resource

Overview

This activity aims to teach students about the need for consistency and atomic blocks through the use of transactions. Consistency and atomic blocks are the idea that threads should only be able to access data at the right times and in the correct order to complete their task corretly.

Suitable For

Higher/Advanced Higher Computer Science

Key Concepts

The use of transactions, multi-threading, parallelism, basic understanding of dictionaries, consistency and atomic sections.

Learning Intentions

* Remember what atomic operations are.
* Apply the knowledge learned about atomic operations in order to put the bank transactions in an order that will generate the correct bank balance.

How you are learning – recipe to complete a task

* Put a series of transactions in order that makes them atomic, using read-modify-write routine
* Identify why would a non-atomic sequence of instructions result into a wrong end state.

Time Required

15-20 minutes, introductory

Preparation

1. Print out the blocks on the last 9 pages on separate pieces of different coloured paper to show the separation of the transactions. Cut out the blocks, each block can be placed above or below each other in a line.

Prior Learning Assumed

None - this is a new topic.

Outline of Activity

1. Outline to the class that in order for banks to keep track of the bank accounts correctly they have to make sure that actions are done in the right order otherwise mistakes are made.
2. Outline that more than one transaction (a series of actions) can happen at one time, for example someone might be withdrawing £100 out for their gas bill, and at the same time £200 is coming out for their electric bill. If these are done in the wrong order then mistakes might happen and the bank account could hold an unexcpeted value. Show the class the set of green bloc ks (pin them up on the board). Outline that these blocks are the transaction outlined for John’s bank account to have his £200 electricity bill being taken out. These are called read-modify-write operations, where you read an initial value, then change it in some way, then write back the new result.
3. Using all the pink and green blocks on the board, interleaved them one pink after one green. (1,4,2,5,3,6). Outline that if the two transactions are interleaved, they can hold different values at the end of the transactions running. Explain that this is wrong and not atomic.
4. Introduce a concept of write dependency by taking the interleaved blocks and noting that the correct value of the bank account in the end is only achieved if the data dependency between instructions in different atomic transactions is respected.
5. Introduce functional dependency(10,11,12) by using the red set of transactions and show that the atomic transaction must not succeed, because the account would end up in debt, depending on previously covered material you might illustrate that this needs to be reverted because a post-condition does not hold.
6. Illustrate allowed interleaving by using transactions on 2 independent accounts (1,7,2,8,9,3), note that there is no data dependency between seperate accounts.
7. Ask the students to suggest an order that will make the instructions work from 3. work. By putting the transactions in order, this is atomic and it means that there is only ever going to be one correct value for the bank balance at the end of both of them. If you don’t read or write the transactions correctly, then the two transactions are going to cause problems within the bank account.

The correct order is the three green/three pink and then the other three next.

1. Finalise by explaining that this is the way atomic operations are set up, so that you don’t have overlapping values for example mistakes being made in the bank balances.

The blocks can be used in other ways to demonstrate that the transactions should be linked correctly, asking students to arrange the blocks in different ways, and recording the transactions output to show the different results that can happen – however it always needs to go back that the read-modify-write routine is the correct way to use these instructions.