Programming Languages: Program Assignments in Ada

Jan Daciuk

November 23, 2021

Organization of the Course

In the second half of the semester, there will be two programming languages on schedule:

1. Ada
2. Smalltalk

Each language part can be divided into two classes:

1.
   - Introduction
   - Specification of the exercises
2.
   - Checking the exercises done at home (up to 7 points)
   - Extending the exercise done at home according to the teacher's instructions, done in the lab (up to 8 points)

Ada Asignments

The assignments must fulfill the following conditions:

1. they should teach significant features of a programming language,
2. they should be different for different students to prevent copying.

A significant feature of Ada is the *rendez-vous* mechanism. Those *rendez-vous* have variants that can be used to differentiate assignments. Another mechanism will be used for extending the program.

Ada Asignments

ASCII code of the third letter of the family name stripped of any diacritics should be divided by four. The remainder determines the variant of the select instruction:

0. selective accept
1. timed entry call
2. conditional entry call
3. asynchronous select

Ada Asignments

The example Ada program provided inside the first Ada program asignment at eNauczanie contains producers tasks (processes), consumers tasks, and the buffer task. It can be modified to accomplish the asignment given by the teacher. It does not have sufficient synchronization, e.g. when the buffer is full, the product that is being delivered there is lost. When there are not enough products for an assembly, an empty assembly number 0 is sent. That must happen in the target program. The program must use the select mechanism determined by the character code. It must be accompanied by a *legend*, i.e. what the program simulates by printing various messages — as a copy prevention measure. There are many possibilities, e.g.:

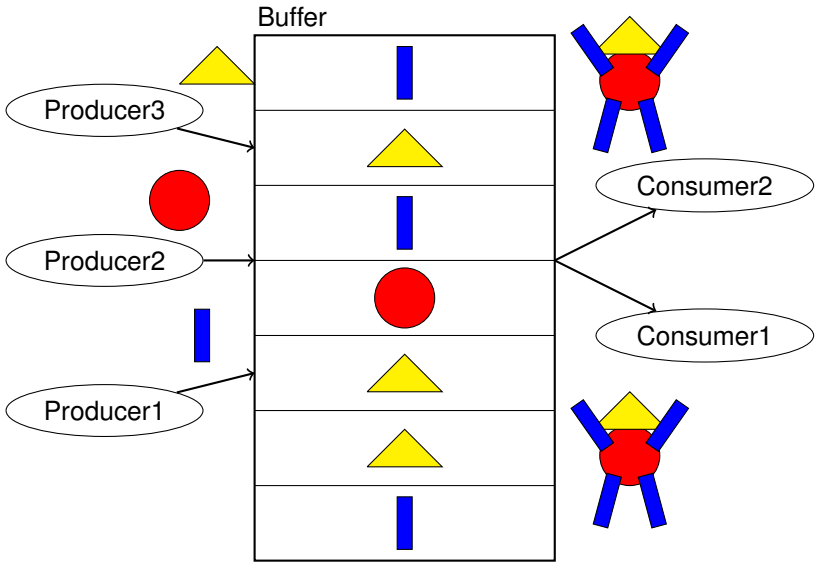
1. production of something, e.g. cars, tv sets, furniture, etc.
2. services, e.g. canteen/restaurant, laundry, etc.
3. communications (protocols)
4. transport, e.g. trains, tramways on rails, starting planes, etc.

Ada Asignments

The program should:

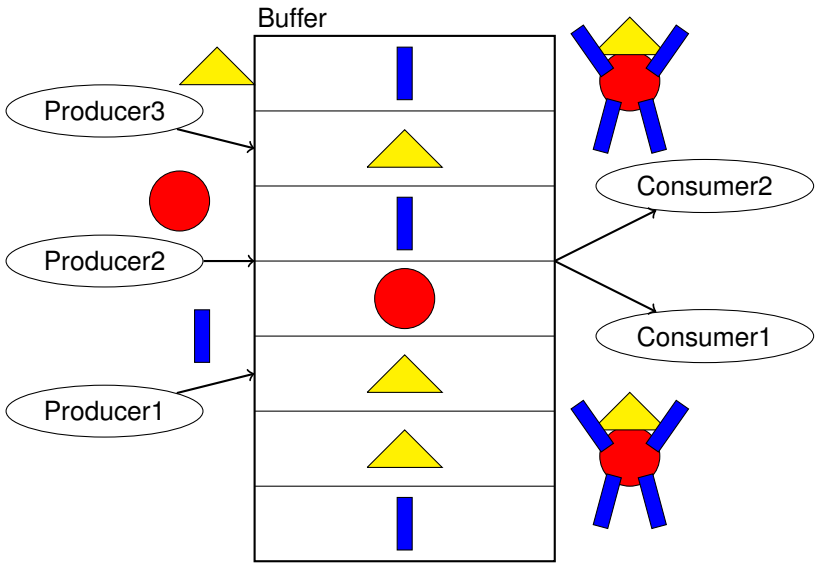
1. contain the specified select instruction variant,
2. show its usage (`select` instruction cannot be executed in the same way each time),
3. make it possible to trace its execution by following messages that explain what happens in the program (e.g. I bake a cookie number 7), especially the flow of control in the chosen synchronisation mechanism,
4. be developed individually at home,
5. be delivered for the teacher using eNauczanie platform not later than the beginning of the class (for the program developed at home), and not later than the end of the class (for the extended program).

Synchronisation



- Producers produce parts at random time, and send them to the buffer.
- One part may be produced by more than one producer.
- If there is room, the buffer accepts the part, and puts it on a ”shelf”.
- A consumer places an order for an assembly at random times.
- Various consumers can order the same or different assemblies.
- When there are parts, the buffer handles the order by assembling the parts from the shelves and sending the assembly to the consumer.

Synchronisation

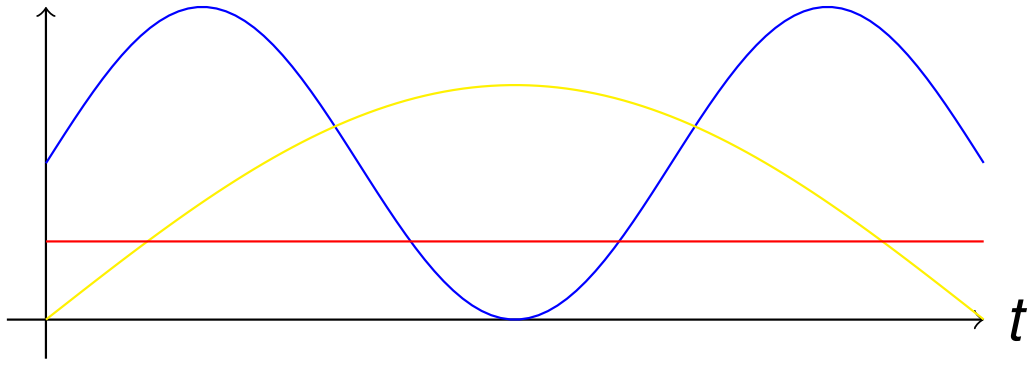


- When there is no room in the buffer, the producer waits.
- When there are not enough parts in the buffer, the consumer waits.
- The buffer cannot allow for a deadlock: nothing can be put or taken from the buffer.

The buffer can accept a part if there are enough parts for one assembly, or if there is enough room for them.

Too Simple Synchronisation

It is tempting to take parts in groups for the whole assemblies. That is not a good solution. Producers may have various production schedules.



One should not slow down the system by taking to and sending out parts from the buffer sequentially one after another, as it is in the example program. One should use a select instruction there for concurrent operation.

Grading

The program must compile. It should be delivered at eNauczanie before the class. It should be contained in **one** file with the source program. The name of the file should contain only **lowercase** letters and digits.

One can get negative points for (among other things):

- execution errors,
- lack of control flow though both branches of select instruction,
- lack of messages explaining the program execution, including the passage of control through both select branches,
- too slow or too fast display of messages in a way that makes it difficult to check the program in appropriate time,
- incorrect synchronization, e.g. loosing items or slowing down the system.