

Example Solutions for Programming Languages G22.2110 Summer 2007 hw10

Assigned Th 7/26/2007, no due date.

These are example solutions. Please keep in mind that often, there is not just one correct solution to a question. If you come up with different answers to the homework, then it may be that both your answers and these answers here are correct. Of course, these answers here may also contain mistakes; if you spot some, please let us know so we can correct them.

1. Concurrency (0 points)

Consider the following (buggy) Java shared list class:

```
class SharedList {
    private static class Cell {
        int data = 0;
        Cell next = null;
    }
    private Cell head = null;
    public void put(int newData) {
        Cell oldHead = head;
        head = new Cell();
        head.data = newData;
        head.next = oldHead;
    }
    public int get() {
        while (null == head) { }
        int result = head.data;
        head = head.next;
        return result;
    }
}
```

- 1a. Assume that the list starts out empty, and a producer thread `p` calls `put(123)`, getting as far as shown in the following table before a consumer thread `c` gets scheduled instead.

head	head.data	head.next	oldHead	result	thread: code
null	(n/a)	(n/a)	(n/a)	(n/a)	p: Cell oldHead = head
			null		p: head = new Cell()
new	0	null			p: head.data = newData

Show an interleaving that eventually causes a `NullPointerException`.

<i>Example solutions</i>					
head	head.data	head.next	oldHead	result	thread: code
null	(n/a)	(n/a)	(n/a)	(n/a)	p: Cell oldHead = head
			null		p: head = new Cell()
new	0	null			p: head.data = newData
new	123				c: while (null == head) { }
				123	c: int result = head.data
					c: head = head.next
null	(n/a)	(n/a)			p: head.next = oldHead

1b. Fix the code to prevent the race condition.

<i>Example solutions</i>
The fix consists of putting a synchronized modifier in front of the declarations of both methods, put and get.
<pre>public synchronized void put(int newData) { ... } public synchronized int get() { ... }</pre>

1b. Fix the code to prevent deadlock.

<i>Example solutions</i>
The fix consists of replacing the empty while loop body with a call to <code>wait()</code> , and adding a call to <code>notify()</code> at the end of the <code>put()</code> method.
<pre>public synchronized void put(int newData) { ... notify(); } public synchronized int get() throws InterruptedException { while (null == head) wait(); ... }</pre>

2. How to learn a language (0 points)

Reflect on your experience in learning how to learn a language. As a reminder, here are the steps you were asked to repeat for each of the languages you learned this semester:

- (i) Find peers and gurus.
- (ii) Get compiler, and figure out how to use it.
- (iii) Read tutorial.
- (iv) Bookmark language and library reference.
- (v) Read example code.
- (vi) Write little programs exercising I/O, types, control flow, and library usage.
- (vii) Learn how to understand common error messages.
- (viii) Write slightly larger programs than in Step (vi).

2a. Which step helped you the most, and why?

Example solutions

Reading the tutorial (iii) helped me the most; without it, I couldn't learn the language at all. Writing little programs (vi) came in as a close second. It got me started, and the code I wrote became a good reference to go back to when writing less trivial code.

2b. Which step took the most time? How long?

Example solutions

Reading the tutorial (iii) took the most time. I don't remember exactly how long, but about 8 hours seems like a realistic time for a typical introductory tutorial.

2c. Think of another helpful finger exercise to add to Step (vi).

Example solutions

For languages that provide object-oriented features (Python, C++, Ada, Java), reimplementing the zoo code example in [hw07.pdf](#) Question 4 is useful. An addition to that exercise would be to provide a constructor in a superclass that gets called by a subclass constructor.

2d. What were the exercises for Step (viii) that you did as part of homeworks for this class?

Example solutions

Python: hw03 Q5 (for example, extend the data formatter example to sort the table by a user-specified column)
Scheme: hw04 Q1 (list-sort)
C: hw05 Q3 (arrays/pointers); hw08 Q5 (call-by-reference)
Ada: (none)
Java: hw08 Q1 (given vtables, write code)
SML: hw10 Q3 (listSortInt); hw10 Q4 (generic listSort)

3. SML (0 points)

3a. Write a function `listMinInt` that finds the minimum element of an integer list. Your function should have the type `int list -> int`. Your function does not need to yield a sensible result for empty lists.

Example solutions

```
fun listMinInt (h::[]) = h
  | listMinInt (h::t ) =
    let val r = listMinInt t
    in if h < r then h else r end;
```

3b. Write a function `removeFirst` that takes a pair of an item and a list, and returns the list with the first occurrence of the item removed, if any. Your function should have the type `'a * 'a list -> 'a list`.

Example solutions

```
fun removeFirst (_, [] ) = []
  | removeFirst (x, h::t) =
    if x = h then t else h::removeFirst(x, t);
```

- 3c. Write a function `listSortInt` that sorts an integer list in non-descending order. You can reuse the functions from Questions 3a and 3b for this step. Your function should have the type `int list -> int list`.

Example solutions

```
fun listSortInt [] = []
  | listSortInt (h::t) =
    let val m = listMinInt (h::t)
    in m::listSortInt(removeFirst(m, h::t)) end;
```

4. SML (0 points)

- 4a. Write a function `listMin` that finds the minimum element of a list, according to a comparison function it receives as a parameter. Your function should be curried: given the comparison function, it should return another function that then takes a list parameter and returns the minimum element. For example, `listMin (op <)` should return a function that works the same as `listMinInt` from Question 3a. The type of your function should be `('a * 'a -> bool) -> 'a list -> 'a`.

Example solutions

```
fun listMin _ (h::[]) = h
  | listMin f (h::t) =
    let val r = listMin f t
    in if f (h, r) then h else r end;
```

- 4b. Write a function `listSort` that sorts a list in the order specified by a comparison function it receives as a parameter. Your function should be curried: given the comparison function, it should return another function that then takes a list parameter and returns the sorted list. For example, `listSort (op <);` should return a function that works the same as `listSortInt` from Question 3c. The type of your function `listSort` should be `('a * 'a -> bool) -> 'a list -> 'a list`.

Example solutions

```
fun listSort _ [] = []
  | listSort f (h::t) =
    let val m = listMin f (h::t)
    in m::(listSort f (removeFirst(m, h::t))) end;
```

<http://www.cs.nyu.edu/courses/summer07/G22.2110-001/hw10-example-solutions.pdf>

Total points: 0.