

## COP4020 Homework 2 (due date: September 30, 2002)

1. “*Masters of the Programming Universe*” [article in IEEE Spectrum, Sept. 2002]. “Last month, three luminaries of the computer science and computer programming community died days of each other: Ole-Johan Dahl, and Kristen Nygard, both of Norway, and Edsger Wybe Dijkstra, of the Netherlands. [...] Ole-Johan Dahl and Kristen Nygard created object-oriented computer programming, the programming model of choice for many modern applications. The Simula languages they developed during the 1960s provide the underlying logic for many languages in use today, including C++ and Java. [...] Edsger W. Dijkstra is best known for establishing that mathematical logic must be the basis for computer program construction and for his contributions to programming methodology. He was responsible for the idea of building operating systems as explicitly synchronized sequential processes, and developed the stack model of computation. [...] His 1968 article in Communications of the ACM, “Go To Statement Considered Harmful” (<http://www.acm.org/classics/oct95/>), is one of the most famous (and succinct!) articles in all of computing literature.”

Read Dijkstra’s famous article and write a one-page summary (500 words minimum) with personal observations and opinions on the use of go-to’s.

2. Functional programming advocates and supports code re-use through higher-order functionals rather than classes and class/function templates of object-oriented languages such as C++ (although both approaches can be used in modern functional languages). The idea is that a set of (higher-order) functions that operate on generic and polymorphic data types, such as lists, is sufficient to build more complex functions and powerful programs.
  - (a) Write four Scheme functions that take a word (Scheme atomic symbol) as an argument and return either `#t` or `#f` depending on the grammatical classification of the word. The four functions should be named `det?`, `noun?`, `verb?`, and `adj?`. You may assume that the vocabulary is limited to the words: `a`, `an`, `the`, `apple`, `car`, `dog`, `road`, `eats`, `occupies`, `rides`, `walks`, `hairy`, `hot`, `red`. Hint: you can use the `member` function to implement each of the four functions. Save your Scheme source in the file named `homework2.scm`. Login with ssh to `linprog2.cs.fsu.edu` and type `'scheme'`. Test your Scheme functions. For example:

```
linprog2> scheme
Scheme saved on Wednesday July 18, 2001 at 11:27:20 PM
Release 7.5.17
Microcode 14.4
Runtime 14.189
1 ]=> (load "homework2")
2 ]=> (det? 'the)
;Value: #t
3 ]=> (noun? 'hairy)
;Value: #f
4 ]=> (verb? 'the)
;Value: #f
5 ]=> (adj? 'hairy)
;Value: #t
```

- (b) This exercise illustrates a simple form of code re-use using the `map` and `reduce` higher-order functions. Explain what the program fragment below does and how the result is computed:

```
(define \/  
  (lambda (a b)  
    (if a #t b)  
  )  
)  
(reduce \/ (map det? '(the hairy dog eats a red apple)))
```

What is the result when the sentence does not contain any determiners, e.g. `dog eats apple`?

Note: The `map` and `or` (logical or) functions are built-in Scheme functions. The `reduce` function can be found in course notes 2.

- (c) Copy the `filter` function from the course notes 2 into your `homework2.scm` file. Write a new function that uses the `filter`, `length`, and `adj?` functions to count the number of adjectives in a sentence. Name your function `adjectives`.

```
1 ]=> (load "homework2")  
2 ]=> (adjectives '(a hairy red dog eats a hot dog))  
;Value: 3
```

- (d) Write a function named `reject` that returns `#t` when more than 25% of the words in a sentence are adjectives and `#f` otherwise. For example:

```
1 ]=> (load "homework2")  
2 ]=> (reject '(a hairy red dog occupies the hot red car))  
;Value: #t  
3 ]=> (reject '(a red car rides the road))  
;Value: #f
```

Submit your comments on Dijkstra's article and the `homework2.scm` file with the Scheme functions to [engelen@cs.fsu.edu](mailto:engelen@cs.fsu.edu).