McMaster University
Department of Computing and Software
Dr. Wolfram Kahl

SE 3E03, Fall 2002
Assignment 4

Design and Selection of Programming Languages

17th September 2002

Problem 15

For Oberon, read section 2 of the language definition handed out on Monday to learn the EBNF used there and refer to section 3.2 for number literals.

For Java, read chapter 2 of the Java Language specification to learn the grammar formalism used there, and then refer to sections 3.10.1 and 3.10.2 for integer and floating-point literals.

a) For each of the both languages, translate the definition of its numerical literals into the formalism used to define literals in the other language.

b) Highlight differences between the two languages in both versions of the definition pairs.

c) Provide examples of numerical literals for all kinds of differences you detect — literals allowed in one language, but not in the other, or literals interpreted in different ways between the two languages.

Problem 16 (More Java)

Continue the Java tutorial at the following web site:

URL: http://java.sun.com/docs/books/tutorial/

Follow the whole trail "Learning the Java Language", repeating what you had seen before, and complete all the exercises.

From the trail "Essential Java Classes", follow the lesson "Handling Errors Using Exceptions", and again complete all the exercises. Then, from the lesson "I/O: Reading and Writing" familiarize yourself with at least the material in the sections "Overview of I/O Streams" and "Using the Streams".

Problem 17 (Lexical grammar of HTML)

A typical HTML page looks like the following:

```
<HTML>
<HEAD>
<TITLE>Sample HTML Page</TITLE>
</HEAD>
<BODY text="#0000D0" bgcolor="#FFFF2C">
<H1 align="center">Sample HTML Page</H1>
<P>
<EM>Voil&agrave;</EM>, this should look <STRONG>great</STRONG> in your browser.
</P>
</BODY>
</HTML>
```

HTML is an SGML application, and therefore inherits SGML nomenclature for its syntax.

An HTML document consists of nested *elements* that are enclosed in a *start tag* and an *end tag*; at the toplevel, it should have precisely one HTML element.

A start tag starts with the "start tag opening" (STAGO) sequence "<" and ends with the "tag closing" (TAGC) sequence ">"; an end tag starts with the "end tag opening" (ETAGO) sequence "</" and also ends with the TAGC. End tags contain only the *generic identifier* (GI) of their element; start tags can contain attribute-value pairs after the GI, where values are usually "character data" (CDATA).

Elements can contain other elements, and also "parsed character data" (PCDATA) that includes *entity references* (short: *entities*) delimited by "entity reference open" (ERO) "&" and "reference close (REFC) ";" , like that used for the accented character above; PCDATA never includes the STAGO or ETAGO or ERO.

(There are entities lt and amp that may be referenced by entity references "<" and "&" for denoting the characters used as STAGO and ERO in HTML.)

a) Using only the information given above, extrapolate from the example and give regular expressions for CDATA, PCDATA, entities, start tags and end tags.

   Even if what you define is not really any existing version of HTML, take care to make sure that it is unambiguous! For example, do your definitions unambiguously determine where an attribute value ends?

b) Find HTML tutorials and definitions on the web (starting from the links on the course page) and determine the differences between your solutions to a) and the standards.

c) Extend your solution to a) into a context-free grammar (may be given in EBNF) that generates at least the example document.

d) Implement a Java class STAG for representing all information contained in start tags. There should be a useful interface for generating and printing start tags.

Problem 18 (optional)

If you are interested in Applet programming, finish problem 14.

You may want to refer to the corresponding parts of the Java Tutorial first, and you should be aware of your choices:

- The "old" GUI library awt should work under most browsers, but is in fact being phased out.
- The new Swing GUI library may not work with all browsers; in the labs, netscape6 should work, and it is safest to test with the appletviewer from the JDK 1.4 installation in /usr/local/j2sdk1.4.0.

  You can check whether you setup already refers to that installation by issuing the following commands:

  ```
  which java
  which appletviewer
  ```

  If you include the line

  ```
  source /u2/se3e03/admin/etc/.cshrc
  ```

  in your $HOME/.cshrc (or issue the command interactively), then the JDK 1.4 installation will be used if you call java* or appletviewer commands.