IN2009 Language Porcessor:s Sample Questions

1)

a)

Consider the following grammar for strings of balanced parenthesis:

S → S S
S → (S)
S → a

Explain what it means for a context-free grammar to be ambiguous. Using your explanation, show that the balanced parenthesis grammar is ambiguous using the shortest string that will illustrate the ambiguity.

b) The reference manual for a MiniJava-like programming language contains the following grammar for a repeat-until statement:

Statement → ***repeat*** *Statement* ***until*** *(Exp)*

i) Sketch a possible abstract syntax for the *repeat-until* statement.
ii) Show how semantic actions in a grammar for a parser-generator such as JavaCC can be used to produce abstract syntax trees for the *repeat-until* statement.
iii) Informally describe an appropriate typecheck for the *repeat-until* statement.
iv) Suppose a compiler for a MiniJava-like language that includes the *repeat-until* statement translates all statements and expressions into intermediate code (e.g. intermediate representation (IR) trees). Outline the intermediate code that might be generated in translation of the *repeat-until* statement.
You may wish to use a simple example to explain your translation, e.g.:

```
repeat
  { sum = sum + x; prod prod * x; }
until (x < 20)
```

You can assume that the expression tree for any variable `v` is simply `TEMP v`. Do not show translations for the body of the example *repeat-until* statement (in braces in this example `{...}`).

2)

a) The following regular expression recognises certain strings consisting of the letters a, b and c:

```
(a|c)((bc)|c)* c*
```

i) For the following 5 strings, indicate whether or not they are recognised by the above regular expression:

```
cbc, abbc,c abcbcbccc,ccbbcbcc
```

ii) Show three more strings that are recognised by the above expression.

iii) Show two more strings consisting of the letters a, b and c that are ***not*** recognised by the above regular expression.

b)

i) Explain why left-recursion must be eliminated from grammar productions which are to be used in construction of a recursive-descent parser.

ii) Write down a general rule for rewriting left-recursive grammar productions to equivalent right-recursive grammar productions.

iii) Use the general rule from part ii) to rewrite the following productions to be right-recursive:

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (\mathrm{E})$

$F \rightarrow \texttt{integer}$

3)

a) Some programming language implementations avoid in some circumstances the need to pass parameters via a stack frame. Outline what these circumstances might be and why passing via the stack frame might be avoided. Also, outline situations where the use of a stack frame to pass parameters cannot usually be avoided.