CZ3007 Compiler Techniques

Tutorial 3 (Semantic Analysis – Part 2)

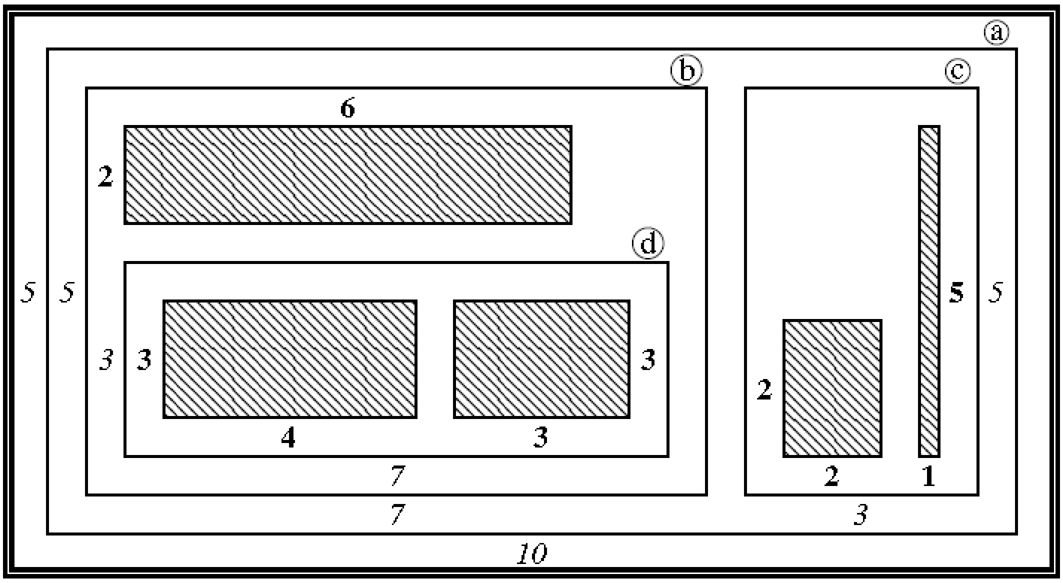
(to be completed in 2 hours)

1. With reference to document `grammar.ast` provided in Lab 3, write a Beaver specification including the semantic actions to build the abstract syntax tree nodes for the following grammar rules.

```
Declarations → Declarations Declaration
             |   λ

Declaration →  Accessibility TypeName   ID   LPAREN OptParameters   RPAREN
LCURLY Stmts  RCURLY
             | Accessibility TypeName  ID  SEMICOLON
             | Accessibility TYPE  ID EQL  STRING_LITERAL  SEMICOLON

Accessibility → PUBLIC
              |  λ

OptParameters →  Parameters
              |  λ

Paremeters → Parameters  COMMA  Parameter
           | Parameter
```

2. Consider the abstract grammar for describing the box scenarios in the lecture:

```
Frame ::= Box;
abstract Box;
HBox : Box ::= Left:Box  Right:Box;
VBox : Box ::= Top:Box  Bottom:Box;
ABox : Box ::= <Width:Integer>  <Height:Integer>;
```



An example box scenario is shown in the diagram above. Atomic boxes are shaded, bold numbers indicate width and height of atomic boxes, italic numbers width and height of containing boxes(dimensions are not depicted accurately).

(i) Define in attribute grammar an attribute, perimeter(), for an AST node Box to compute the perimeter value of a box.

(ii) Define in attribute grammar an attribute, EnclosingHBox(),for an AST node Box to return the reference to the innermost enclosing horizontal box (HBox) for a box. The innermost enclosing HBox for a child box in a HBox is the parent box. The innermost enclosing HBox for a child box in a VBox is the innermost enclosing HBox of the parent. For example, the innermost enclosing HBox for the two boxes in box d is box d. The innermost enclosing HBox for box d is box a.

3. Refer to document `grammar.ast` and document `names.jrag` provided in Lab 3, discuss the process of looking for the declaration (VarDecl) for an AST node VarName.

4. Refer to document `grammar.ast`, document `names.jrag`, document `types.jrag` and document `typecheck.jrag` provided in Lab 3.
   a. discuss the process of type check for the statement “return x + a” in the following program.
   b. discuss the process of type check for “f(10) > 10” in the following program.

```
module M {
     int a;
     int f(int x) {
          return x + a;
     }
     boolean g() {
          return f(10) > 10;
     }
}
```

Question not to be covered in Tutorial:

A simple language is defined by the context free grammar in Figure 1. Suppose the grammar specification file starts with the declarations in Figure 2.

(i) Give the abstract grammar to define the AST nodes for the language.

(ii) Modify the context free grammar specifications to add semantic actions for building the AST.

```
Expr  → Expr  PLUS Term
      | Expr  MINUS Term
      | Term
Term → Term MUL Factor
     | Term DIV Factor
     | Factor
Factor → LPAREN  Expr  RPAREN
       | INT_LITERAL
```

Figure 1

```
%terminals INT_LITERAL, PLUS, MINUS, MUL, DIV, LPAREN,
RPAREN;
%goal Expr;
%typeof INT_LITERAL = "String";
%typeof  Expr, Term, Factor = "Expr";
```

Figure 2