University of Maryland University College

CMSC 330 - Advanced Programming Languages Programming Project 1

The first programming project involves completing a program that parse, using recursive descent, a GUI definition language defined in an input file and generates the GUI that it defines. The grammar for this language is defined below:

```
gui ::=
    Window STRING '(' NUMBER ',' NUMBER ')' layout widgets End '.'

layout ::=
    Layout layout_type ':'

layout_type ::=
    Flow |
    Grid '(' NUMBER ',' NUMBER [',' NUMBER ',' NUMBER] ')'

widgets ::=
    widget widgets |
    widget

widget ::=
    Button STRING ';' |
    Group radio_buttons End ';' |
    Label STRING ';' |
    Panel layout widgets End ';' |
    Textfield NUMBER ';'

radio_buttons ::=
    radio_button radio_buttons |
    radio_button

radio_button ::=
    Radio STRING ';'
```

In the above grammar, the red symbols are nonterminals, the blue symbols are tokens and the black punctuation symbols are BNF metasymbols. Among the tokens those in title case are keywords. The character literals are punctuation tokens.

Below is an explanation of the meaning of some of the symbols in the above productions that should help you understand the actions that are to be performed when each of the productions is parsed:

- In the `window` production the string is name that is to appear in the top border of the window and the two numbers are the width and height of the window

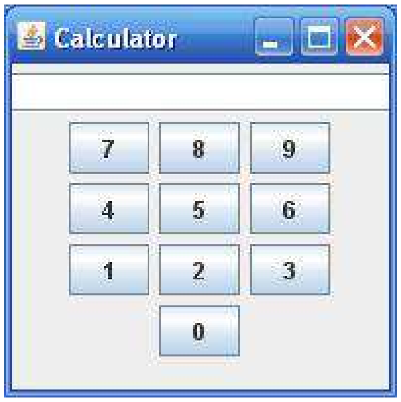
- In the production for `layout_type` that define the grid layout, the first two numbers represent the number of rows and columns, and the optional next two the horizontal and vertical gaps
- In the production for `widget` that defines a button, the string is the name of the button
- In the production for `widget` that defines a label, the string is text that is to be placed in the label
- In the production for `widget` that defines a text field, the number is the width of the text field
- In the production for `radio_button`, the string is the label of the button

You parser should properly handle the fact that panels can be nested in other panels. Recursive productions must be implemented using recursion. Syntactically incorrect input files should detect and report the first error.

Below is an example of an input file:

```
Window "Calculator" (200, 200) Layout Flow:
  TextField 20;
  Panel Layout Grid(4, 3, 5, 5):
    Button "7";
    Button "8";
    Button "9";
    Button "4";
    Button "5";
    Button "6";
    Button "1";
    Button "2";
    Button "3";
    Label "";
    Button "0";
  End;
End.
```

The above input file should produce the GUI shown below:


Calculator
7
8
9
4
5
6
1
2
3
0