CS 241 — Spring 2020 — Assignment 7

Marmoset Notes

For each question, Marmoset has public and release tests, and often has secret tests as well. The weights assigned to each question are given on the assignment.

When you release test a submission, Marmoset will only report on public and release tests. No information about secret tests will be visible until after the assignment deadline.

This means the number of marks you see on Marmoset (prior to the assignment deadline) will not necessarily match the number of marks shown on the assignment page.

Problem 1 — 7 marks of 70 (`a7p1.cfg-r`)

Write a .cfg-r file which solves A6P6.

You may find the `cs241.cfgrl` tool helpful for checking your solutions, which takes a .cfg-r file and produces a left traversal of the same derivation.

Click here to return to the top of the page.

Problem 2 — 7 marks of 70 (`a7p2.cfg-r`)

Write a .cfg-r file that solves A6P7.

Click here to return to the top of the page.

Problem 3 — 30 marks of 70 (`lr.cc` or `lr.rkt`)

Write a Racket or C++ program that reads an LR1 File representing a context-free grammar, an LR(1) machine, and a sequence to be recognized. If the sequence is in the language, output unindented reversed rightmost derivation format above. If the input is not recognized, output "ERROR at *k*" (followed by a single newline character) to standard error, where *k* is one greater than the number of tokens in the longest correct prefix.

For the sample LR1 file as input, the correct output is:

```
term id
expr term
term id
expr term
term ( expr )
```

```
expr expr - term
term id
expr expr - term
S BOF expr EOF
```

If you replace the last line of this file by:

```
BOF id - id ) - id EOF
```

the correct output (to standard error) is:

```
ERROR at 5
```

You may also test your program with the WLP4 grammar and parse table, in .lr1 format after adding a sequence to be recognized.

You can use `cs241.slr` to generate the LR(1) DFA for any SLR(1) grammar. You can use it to generate parsers for your own grammars, or to generate additional test inputs to this problem. `cs241.slr` expects a CFG file file on standard input, and outputs an LR1 file on standard output.

Click here to return to the top of the page.

Problem 4 — 13 marks of 70 (`wlp4parse.cc` or `wlp4parse.rkt`)

Modify your solution to Problem 3 to always use the WLP4 grammar and parse table. It should accept as input the output from Assignment 6 Problem 1 and have the same output as Problem 3, except that each terminal is printed with its lexeme at the appropriate point in the derivation. For example, if the input to the program is **(note the lack of BOF and EOF)**:

```
INT int
WAIN wain
LPAREN (
INT int
ID foo
COMMA ,
INT int
ID bar
RPAREN )
LBRACE {
RETURN return
NUM 42
SEMI ;
RBRACE }
```

The output should be:

```
BOF BOF
INT int
WAIN wain
LPAREN (
INT int
type INT
ID foo
dcl type ID
COMMA ,
INT int
type INT
ID bar
dcl type ID
RPAREN )
LBRACE {
dcls
statements
RETURN return
NUM 42
factor NUM
term factor
expr term
SEMI ;
RBRACE }
main INT WAIN LPAREN dcl COMMA dcl RPAREN LBRACE dcls statements RETURN expr SEMI RBRACE
procedures main
EOF EOF
start BOF procedures EOF
```

Errors should be handled in the same way as Problem 3.

Click here to return to the top of the page.

Problem 5 — 13 marks of 70 (`wlp4parse.cc` or `wlp4parse.rkt`)

Modify your output to Problem 4 to follow the WLP4I format. The format of a `.wlp4i` file represents the derivation as well as the tokens in the program. Errors should be handled in the same way as Problem 4. You should make a parse tree and print it using a preorder traversal to achieve this.

You can use `wlp4scan` to help generate inputs to your program, and `wlp4parse` and `wlp4icheck` to help check your output.

For example, when using the input from the previous example, the corresponding output is:

```
start BOF procedures EOF
BOF BOF
procedures main
main INT WAIN LPAREN dcl COMMA dcl RPAREN LBRACE dcls statements RETURN expr SEMI RBRACE
INT int
WAIN wain
LPAREN (
dcl type ID
type INT
INT int
ID foo
COMMA ,
dcl type ID
type INT
INT int
ID bar
RPAREN )
LBRACE {
dcls
statements
RETURN return
expr term
term factor
factor NUM
NUM 42
SEMI ;
RBRACE }
EOF EOF
```

Click here to return to the top of the page.