# PROGRAMMING LANGUAGES

## Assignment 3: Parser for Spreadsheet language

Read the description file common to assignments 2-4

In Assignment 3: you need to specify the structure of formulas as a yacc grammar, and call the appropriate operations that will be provided to you as a module.  Your implementation will be modular, in that you should be able to define your parser and create the interpreter framework irrespective of the actual implementation of the backend functions.  You may assume the types of the backend functions.

Informal Syntax

Informally the grammar of our spreadsheet formula language is:

- An instruction has a head and a body. The head consists of a cell indexing the output destination of the formula, and the body consists of an expression  being evaluated with respect to the spreadsheet. The head and the body are separated by an assignment sign (**:=**)
- A formula ends with a semicolon (**;**)
- The arguments of a function are either indices, ranges, constants or expressions using mathematical operations
- An index is a pair of integers.  General format for an index $I$ of a cell is [$i, j$]   Examples are [0,0], [3,4], …
- A range consists of two indices standing for the top left index and the right bottom index. General format for an range $R$ of cells is **(** $I$ **:** $I$ **)** with an example being **( [** 5, 6**] : [**100, 6**] )**

The formulas provided to you are one on each line. The general format of a formula is given below:

Type 1 formula (unary operations on a range of cells)

$I$ **:=** FUNC  $R$  **;**

Type 2 formula (binary operations on ranges)

$I$ **:=** FUNC $R$ $R$ **;**

$I$ **:=** FUNC  $C$ $R$ **;**

$I$ **:=** FUNC $R$ $C$ **;**

$I$ **:=** FUNC  $I$ $R$ **;**

$I$ **:=** FUNC $R$ $I$ **;**

**Backend**

You may assume there exists a module in OCaml with the following functions.

- full_count: sheet -> range -> index -> sheet : Fills count of valid entries in the given range into the specified cell
- row_count: sheet -> range -> index -> sheet : Fills count of valid entries per row in the given range into the column starting from the specified cell
- col_count: sheet -> range -> index -> sheet : Fills count of valid entries per column in the given range into the row starting from the specified cell.


- full_sum: sheet -> range -> index -> sheet : Fills the sum of entries of cells in the given range into the specified cell
- row_sum: sheet -> range -> index -> sheet : Fills the sum of entries of cells per row in the given range into the column starting from the specified cell
- col_sum: sheet -> range -> index -> sheet : Fills the sum of entries of cells per column in the given range into the row starting from the specified cell


- full_avg: sheet -> range -> index -> sheet : Fills the average of entries of cells in the given range into the specified cell
- row_avg: sheet -> range -> index -> sheet : Fills the average of entries of cells per row in the given range into the column starting from the specified cell
- col_avg: sheet -> range -> index -> sheet :  Fills the sum of entries of cells per column in the given range into the row starting from the specified cell


- full_min: sheet -> range -> index -> sheet : Fills the min of entries of cells in the given range into the specified cell
- row_min: sheet -> range -> index -> sheet : Fills the min of entries of cells per row in the given range into the column starting from the specified cell
- col_min: sheet -> range -> index -> sheet : Fills the min of entries of cells per column in the given range into the row starting from the specified cell


- full_max: sheet -> range -> index -> sheet : Fills the max of entries of cells in the given range into the specified cell
- row_max: sheet -> range -> index -> sheet : Fills the max of entries of cells per row in the given range into the column starting from the specified cell
- col_max: sheet -> range -> index -> sheet : Fills the max of entries of cells per column in the given range into the row starting from the specified cell


- add_const: sheet -> range -> float -> index -> sheet : adds a constant to the contents of each cell in the selected cell range
- subt_const: sheet -> range -> float -> index -> sheet : subtracts a constant from the contents of each cell in the selected cell range
- mult_const: sheet -> range -> float -> index -> sheet : multiplies the contents of each cell in the selected cell range by a constant.
- div_const: sheet -> range -> float -> index -> sheet : divides the contents of each cell in the selected cell range by a constant.


- add_range: sheet -> range -> range -> index -> sheet : adds the cell contents for each corresponding pair of cells in two selected cell ranges

- subt_range: sheet -> range -> range -> index -> sheet : performs a subtraction on the cell contents for each corresponding pair if cells in two selected cell ranges
- mult_range: sheet -> range -> range -> index -> sheet : multiplies the cell contents for each corresponding pair of cells in two selected cell ranges
- div_range: sheet -> range -> range -> index -> sheet : performs a division on the cell contents for each corresponding pair  of cells in two selected cell ranges

# Submission status

| Submission status | Submitted for grading |
| --- | --- |
| Grading status | Not graded |
| Due date | Sunday, 8 March 2020, 11:59 PM |
| Time remaining | Assignment was submitted 42 mins 14 secs early |
| Last modified | Sunday, 8 March 2020, 11:16 PM |
| File submissions | Assignment3_2018CS50098.zip |
| Submission comments | ▶ Comments (0) |

# Feedback

| Grade | 10.00 / 100.00 |
| --- | --- |
| Graded on | Sunday, 23 August 2020, 11:07 AM |

**NAVIGATION**

Dashboard

🏠 Site home

    Site pages