Интерпретатор для клеточного робота

Вариант 75 (***)

Разработать систему для управления клеточным роботом, осуществляющим передвижение по клеточному лабиринту. Клетка лабиринта имеет форму квадрата.

Робот может передвинуться в соседнюю клетку, если она не отмечена в файле описания лабиринта как заполненная.

1. Необходимо разработать формальный язык для описания действий клеточного робота с поддержкой следующих литералов, операторов и предложений:
   - Знаковых целочисленных литералов;
   - Логических констант **T** и **F**;
   - Меток формата **<~<номер метки>>**
   - Переменных в форматах:
     - Целочисленные: **<,<числовой номер переменной>>**, например «**,15**», «**,45**»;
     - Логические: <**.<числовой номер переменной>**>, например «**.15**», «**.45**»;
     - Переменных процедур <**$<числовой номер переменной>**>, например «**$15**», «**$45**»; процедуры, как и все переменные, находятся в глобальной области видимости, обмен данными с процедурой осуществляется через переменные, непосредственно значение процедура не возвращает и не может использоваться в выражения, кроме оператора присваивания; переход по метке из процедуры во вне ее, а также переход пометке внутрь процедуру недопустим.
     - Целочисленных массивов **<.<числовой номер переменной>: <индекс в размерности 1>**[**-<индекс в размерности 2>, …**]**>>**, индекс по умолчанию 0; в качестве индексов могут использоваться целочисленные литералы, целочисленные переменные и элементы массивов (допускается любой уровень вложенности и рекурсии); память под элементы массива выделяется при первом обращении к соответствующему элементу.
     - Логических массивов **<,<числовой номер переменной>: <индекс в размерности 1>**[**-<индекс в размерности 2>, …**]**>>**, индекс по умолчанию 0; в качестве индексов могут использоваться целочисленные литералы, целочисленные переменные и элементы массивов (допускается любой уровень вложенности и рекурсии); память под элементы массива выделяется при первом обращении к соответствующему элементу.
     - Массивов процедур**<$<числовой номер переменной>: <индекс в размерности 1>**[**-<индекс в размерности 2>, …**]**>>**, индекс по умолчанию 0; в качестве индексов могут использоваться целочисленные литералы, целочисленные переменные и элементы массивов (допускается любой уровень вложенности и рекурсии); память под элементы массива выделяется при первом обращении к соответствующему элементу.

   (номера целочисленной и логической переменной, массивов и процедур в пределах одной программы совпадать не могут)
- Арифметических операторов:
  - Инкремент значения целочисленной переменной **,#<номер переменной>**;
  - Декремент значения целочисленной переменной **,*<номер переменной>**;
- Операторов присваивания '**<-**';
- Оператор присваивания для процедур
  - **<имя процедуры><-<имя процедуры или элемент массива процедур>**;
  - **<имя процедуры><-{<предложения языка>}**;
  - при присвоении процедур ни присваиваемый код, ни сама процедура не выполняется.
- Логических операторов:
  - **.# <логическое выражение>** - стрелка Пирса;

  (операторные скобки могут использоваться).
- Операторов сравнения:
  - **<арифметическое выражение> eq <арифметическая константа>**;
  - **<логическое выражение> eq <логическая константа>**;
  - **<переменная процедура> eq np**
- Операторов цикла **(<логическое выражение>) <оператор>** с объединением операторов с помощью фигурных скобок **{ }**;
- Операторов условного перехода **[[логическое выражение]] [please] <метка>**;
- Оператор отсутствия каких-либо действий **np**;
  - Применяется как значения по умолчанию для неинициализированных процедур, также может встречаться в пользовательском коде.
- Оператор связывания идентификаторов с процедурами
  - **<идентификатор> @ <переменная процедура>**
  - при обращении к связанному идентификатору сначала вызывается связанная с ним процедура, затем возвращается значение идентификатора (возможно модифицированное связанной процедурой);
  - в качестве идентификатора может быть любая переменная, в том числе процедура; рекурсивное связывание не допускается, при попытке связывания, которое образует рекурсию, оператор не выполняется, и возвращает **F**, иначе возвращает **T**;
  - с идентификатором может быть связано несколько процедур;
  - процедура может быть связана с несколькими идентификаторами;
- Оператор разрыва связи между идентификатором и процедурой
  - **<идентификатор> % <переменная процедура>**
  - если оператор вызывается не для связанных ранее переменных, то ничего не происходит; в любом случае возвращается значение **T**
- операторов перемещения робота на одну клетку в заданном направлении относительно текущего **mf, mb, mr, ml**. После выполнения оператора робот разворачивается и перемещается в указанном направлении. Если оператор невозможно выполнить из-за наличия препятствия, он возвращает логическое значение **F**. Иначе – **T**.
- оператор телепортации робота в случайную, еще не посещенную им клетку лабиринта **tp**; робот имеет ограниченное количество попыток телепортации, определяемой средой выполнения; при успешной телепортации возвращается T, иначе F; после телепортации робот не знает свое местоположение.

Предложение языка завершается символом перевода строки.

2. Разработать с помощью flex и bison интерпретатор разработанного языка. При работе интерпретатора следует обеспечить контроль корректности применения языковых конструкций (например, присваивание значения нелеводопустимому выражению); грамматика языка должна быть по возможности однозначной.

3. На разработанном формальном языке написать программу для полного обхода роботом всей доступной ему части лабиринта. Описание лабиринта и начальное положение робота задается в текстовом файле. В файле задаются координаты робота и координаты клеток с препятствиями.