Formal Languages and Compilers

Laboratory n° 3

1 Exercise (mini C)

Using JFLEX and CUP, write a program which recognizes the syntax of a subset of the C language (*mini C*). Given an input file this program must indicate if the file is a correct mini C source.

In particular, the language characteristics are the following:

- `main` and functions do not exist: thus, the whole program will be written in a single input file which represents the `main`.
- Variables of type `int` and `double` and one-dimensional vectors of those type can be declared. The variables cannot be initialized in the declaration phase (e.g. an instruction like `int a=0;` is not supported).
- The vectors indexes can be variables or integer numbers but complex expressions (e.g. correct assignment instruction: `a[2]=3*b[c]-a[3];`; invalid assignment instruction: `a[2+4]=0;` or `a[c+1]=2;`).
- Assignment instruction can be executed (exactly like in C). The language allows the use of a particular print instruction `print(<variable>);` that allows to print the value represented by the variable with name `<variable>` (e.g. `print(a[2]);` print the vector `a` value of index 2).
- The `while` and `if` have exactly the same syntax of the C language. Handle both the syntax where an instructions list is enclosed within curly brackets and the case where the `if` branches contain only one instruction (i.e. curly brackets are not mandatory).
- The *boolean* expressions inside the `while` and `if` conditions must allow the use of the comparison operators "==", "<", "<=", ">", ">=" and the boolean operators "&" (AND), "|" (OR) and "!" (NOT). Handle correctly the **precedence** of the operators listed above (e.g. `if (3+2-a[4] < 3-3*a[c]+1 & b==3 | a[2]<=3*b+1)`).

1.1 Input file example

An input file example might be the following:

```
/* Esempio algoritmo di ordinamento Bubble sort */

double x[5];
int i, j;
double swap;
int pos;

/* Inizializzazione vettore */
x[0] = -2.0;
x[1] = -3.0;
x[2] = 3.0;
x[3] = 5.0;
x[4] = 2.5;

/* Bubble sort */
pos = 5;
while(pos > 0){
  i = 0;
  while (i < pos - 1){
    j = i + 1;
    if (x[i] > x[j]){
      swap = x[j];
      x[j] = x[i];
      x[i] = swap;
    }
    i = i + 1;
  }
  pos = pos-1;

}

/* Stampa risultati */
i = 0;
while(i<5){
  print (x[i]);
  i = i + 1;
}
```

2 Exercise (Grammar derived from an exam)

Using Jflex and Cup, write a scanner and a parser which recognize the language for the management of a library.

The input file is subdivided into two sections separated by the symbol "%%" (two percent symbols).

The first section is composed by a non-empty list of writers and the books written by them. Each list element has the following fields:

`<writer name> > <books list>;`

Where `<writer name>` is a string of letters enclosed by the characters " (Double quote). `<books list>` is a non-empty list of books written by a writer and separated by a "," (comma).

Each list element is composed as follow:

`<ISBN code>:<book title>:<number of pages>:<collocation>`

`<ISBN code>` consists of two numeric characters, followed by a dash, followed by two numeric characters, followed by a dash, followed by 5 hexadecimal characters, followed by a dash and followed by a letter or a numeric character. `<collocation>` (is optional) and is composed by the word `LI` or `LS` (*letteratura italiana* or *letteratura straniera*) followed by the genre `AV`, `BO` o `SO` (*Avventuroso*, *biografico* or *sociale*), followed by an integer number and eventually followed by a letter. The genre `LI BO` does not exist: handle this case.

The second section is composed of a non-empty list of users. Each list element is defined as follows:

`<user name>:<loans list>;`

`<loans list>` is a set of loans associated to a library user, separated by the character "," (comma).

For each loan the loan date and the book ISBN code are reported. The date is in the format "DD/MM/YYYY", where DD is a number between 01 and 31, MM is a number between 01 and 12.

The program must recognize the previously described language and write if it is grammatically correct.

2.1 Input file example

```
"Hesse Herman" -> 88-17-83457-X:"Narciso e Boccadoro":200:LS SO 127 A,
                  88-14-24B43-2:"Siddhartha" : 236 : LS SO 127 B,
                  88-12-34AA3-B:"Lupo della steppa, Il":262:LS SO 127 C;
"Baricco Alessandro"-> 88-17-10625-9:"Seta":100:LI AV 1,
                       88-17-86563-X:"City":319:LI AV 2 A;
"F. Christiane"-> 88-17-11520-7:"Noi, i ragazzi dello zoo di Berlino":346:LS BO 1;
%%
"Giovanni": 02/10/2006 88-17-11520-7;
"Stefano" : 12/04/2007 88-17-83457-X,
            20/09/2007 88-14-24B43-2,
            29/09/2007 88-17-11520-7;
"Giovanni": 02/10/2007 88-17-10625-9,
            02/10/2007 88-17-86563-X;
```

3 Exercise (Facultative)

As an extension of the Exercise 1, write a grammar which recognizes the following C language subset:

3.1 C subset to recognize:

- Declaration of variables of all predefined types (with additional modifiers `signed` and `unsigned`), arrays and pointers.
- The definition of functions with an arbitrary number of arguments (from 0 to n) and a returned value chosen among predefined types.
- Use of arithmetic or boolean expressions that can contain variables and functions of one of the format specified above.
- Use of conditional constructs `if-else`, `switch`, `while`, `do-while` and `for`.

3.2 C subset not to recognize

- Declaration of types using `typedef`, declaration and use of structures (`struct`) and unions (`union`), use of `enum`.
- Variables that represent pointers to function.
- Cast

3.3 Input file example

```
extern int *fn1(int a, int b, char *c[]);
register int ff;

int fn2() {
  static unsigned long int k = 1, i;
  for(i = 0; i < 10; i++) {
    k-1;
  }
}

int main() {
  char *miovett[] = {"Inverno", "Estate"};
  while(fn1(2,3, miovett) != 0) {
    ff++;
  }
  return ff;
}
```