C to Assembler

Aarón Arias Pérez
José Joaquín Arias Gómez-Calcerrada

What do we need to do?

- Ascendent grammar
- Generate an intermediate AST
- Generate assembler code from that AST

Problems

- Ambiguity
- Labels
- Offset
- Tables

```c
int fact(int n) {
    if(n<=1) {
        n=1;
    }
    else {
        return n*fact(n);
    }
    return fact(n);
}

int numero;

int main() {
    scanf("%d", &numero);
    printf("El factorial = %d\n", fact(numero));
    return 0;
}
```

Tables	Variable name
Global variables	variables
Local variables	variables_locales
Argument variables	variables_args

```
%{

#include <iostream>
#include <string>
#include <vector>
#include <map>
#include <fstream>
#include <stack>
#include "TablaVariables.h"
#include "node.h"

using namespace std;

int yyerror(const char* msj);
int yylex(void);

TablaVariables variables;
TablaVariables variables_locales;
TablaVariables variables_args;

vector<string> call_pushs;

stack<int> etiquetas_while;
stack<int> etiquetas_if;
stack<int> etiquetas_and;
stack<int> etiquetas_or;
stack<int> etiquetas_not;
stack<int> etiquetas_eq;
stack<int> etiquetas_neq;
stack<int> etiquetas_gte;
stack<int> etiquetas_gt;
```

Return node

- If $2 is a variable, we check the tables
- Checking order:
  - Local variables
  - Argument variables
  - Global variables

Denoting the offset

Labels

- Usage of a stack for each situation
  - "While" stack
  - "If" stack
  - "And" stack
  - Etc..
- Zero nesting problems

Demo time