Proyecto Final

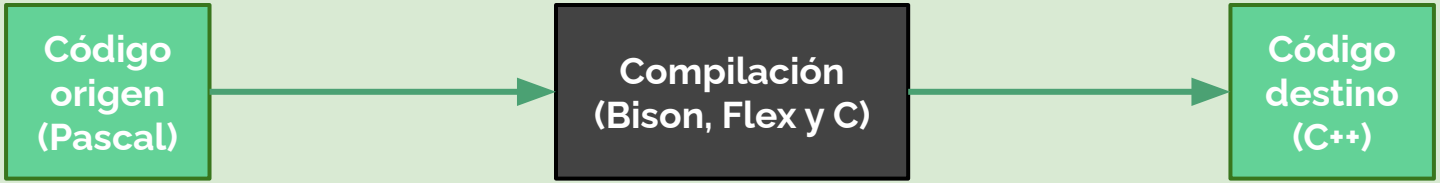
Teoría de Lenguajes y Programación

Miguel Ángel Flores Richter
Pilar Pineda Crevantes
Santiago Valera Barreiros
Bernardo González Herrero
Ian Nikolay Doshner Galland

Objetivo

Construir un compilador, utilizando Bison y Flex, que traduzca del lenguaje de programación Pascal al lenguaje de programación C++ .

Proceso de compilación



Implementación de Flex y Bison

```
struct identificador
{
    char *acumulador;
    char nombre[127];
    char tipo[10];
    int fila_de_declaracion;
    int filas_de_uso[NUM_ELEMENTOS];
    int ubicacion;
    char ambito[127];
    identificador *next;
};
```

```
%union {
    identificador temp;
}
```

```
| WRITE '(' CONSTANTE_CADENA ')' {
    strcpy($$.acumulador, "cout << ");
    strcat($$.acumulador, $3.acumulador);
}
```

```
const_cadena \"[^\n]*\"
digito [0-9]
letra [A-Za-z]
identif {letra}({letra}|{digito})*
addop [\+]|[-]
mulop [\*]|[\/]
simbolo [\.\(\);=\[:,\]\"]
entero [+-]?[0-9]+
real [+-]?{entero}+\.{digito}+([eE]{addop}?{digito}+)?
```

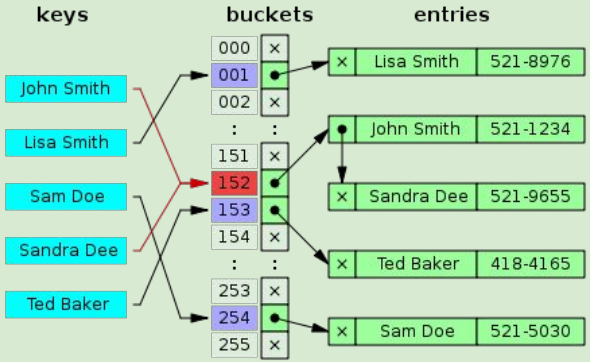
```
{const_cadena}      {
                        // printf("Se encontro const cadena %s\n",yyt
                        yylval.temp.acumulador = strdup(yytext);
                        columna = columna + strlen(yytext);
                        return CONSTANTE_CADENA;
                    }
```

```
if                  {
                        // printf("Se encontro %s\n",yytext);
                        columna = columna + strlen(yytext);
                        return IF;
                    }
then                {
                        // printf("Se encontro %s\n",yytext);
                        columna = columna + strlen(yytext);
                        return THEN;
                    }
```

```
{identif}           {
                        // printf("Se encontro %s\n", yytext);
                        yylval.temp.fila_de_declaracion = fila;//para la ta
                        yylval.temp.acumulador = strdup(yytext);
                        columna = columna + strlen(yytext);
                        return IDENTIFICADOR;
                    }
{const_cadena}      {
                        // printf("Se encontro const cadena %s\n",yytext);
                        yylval.temp.acumulador = strdup(yytext);
                        columna = columna + strlen(yytext);
                        return CONSTANTE_CADENA;
                    }
```

Tabla de símbolos

- Checar dos características principales
  - Verificación de una semántica correcta
  - Apoyar en la correcta generación de código
- Implementada utilizando un mapa hash
  - Solucionar colisiones con método chaining
- Apoyada por una pila para el seguimiento del ámbito



IDENTIFICADOR	TIPO	FILA DE DECLARACION	AMBITO	FILAS DE USO
num1	int	5	max	8, 9,
num2	int	5	max	8, 11,
a	int	3	maxNum	15, 17,
b	int	3	maxNum	16, 17,
max	int	5	max	9, 11,
ret	int	3	maxNum	17, 20,

Resultados

Código en Pascal

```pascal
program main();
    var a,b,c,d: integer;

    function Add(a: integer; b: integer) : integer;
    begin
        Add := a + b
    end;

    function Mult(c: integer; a: integer) : integer;
    begin
        Mult := c * a
    end;

begin
    a := 9;
    b := 7;
    c := Add(a, b);
    d := Mult(c, a);
    while (a > b) do
      b := b + 1;
    writeln(c);
    writeln(d)
end.
```

Código en C++

```cpp
#include <stdio.h>
#include <string>
#include <iostream>

using namespace std;

int a,b,c,d;

int Add( int a, int b) {
return a+b;
}

int Mult( int c, int a) {
return c*a;
}

int main() {
a = 9;
b = 7;
c = Add(a, b);
d = Mult(c, a);
while((a>b)) {
b = b+1;
}
cout << c << endl;
cout << d << endl;
}
```

Manejo de Errores

- %option yylineno
- exit(1)
-

Conclusiones

Haciendo uso le lex y yacc se pudo hacer un reconocedor sintáctico capaz de no solo reconocer una gramática sino también de traducirla de un lenguaje a otro, de Pascal a C + +.

- Reconoce
- Traduce
- Facilita