C2JS编译器说明文档

一． 实验环境

1. 编程语言: Python

2. 解释器版本:

- Python: 3.7.1  
3. 依赖库:

Python: ply

**Javascript:**  readline-sync

二． 使用方式

详见src文件夹下README.md。

三． 实现功能

1. C语言转Javascript

2. 实例代码：

* 回文检测：palindrome.c
* KMP 字符串匹配: kmp.c
* 四则运算计算: calculatior.c
* 额外实例：
  + 聚合对象 array示例：array.c。通过printArray，reverseArray,genPrime(生成不大于正整数n的所有素数)等函数。展示了array对象的初始化，按下标取值，赋值，获取长度，作为函数参数和返回值等

四． 思路与难点

1. 词法分析：lex.py

* 功能：提取字符流中的token,提供lexer()接口便于语法解析
* 实现方式：ply.lex模块的保留字定义，符号定义和正则表示等
* 难点：正则表示提取token
* 效果：
  + 变量类型：int, float, double, char, void(函数返回值), 字符数组，字符串
  + 保留字：for, while, do, if, else, else if, switch, case, default
  + 控制语句：return, break, continue
  + 字面量：true, false
  + 注释：行注释，块注释
  + 运算符：四则运算(含自运算)，条件运算，逻辑运算，位运算
  + 其他： 标点，变量名，include语句等

2. 语法分析：yacc.py

- 功能：根据语法规则生成语法分析树ast

- 实现方式：使用ply.yacc模块，定义通用抽象语法树节点，基于语法制导(SDD)的语义处理

- 难点：定义语法规则，消除移进-归约冲突，归约-归约冲突（但其实并没有消除）

- 效果：符合C语言文法

* 一段程序只能由函数组成,即不支持定义全局变量（但是这个应该是可以改进的）
* switch case的case包含的语句块需要由花括号括起
* if,while,do while,for包含的语句块需要花括号括起
* 支持初始化字符串，写法为char s[]（=“string”），传参的时候也写成char s[]
* 不支持数组（元素是整数，浮点数）
* 没有实现指针，因为Javascript没有对应的数据结构
* 字符串声明以及赋初值等价于

|  |
| --- |
| char s[]; // -> s = new Array()  char s[10] = "abc"; // -> s=new Array() s=['a','b','c','\0'] |

* 单独对C语言的的几个库函数做了转化（但是之后应该会加多）

|  |
| --- |
| gets // ->readline-sync  printf // ->process.stdout.write,同时支持模板语法  strlen // ->s.length-1, 因为转化成的js Array添加了最后一个元素 \0  lenA // 获取聚合数据类型array长度 |

3. 中间代码生成：codegen.py

- 功能：遍历语法分析树ast,生成最终代码

- 实现方式：在yacc生成语法树时已经指定了树节点的类型[type]，自上而下遍历整个语法树，根据树节点的[type],对于每个在yacc.py内定义的非终结符，使用相应的${name}\_gen生成代码，对于每个终结符，使用terminator\_gen生成代码

- 难点：结合yacc.py定义的语法结构进行代码生成，代码缩进处理，库函数转化（但并没有说的这么难）

- 效果：实现了作业要求下的几个实例代码（但是可能还有其他代码）转Javascript的生成

五． 创新点

1. 较难的功能

- printf格式化输出转Javascript模板字符串

C语言Printf可以直接打印字符串，也可以模板化输出。对于抽象程度更高的Javascript而言，字符串本身就是可模板化的，我们利用这个特性实现了printf的库函数转化，例如：

|  |
| --- |
| printf(“hello, world!”); // -> process.stdout.write(`hello, world!`)  printf(“hello, %s”, name); // -> process.stdout.write(`hello, ${name}`)  printf(“Minus: [%d-%d=%d]”, a, b, a-b); -> process.stdout.write(`Minus: [${a}-${b}=${a-b}]`) |

- 字符数组存放方式

因为C语言本身是没有string类型的，C语言的字符串表达是char s[]或者char \*s,为了保持相似性，我们没有使用转化起来相对更简单的Javascript string而是使用Array,这样保证了结构的一致性，也保留了操作的一致性（如,按下标访问和末尾字符为\0）

- 缩进的实现

在codegen生成代码时，我们用Global变量存储当前代码缩进，我们精心设计了yacc的语法规则，只需要通过在非终结符block的递归调用与返回的过程中更新此变量便可生成Javascript的代码风格

2. C语言没有的功能（待添加）

- 布尔类型：定义了bool数据类型和true, false字面量

- 实现了聚合对象 array：array对应javascript里的array类型，其原始可以是int,double,float,bool,char,string等，也可以是这些对象的混合。实现了初始化和按下标取值和赋值操作，实现了针对array的方法lenA,用以获取Array长度，并实现了array作为函数参数和返回值

- 字符串直接运算？

- lambda表达式转箭头函数？

- 高阶函数？

- for … in/of 语句？

- struct与面向对象编程？

- 前端编程：C+html+css?

3. 实例代码（但是可能会有）

- ……

六． 实验总结