情報科学演習Ｄ　課題１レポート

氏名　山久保孝亮
所属　大阪大学基礎工学部情報科学科ソフトウェア科学コース
メールアドレス　u327468b@ecs.osaka-u.ac.jp
学籍番号　09B22084

提出日　2024 年 10 月 12 日
担当教員　桝井晃基　松本真佑

1　システムの仕様

課題 1 で作成したプログラムの仕様は以下の通りである.

- Pascal 風言語で記述された pas ファイルを入力として，字句解析の結果である ts ファイルを出力する．また，開発対象の run メソッドの第一引数は pas ファイル，第二引数は ts ファイルを指定する.
- 正常に処理が終了した場合は文字列”OK”を返し，入力ファイルが見つからない場合は文字列”file not found”を返す.

2　課題達成の方針と設計

課題 1 は字句解析と，解析したトークンをそれに対応する文字列とともに ts ファイルに書き込む処理に分けられる.

2.1　字句解析の処理

字句解析の処理は以下のオートマトンに基づいて開発した.

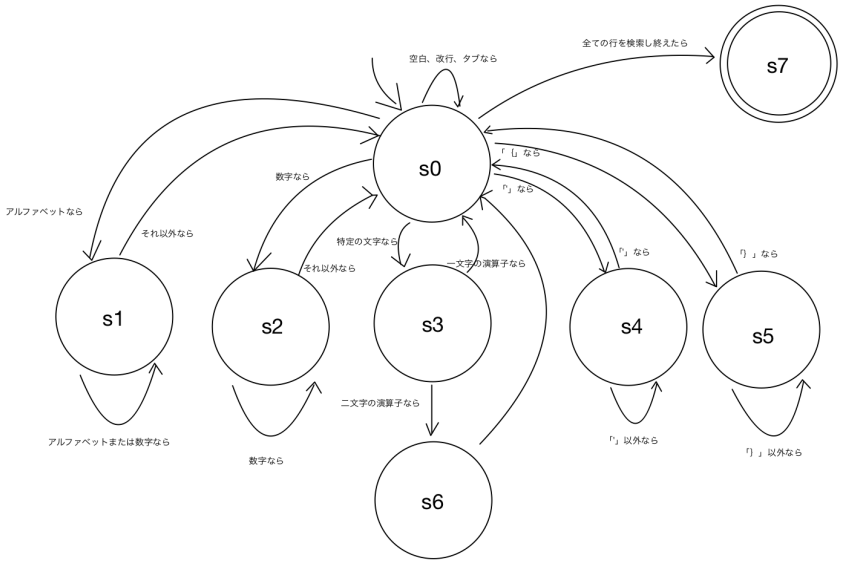

図 1: 字句解析部分のオートマトン

このオートマトンは合計 8 つの状態が存在し, 初期状態は s0, 受理状態は s7 である. それぞれの有効辺の条件分岐は入力された pas ファイルの一文字について判定している．状態が変わるとその次の文字について調べ，pas ファイルの最後の文字になると受理状態に遷移する．それぞれの状態において s0 に遷移する際に ts ファイルに出力する処理を行う．また，一字ずつ取り出す際に token_loaded_letter というリストを用いて，読み込み中のトークンの各文字を記憶しておく．そして，トークンの切れ目が確定してから各要素を合わせて文字列に変換する．その後 token_loaded_letter は初期化する．それぞれの状態の処理は 3 の実装プログラムで記述する.

2.2 ts ファイルに書き込む処理

ts ファイルに出力する処理を行うために，ソースコード上の字句，字句解析機上でのトークン名，ID のそれぞれを格納するリストを事前に用意しておく．[1] ソースコード中の字句を token，字句解析機上でのトークン名を token_name，ID を token_id というリストにそれぞれ格納しておく．そして，取り出したトークンが token の内いずれに当てはまるかについて判定し token_name と token_id に格納されている対応する文字列を書き込む．

3 実装プログラム

2 で記述した方針で作成するために，ここでは pas ファイルから文字を取り出す処理，字句解析のオートマトンのそれぞれの状態の処理，ts ファイルへの書き込み処理の 3 つについて記述する．

3.1 pas ファイルから文字を取り出す方法

pas ファイルから各文字を一字ずつ評価するために，pas ファイルから一行ずつ取り出し，各行から一字ずつ取り出す処理を行う．以下はその部分のプログラムを抜粋したものである．

```
1  final List<String> buffer = Files.readAllLines(Paths.get(inputFileName));
2  String line = buffer.get(i) + "\n";
3  while(i<buffer.size()) {
4      String line = buffer.get(i) + "\n";
5      while(j < line.length()) {
6          char c = line.charAt(j);
7          j++;
8      }
9      i++;
10 }
```

これによって，c にプログラムの各文字が格納されていくことになる．また，2 行目で line の最後に \n を追加しているのは Files.readAllLines() では改行の文字が含まれないためである．

3.2 字句解析のオートマトンの処理

まず図 1 のオートマトンの各状態に遷移するための条件分岐について記述する．3.1 で取得した c について評価してどの状態に対応するかを判定する．以下はその部分のプログラムを抜粋したものである．

```
1      //s0
2      if(c == ' ' || c == '\t'|| c == '\n') {
3      }else if(c == '{'){//s5
4      }else if(c == '\'') {//s4
5      }else if("/=<>+-*()[]:;.,".indexOf(c) !=-1){//s3
6          if((c == '<'||c == '>'|| c == ':'|| c == '.')&&( j + 1 < line.length()))
               {//s6
7          }else {//s3
8          }
9      }else if("0123456789".indexOf(c)!=-1){//s2
10     }else {//s1
11     }
```

この部分は 3.1 の二つの while 文中に記述されているので，pas ファイルの文字数分だけ実行される．また，”文字列”.indexOf(c) は文字列中に c が含まれているかどうかを判定しており，これで数字と特定の文字列が含まれているかを判定している．また，5 行目の s3 に遷移するかどうかを判定する際の文字列では，指導書の表 6 のトークン一覧に記載されていた記号を羅列したものでる．図 1 のオートマトンの特定の文字というのはこの””で囲まれた文字列に対応する．また，6 行目の s6 に遷移するかどうかを判定する際の文字列では，トークン一覧に記載されていた二文字の記号の内の一文字目を羅列したものである．

3.2.1 状態 1

というリストを用いて，トークンを設定する．

3.3 ts ファイルに書き込む処理

ts ファイルに書き込むための処理として，writeToken() を定義した．引数は以下の通りである．

```
1 void writeToken(String token_str, List<String> token, List<String> token_name, List<
      String> token_id, int line_counter, BufferedWriter writer)
```

token_str はオートマトンによって確定したトークンの文字列である．また，line_counter は token_str が記述されていた行番号を表す．具体的な writeToken() の処理内容は以下の通りである．

```
1  for(i=0;i<token.size();i++) {
2      if(token_str.equals(token.get(i))) {
3          tokenName = token_name.get(i);
4          tokenId = token_id.get(i);
5          break;
6      }else if(token_str.matches("\\d+")){
7          tokenName = "SCONSTANT";
8          tokenId = "44";
9          break;
10     }else if(token_str.indexOf("\'") !=-1){
11         tokenName = "SSTRING";
12         tokenId = "45";
13         break;
14     }
15 }
16 if(i == token.size()) {
17     tokenName = "SIDENTIFIER";
18     tokenId = "43";
19 }
20 writer.write(token_str + "\t" + tokenName + "\t" + tokenId + "\t" + line_counter);
21 writer.newLine();
```

まず token_str が token に格納されている字句と一致するものがあるかどうかを for 文と if 文によって判定している．一致するものがあれば tokenName にその字句に対応する字句解析機上でのトークン名が，tokenId に ID が格納される．token_str が符号なし整数である場合と文字列である場合はそれに対応するトークン名と ID が格納される．また，それぞれの判定方法は，符号なし整数であるかは正規表現で，文字列であるかは「’」が含まれているかどうかで判定する．token 内の字句，符号なし整数，文字列のいずれでもない場合は識別子であるので，16 から 19 行目のように i が token の要素数と同じかどうかの条件分岐を作成した．そして 20 行目のように writer.write() を使って指定した ts ファイルに書き込んだ．

4　考察や工夫点

課題 1 において工夫した点は以下の二つである．

- 一つ目は token，token_name,token_id の順番を対応させた点である．例えば，token，token_name,token_id の 0 番目にそれぞれ and,SAND,0 を格納したように，それぞれのリストのインデックスに対応する文字列を格納した．これにより，ts ファイルに出力する際に取り出したトークンが token のどの要素に一致するかを調べるだけで，同じインデックスを使って字句解析機上でのトークン名と ID を取り出すことができた．また，div と \ はどちらも格納し，token_name でも同じトークン名で，token_id でも同じ ID で格納した．
- 二つ目は ts ファイルに出力する処理をメソッドにしたことである．それぞれの状態から s0 に遷移する際にトークンを主強くする必要があるので，メソッドにしないと同じ内容を繰り返し書くことになるので冗長になってしまう．これにより，プログラムの可読性を高めることができた．

5　感想

今回の課題を通して字句解析機の作成方法を学ぶことができた．符号なし整数と識別子の定義を間違えて認識していたので，デバッグの際に少し苦戦した．プログラムを記述する前にオートマトンの仕様を完全に決定できるとスムーズに開発が進むと感じた．

6　謝辞

今回の課題 1 にあたって質問対応してくださった教授，TA の方々，有難うございました．今後の課題もよろしくお願いします．

参考文献

[1] 情報科学演習 D 指導書