神奈川大学理学部　情報科学科　馬谷研究室

スクリプト言語 Inutoba の処理系内部の調査報告

ドーベットアン　201902979

1　はじめに

通常、機械は人間が書いたプログラムをそのまま理解することはできない。そのため、人間が理解できる言語から機械語（機械が理解できる言語）に翻訳する必要がある。インタプリタは、処理方式の 1 つで、プログラムの実行時にコードを 1 行ずつ機械語に翻訳することが特徴である。私は Python を頻繁に使用しており、その Python でインタプリタの構造を知りたいと考えている。[1] を読み、インタプリタの実行の流れや字句解析や中間コードを調査する。

2　Inutoba 言語仕様

[1] で作成するインタプリタは Inutoba と呼ばれる。Inutoba の実装ソースコードは一つファイルだけである。Python でこのファイルを実行すると Inutoba インタプリタが起動する。

1. ソース

扱うソースはテキストファイルで、行の集まりである。行の先頭から改行までが 1 行である。Inutoba インタプリタではインデントには実行結果に影響がない。行の先頭に「#」を書けば、コメント行になる。

2. 定数、文字列定数

定数は１０進の実数のみ使える。小数点は定数内に 2 個以上あってはいけない。文字列定数はダブルクォーテーションを用いて”hello”のように書き、print 文、input 文でのみ使える。

3. 変数

１０進の実数のみで、文字列変数はない。

ローカル変数は予約語以外で、アルファベットかアンダースコアか数字の組合わせである。先頭は数字ではいけない。

グローバル変数は先頭が＄で、次はアルファベットかアンダースコアか数字の文字列である。

配列も使えるが、配列はグローバル変数のみである。使用時には次のように宣言する。

```
dim $xx[10]
```

配列変数は次のように使う。

```
$xx[3] = 26
```

4. 関数

関数の定義は func 文で始まり end 文で終了する。while, for, if 文のブロック内や、他の関数の内では定義できない。

3　インタプリタの実行の流れ

まず、Inutoba インタプリタの実行の流れについて、説明する。プログラマは Inutoba に読ませるためのソース（テキストファイル）を用意する。Windows であれば、メモ帳などで作成できる。ソース名は Inutoba ソース先頭付近で定義されている変数 fname にセットする。フォルダはメインルーチンの

```
path, b = os.path.split(__file__)
```

で取得している。システムの変数、__ file __ には Python の実行ファイルがフォルダを含めて入っている。上のコードで変数 path にフォルダを取り出している。上で取得したフォルダの、fname で決まるファイル名としている。要するにメインルーチンの

```
source = readSource(path+os.sep+fname)
```

のところでソースファイルの位置がきちんとシステムに伝われば良いだけのことなので、path に適切なフォルダ名を入れれば良い。

4　字句解析

```
for(i = 1,s =0; i <=10; i++){
    s +=i;
}
```

s はトークン（字句）を格納する。「＝」、「０」、「１０」、「for」　もトークンである。「ｓ」は変数、「＝」は変数に数値を代入する動きがあり、「for」は繰り返しの操作を提示する。言語処理系はまずソースを読んだら、そのソースを分析して、「この部分は変数、数値...」とトークンに分けなければならない。これが字句解析処理である。

5　中間コード

1. 変数、数値、文字列の中間コードへの変換例

Inutoba インタプリタで扱う変数にはローカル変数とグローバル変数がある。ソースを順に調べ、ローカル変数は LTable に、グローバル変数は GTable に変数名など変数の情報をまとめて、登録する。

- 数値も、すべて VTable に登録する。例えば、12345 という数値が VTable[10] に登録されたとすると、中間コードは　「DblNum,10」　となる。
- 文字列は STable に登録する。例えば、「こんにちは」が STable[7] に登録された場合、中間コードは　「Str、７」　となる。
- 配列変数は：　インデックスの指定がある。配列は、すべて DTable に登録される。

LBracket は「[」、RBracket は「]」 を表す。

2. 中間コードの具体例

2. 1: 代入文

。ソース:　　ｘ＝４６

。中間コード　[Lvar][18][Assign][DblNum][20]

Assign は代入を意味する中間コードである。ローカル変数ｘが LTable[18] に登録された。４６という数値が VTable[20] に登録された。インデックスの１８や２０は説明のための例である。

2.2: While 文

。　ソース：　 while i ¡=10

。　中間コード:

[while][120][Lvar][3][LessEq][DblNum][5]

120 は while に対応する end の行番号である。LessEq は「¡=」の中間コードである。i が LTable[3] に、１０が VTable[5] に登録されている場合の例である。

2.3: if 文

。　ソース：

```
if x ==23
...
else
...
end
```

。　中間コード:[If][46][54][Lvar][13][Equal][DblNum][35]

46 は次の elif か else か end の行番号である。54 は if に対応する end の行番号である。Equal は「==」の中間コードである。２３が VTable[35] に登録されている, とした。

。　ソース　... else

。　中間コード：　[Else][54][54]

１つ目の５４は else の次の end の行番号、２つ目の５４は else に対応する end の行番号である。要するに else の後には同じ行番号が 2 つ並ぶ。

6　まとめ

本研究ではスクリプト言語 Inutoba の処理系内部を研究して、インタプリタの実行の流れや字句解析や中間コードを十分に認知している。今後の課題としてインタプリタの構文解析を理解したいと考えている。

参考文献

[1] 吉田　節、Python でインタプリタを作る本、2021 年 2 月 26 日、株式会社インプレス RD