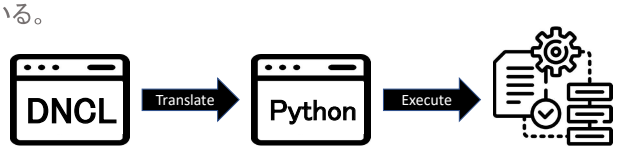
DNCL→Python翻訳プログラムの開発

早稲田大学高等学院3年 鈴田夏都季 ソースコード及びテストデータ(URL) : https://github.com/k1suxu/DNCL-Python-Translator

1.研究の概要

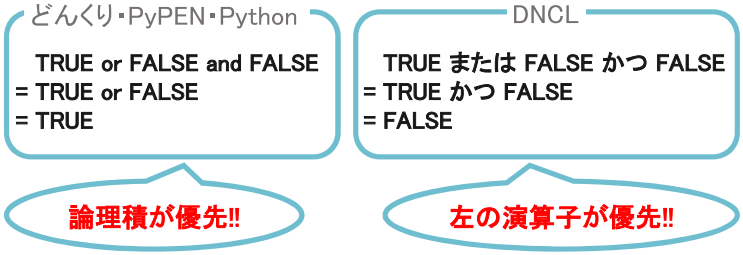
DNCL(2011年度時点の仕様[1])(以下DNCL2011)をPythonに翻訳するプログラムをC++を用いて開発した。DNCLとは大学入学共通テストの科目の一つである「情報関係基礎」にて用いられている**日本語記述型の疑似言語**である。今回私は、Pythonとの文法の互換性を実践形式で学べるようにすること、DNCL2011を実機で実行可能にすることを目標として作成を行った。そのため、本翻訳プログラムを、Pythonを中間言語としてDNCL2011を実行するためのトランスパイラのようなものとして位置付けている。



なお、研究に用いたテストデータやその出力結果、翻訳プログラムのソースコードはタイトル右下のURL(またはQRコード)にて閲覧できる。

2.先行研究

現在、DNCLをWeb上で実行できる環境として、どんくり[2] 、WaPEN[3]などが提供されている。また、PyPEN[4]においてはDNCLの文法体系や挙動をPythonに寄せたものをPythonに翻訳することができる。今回の翻訳プログラムが上記先行研究と異なる点は、**文法が仕様書通りである**DNCLと翻訳後のPythonプログラムが**全く同じ挙動**を示すように作成されているという点である。例えば、どんくり・PyPENは論理演算子の挙動がDNCLの仕様書と異なり、WaPENにはコードをPythonに翻訳する機能がついていない。



DNCLからPythonへの移行をスムーズにし、文法の差を実感しながらプログラミング学習を進められるようにするためには、**仕様書通りのDNCLをより忠実にPythonに翻訳**できるようになることが必要だと考えた。

3.方法(翻訳プロセス)

今回の翻訳プログラムは、**「文章整形」「文法解析」「翻訳後コード生成」**の3段階で翻訳を行う。また、そのアルゴリズムは**決定的**である。作成終了後、テスト用データ(どんくりのサンプルプログラム(改)+α)を用いて翻訳機が**正常に動作しているかを確認した**。

1. 文章整形①

まず、与えられた翻訳前コードの書き方(空白、インデント、大文字など)を統一する。これは、翻訳アルゴリズムの簡略化につながる。

図1: 文章整形①の動作例

2. 文章整形②

次に、論理演算子の優先順位の違いによる挙動の差が生まれないように条件文を翻訳する。Pythonでは論理演算子の優先順位が**「括弧→not→and→or」**の順だが、DNCLでは**論理演算子自体に優先順位はなく常に左の演算子が優先**される(括弧は使用可能)。DNCLの仕様に沿ってPythonプログラムを動作させるために、演算子が出現するたびに適当な場所を括弧で囲むという方法でこれを解決した(右図2)。

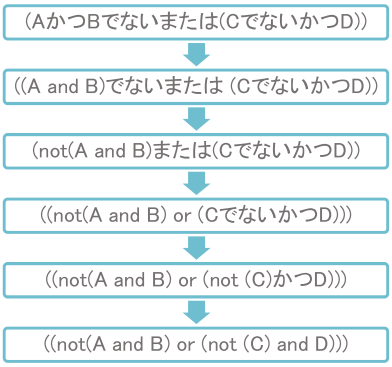


図2: 論理演算子の翻訳例

3. 文法解析

本翻訳プログラムでは、文法項目ごとの固有な**接頭辞**や**接尾辞**(表1)を見つけることにより文法解析を行った。

表1: DNCL文法と接頭辞接尾辞の対応

文法項目(DNCL)	固有な接頭辞・接尾辞
条件分岐1:If文(改行あり)	接頭辞:「もし」, 接尾辞:「ならば」
条件分岐2:If文(改行なし)	接頭辞:「もし」
条件分岐3:Else if文	接尾辞:「を実行し,そうでなくもし」
条件分岐4:Else文	接尾辞:「を実行し,そうでなければ」
繰り返し処理1:While文	接尾辞:「の間,」
繰り返し処理2:増加for文	接尾辞:「増やしながら,」
繰り返し処理3:減少for文	接尾辞:「減らしながら,」
表示関数:print	接尾辞:「を表示する」
インデントの引き下げ(スコープの区切れ) **Pythonではインデントの量で挙動が変わるため上手くやる必要がある**	2パターンあり ①全文:「を実行する」 ②全文:「を繰り返す」
その他(変数)	なし

4. 翻訳後コード生成

文法解析の結果をもとに、それぞれの文法項目に適した関数を用いて翻訳後コードを生成した(実装を参照)。一行で記述するif文に関しては、再帰的な処理を行った(図3)。またDNCLにおいては、配列型変数の初期化の際、**その次元数や要素数が宣言されない**という特徴がある。そのため、配列型変数に対してアクセスが行われるたびに必要な**次元数**を更新し、要素数については**defaultdict**[5]というPythonライブラリを用いることで、翻訳を行った。(k次元配列→k重ネストのdefaultdict)。

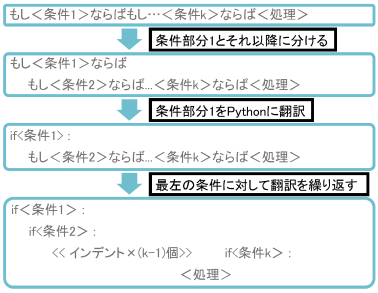


図3: 一行if文の再帰的翻訳過程

4.結果

完成した翻訳機を用いていくつかのDNCLサンプルプログラムを翻訳・実行した。その結果、翻訳前のDNCL2011と翻訳後のPythonが同じ挙動を示していることが確認できた。

5.考察・展望

今回私は、決定的なアルゴリズムを用いてDNCL2011をPythonに翻訳するプログラムを作成した。現状ではDNCL2011のみにフォーカスを当てているが、私が最終的に目指すのは、より**柔軟な記述が可能となる日本語プログラミング言語**の開発である。今後はAI・深層学習などを活用し、上記目標達成のためのDNCL機能拡張を目指す。

[1]独立行政法人大学入試センター. (2011). センター試験用手順記述標準言語(DNCL)の説明. https://web.archive.org/web/20220707074107/https://www.dnc.ac.jp/albums/abm00004841.pdf, (参照2023年1月31日).
[2]本多佑希・兼宗進. (2019). DNCLのオンラインプログラミング学習環境「どんくり」の開発. 第81回全国大会講演論文集, 2019(1), 583-584.
[3]中西渉. (2018). WaPEN...DNCLのWebブラウザ上の実行環境におけるフローチャートなどの実装. 情報教育シンポジウム論文集 2018.31, 210-214.
[4]中西渉. (2021). Webブラウザ上のプログラミング学習環境 PyPEN を用いた授業の提案. 第83回全国大会講演論文集 2021.1, 411-412.
[5]Python Decimal, https://docs.python.org/3/library/decimal.html, (参照2023年1月31日).