

日本語プログラミング言語によるプログラムの記述

馬場 祐人[†] 笥 捷彦[‡]早稲田大学大学院 基幹理工学研究科[†] 早稲田大学 基幹理工学部[‡]

日本語プログラミング言語において、日本語として違和感のない構文や命名規則を検討し、実用的なロジックを含むプログラムを挙げる。

1. はじめに

本研究では、日本語プログラミング言語において、他人がプログラムを読んでもプログラムのロジックを自然に理解できるような日本語として違和感のない構文と命名規則を考えることを目的とする。今回構文を検討するにあたり、検討した構文を筆者らが作成している日本語プログラミング言語「プロデル」^[1]の言語仕様に反映し、検討した命名規則を適用して複数のプログラムをプロデルで作成した。

2. 日本語プログラミング言語の利点

プログラムを日本語で記述すると、日本人の作成者は、日本語で思い浮かべた設計を英単語の表現へ置き換えることなく、直にプログラムで表現できる。さらにプログラムを読む者も日本人であれば、そのプログラムのロジックを読む者が直感的に理解できる可能性が高くなる。Java などの多くの言語では日本語を識別子として使うことはできるが、構文には英単語の予約語が使われている。日本語プログラミング言語は構文も日本語で記述でき、日本人が自然に感じる書き方で記述できる。そこで日本語として違和感のない記述ができ、しかも正確に表現ができる命名法も併せて検討する。

3. 定義文と命名規則

3.1. メソッド

メソッドは、動詞で命名する。メソッドはオブジェクトに対する振る舞いであるから、メソッド名は自然と動詞になる。プログラムを読む際にも、動詞で書かれた部分があれば、それが

メソッド呼び出しであることは違和感なく理解できる。メソッドの定義は次のように記述する。なお下線部がメソッド名にあたる。

初期化する手順

...

終わり

多くのメソッドは、動詞で命名することができるが、数値計算や真偽判定を行うメソッドは、「...を求める」「...を判定する」などとどれも同じ動詞になりがちである。そこで、これら戻り値を持つメソッド名には名詞を用いることとし、次のように別の形式のメソッドの宣言文を用意する。なお、仮引数とそれに対応する助詞は()の中に記述する。

最小値を求める手順

...

終わり

([結果] が) 既出かどうかを判定する手順

...

終わり

3.2. 変数

変数は、名詞で命名することが望ましい。局所的な変数は、「番号」「値」などの抽象的な名称で構わない。広域変数やメソッドの仮引数には、それが持つ値の役割がわかる明確な名称を付けることが望ましい。

3.3. 配列

配列は、変数の場合と同様に命名する。配列の添字には、次のように複数の構文を用意する。下線部が添字にあたる。

データ(10)	... (1)
データの 10 番目	... (2)
データの番号が <u>現在番号</u>	... (3)

(1)は BASIC や C 言語に見られる構文であり、多くのプログラミング言語で採用されている記

Programming in a Japanese Programming Language

[†]BAMBA YutoGraduate School of Fundamental Science and Engineering,
Waseda University[‡]KAKEHI Katsuhiko

Faculty of Science and Engineering, Waseda University

述方法である。また(2)と(3)は日本語での構文として導入した記述方法である。(2)は添字が定数である場合に自然な記述である。しかし(2)の構文では、添字に変数や式を指定すると違和感がある。そこで(3)に示すように指定する。

4. プログラムの記述例

3章で用意した構文と命名規則を用いて、作成したプログラムを示す。

4.1. 簡単なアルゴリズムの記述

簡単なアルゴリズムの記述例として最小公約数の計算を行うプログラム(リスト1)を示す。

リスト1 最小公約数の計算プログラム

1071と1029の最大公約数を表示する

```
([一つ目]と、[二つ目]の)最大公約数を求める手順
もし二つ目が0なら、一つ目の値を答えにして抜ける
一つ目を二つ目で割った余りを新しい値とする
二つ目と新しい値の最大公約数を答えにして抜ける
終わり
```

4.2. 実用的ロジックを含むプログラムの記述

実用的なロジックを含むプログラム(リスト2)を示す。リスト2は、Donald E. Knuthの減算アルゴリズムを利用して乱数を生成し、重複のないパスワードを生成するプログラムである。

5. まとめ

日本語プログラミング言語の利点は、プログラムの作成者と読む者が共に日本人であれば、作成者が日本語で考えた設計を直にプログラムへ表現し、読む者もプログラムから設計を自然に理解できることである。本研究では、日本語として違和感のない文法と命名規則を考えた。100行程度の規模のプログラムでは日本語として違和感なく記述できた。今後コンパイラのようなメソッドや変数を多く定義するプログラムを、日本語として違和感なく、また設計を理解できる形で、記述できるかどうか検討したい。

参考文献

- [1] 日本語プログラミング言語「プロデル」
<http://rdr.utopiat.net/>
- [2] 岡田健, 大岩元. 日本語プログラミング言語「言霊」におけるメソッドの記述方法. 第46回プログラミング・シンポジウム. 2005.

リスト2 パスワードの生成プログラム

乱数を起動時間で初期化する
(パスワード生成機能が8文字10個連続生成したもの)を表示する

パスワード生成機能とは
記号列は、"0123456789abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ_-"
生成済みパスワードフラグは、{ }

```
([生成桁数]文字、[生成数]個)連続生成する手順
パスワード一覧は、{ }
生成番号を1から増やしながら生成数まで繰り返す
パスワード一覧(生成番号)に、(生成桁数)文字生成したものを入れる
繰り返し終わり
パスワード一覧を答えにして抜ける
終わり
```

```
([生成桁数]文字)生成する手順
生成列は、{ }
文字位置を1から増やしながら生成桁数まで繰り返す
値は、(乱数の次の値を記号列の文字数で割った余り)+1
生成列(文字位置)に、記号列の(値)文字目を入れる
繰り返し終わり
生成パスワードは、生成列を繋げたもの
もし生成パスワードが既出なら、
(生成桁数)文字生成したものを結果にして抜ける
生成済みパスワードフラグ(生成パスワード)に、○を入れる
生成パスワードを結果にして抜ける
終わり
```

```
([生成パスワード]が)既出かどうかを判定する手順
生成済みパスワードフラグ(生成パスワード)を結果にして抜ける
終わり
終わり
```

乱数とは
乱数表は、{ }
現在番号

```
([シード値]で)初期化する手順
もしシード値が0未満なら
シード値に161803398を足す
でなければ
シード値から161803398を引く
もし終わり
乱数表の55番目にシード値を入れる
乱数値に1を入れる
番号を1から増やしながら54まで繰り返す
次の番号は、(21×番号)を55で割った余り
(乱数表の番号が次の番号)に乱数値を入れる
乱数値にシード値-乱数値を入れる
もし乱数値が0より小さいなら、乱数値に2147483647を足す
シード値に(乱数表の番号が次の番号)を入れる
繰り返し終わり
```

```
4回繰り返す
番号を1から増やしながら55まで繰り返す
乱数表(番号)から
乱数値(1+[(番号+30)を55で割った余り])を引く
もし乱数表(番号)が0より小さいなら、
乱数表(番号)に2147483647を足す
繰り返し終わり
繰り返し終わり
```

番号に0を入れる
終わり

```
次の値を求める手順
現在番号に1を足す
もし現在番号が56以上なら、現在番号に1を入れる
もし現在番号が34以下なら
乱数値は、乱数表の番号が現在番号
—乱数表の番号が(現在番号+21)
でなければ
乱数値は、乱数表の番号が現在番号
—乱数表の番号が(現在番号-34)
```

```
もし終わり
もし乱数値が0より小さいなら、乱数値に2147483647を足す
乱数表の(番号が現在番号)へ乱数値を入れる
乱数値を結果にして抜ける
終わり
終わり
```