

練習問題 2 のテストデータおよび解答例に誤りがありましたので訂正しました。また、演習問題 r7 全般のテストデータを空白をいれて少し読むやすくしました。テストデータは、問題文から copy&paste して解答例のようにプログラムに記述しておくことで打ち間違いを少なくしてください。[2025/05/24]

質問等について

プログラムが未完成だった場合、どこまで取り組めてどこがわからないかコメントを使って記述してください。完成できた場合でも質問があれば書いてください。

時間をかけてききたい場合は TA office hour を利用するなどしてください。

コメントの書き方 (... の部分がコメントになる)

`% ...` 単一行内でのコメント

`/* ... */` 複数行にわたるコメント

日本語を使用する場合はプログラムの冒頭に次の一文をいれておくこと error, warning は出ない。

`:- encoding(utf8).`

練習問題

- 以下の文法 (\mathcal{G}_1) が定義されているとき、与えられた表現が項 (Term) か否かを判定する `isTm1(Term)` のプログラムを作成せよ。`isTm1(f(a))`, `isTm1(f(f(b)))` は成功し、`isTm1(c)` は失敗することを確認せよ。 \mathcal{G}_1 はプログラムが対象とする言語であり、プログラミング言語と混同しないように注意。**(\mathcal{G}_1 そのものをプログラムの一部として記述するのは間違いである。)**

(\mathcal{G}_1)

`Term ::= Alphabet | f(Term)`

`Alphabet ::= a | b`

- この記法は BNF 記法 (Backus-Naur Form または Backus-Normal Form) と呼ばれる代表的な文法記述方法に基づく。
- 大文字からはじまるものは非終端記号 (定義がある)、小文字からはじまるものは定数およびファンクタ (与えられたもの)

- 以下の文法 (\mathcal{G}_2) が定義されているとき、与えられた表現が項 (Term) か否かを判定する `isTm2(Term)` のプログラムを作成せよ。データ `f(a)`, `g(0, f(b))`, `g(a)`, `f(g(a,a))` について動作確認せよ。

(\mathcal{G}_2)

`Term ::= Alphabet | Digit | f(Term) | g(Term,Term)`

`Alphabet ::= a | b`

`Digit ::= 0 | 1`

演習問題 (r7)

* のついている問題はオプションなのでできる者のみ解答せよ .

- (1) 以下の文法 (\mathcal{G}_3) が定義されているとき , 与えられた表現が項 (Term) か否かを判定する `isTm3(Term)` のプログラムを作成せよ . データ `g(f(a), 0)`, `g(0, f(b))`, `g(f(0), b)` について動作確認せよ .

```
Term ::= Alphabet | Digit | f(Term) | g(Term,Alphabet)
Alphabet ::= a | b
Digit ::= 0 | 1
```

- (2) 文法 (\mathcal{G}_2) になかったもののみがデータとして与えられるとする . 与えられた項 Term に Alphabet が出現する回数が C であるという関係を表す述語 `count_alphabet(Term,C)` のプログラムを作成せよ . `r6(2)` で作成した述語 `count_ocr` を使用するのが簡単だが , これを使用せず独自の述語を作成してもかまわない . たとえば `count_alphabet(g(0, f(g(b,f(a)))), C)` は `C=2` となって成功する .
- (3) 文法 (\mathcal{G}_2) になかったもののみがデータとして与えられるとする . 与えられた項 Term に含まれるファンクタ f の数が N であるという関係を表す述語 `n_of_f(Term,N)` のプログラムを作成せよ . たとえば , `n_of_f(g(a, g(f(0),1)), N)` は `N=1` となって成功する .
- (4) 文法 (\mathcal{G}_2) になかったもののみがデータとして与えられるとする . 与えられた項 Term に出現するファンクタ f の数と , g の数の比較をし , 多い方が F であるという関係を表す述語 `compare_fnumbers(Term,F)` のプログラムを作成せよ . 同数の場合は , `F=same` とせよ . 必要な述語は自分で定義せよ . たとえば , `compare_fnumbers(g(a, g(f(0), 1)), F)` は `F=g` となって成功する (`r7(3)` と同様に `n_of_g` を定義し , `n_of_f` の結果と比較するのが基本的な手法だが , `f,g` の数を求める共通の述語を生成する方が望ましい . できる人はこちらでやってみてください .)
- (5) 文法 (\mathcal{G}_2) になかったもののみがデータとして与えられるとする . 与えられた項 T1 に出現する a,b を それぞれ 0,1 に置き換えた結果が T2 であるという関係を表す述語 `subst_ab(T1,T2)` のプログラムを `r6(1)` で作成した `subst_atom` を使用して作成せよ . たとえば `subst_ab(g(g(0,a), f(g(b,a))), T)` は `T = g(g(0,0), f(g(1,0)))` となって成功する . (まず a を 0 に置き換えその結果に対して b を 1 に置き換える と考える .)
- (6)* 論理式が以下のように定義されたとする . 与えられた表現 E が論理式であるかどうかを判定する述語 `isFormula(E)` のプログラムを作成せよ . たとえば , `isFormula(and(p,neg(q)))`, `isFormula(imp(and(p,q), p))` は成功し , `isFormula(or(and(p,q)))` は失敗する .

```
Fml ::= Atm | neg(Fml) | and(Fml,Fml) | or(Fml,Fml) | imp(Fml,Fml)
Atm ::= p | q
```