コンピュータの操作の応用
a. プログラミングの基本

分子系統学的な研究を行うとき，コンピュータを用いて様々な情報を取り扱う場面に出くわす場合が多い．このようなとき，処理をさせるための簡単なプログラムを作成できると大変便利であり，研究の効率化につながる．コンピュータの取り扱いになれるために，今回はスクリプト言語 Perl を用いて簡単なプログラムを作成し，データ処理を行う．

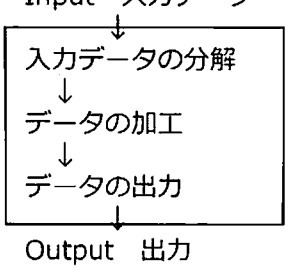
計算の方法

- 手（+足）
- 石・棒
- 暗算
- そろばん（東洋とくに日本で発達）
- 筆算
- 数表
- 計算尺（1/100 まで読み取れる）
- 電卓（電子卓上計算機）
- 関数電卓
- ポケコン
- Excel など表計算ソフト
- プログラム

プログラム

現在もちいられているコンピュータの多くは，ノイマン型コンピュータと呼ばれ，計算手続きを記したプログラムと計算に用いるデータをメモリ上において実行する形式をとる．このため，コンピュータに何らかの仕事をさせ，その仕事を自動化する場合，プログラムが必要となる．

プログラムの構造



Program (Script)

用語

プログラム program～コンピュータ上で動作するソフトウェア．コンピュータに行わせる計算などの手続きを順に示したもの．現在ではプログラム言語やスクリプト言語を用いて手続きを記述するのが一般的．
プログラミング programming～プログラムを作成すること．コーディング coding ともいう．
プログラム言語 programming languages～コンピュータ上で動作するプログラム（ソフトウェア）を作成するための人工的な言語体系．いくつかの命令語と文法をあわせたもの．

プログラム言語の分類～プログラム言語は，大きく分けて 2 つに分類できる．
低水準言語（低級言語）～コンピュータが理解できる 0/1 の 2 進数であらわされる機械語やそれを記号化したアセンブラ言語．
高水準言語（高級言語）～低水準言語に比べ，人間に理解しやすい人工的な言語．コンパイラを用いてアセンブラ言語に直し，さらにアセンブラを用いて機械語に変換する．現在用いられる代表的な高水準言語と歴史的な言語を以下に挙げる．

FORTRAN～科学技術計算向け．
COBOL～事務処理向け．

ALGOL～数値計算や論理演算向け.
Smalltalk～オブジェクト指向言語の元祖.
PL/I～科学技術計算および事務処理向け. FORTRAN や COBOL を参考に作成.
BASIC～初心者向け. FORTRAN を参考に作成.
LISP, Prolog, LOGO～人工知能向け.
Pascal, Ada～構造化されたプログラムの作成に適する.
C, C++～OS などシステムの記述に適する汎用言語. Pascal に類似するが, より簡潔な表記が可能で, かつ機能が強力. UNIX で標準.
awk～テキスト処理用. スクリプト言語の元祖の一つ.
Java, JavaScript, Python など～最近よく用いられる言語.
Ruby～日本産のスクリプト言語の一つ. 最近用いられる機会が多い. Perl と同じような機能をもつが, 言語体系が単純で使いやすい.
Perl～スクリプト言語の一つ. 高機能で, さまざまな言語の寄せ集め的な言語体系をもつ. CGI などでよく用いられている.

UNIX 文化
プログラムは小さく作り, 組み合わせて用いることが多い.

b. Perl について
Perl とは
Perl は, 1986 年 Larry Wall によって UNIX 用に開発された手続き型のインタプリタ言語. 行単位の文字列の処理, ファイル処理, プロセス処理などに向く. Practical Extraction and Report Language (実用データ取得レポート作成言語) または Pathologically Eclectic Rubbish Lister (病的折衷主義のがらくた出力機) の頭文字をとって Perl と呼ばれ, "パール"と読む. 現在の最新バージョンは Perl5.8. フリーソフトとして配布されており, UNIX およびそのクローン, Windows (ActivePerl), Machintosh (MacPerl) などさまざまな環境で利用可能. UNIX の文化を色濃く残すスクリプト言語.

Perl の特徴
長所
1. コンパイルは必要ない. インタプリタ型言語のため.
2. メモリの許す限りファイルサイズ, 再帰呼出し, ハッシュテーブルなどの制限がない. 手軽に書ける.
3. 勉強した分だけ実際に利用できる.
4. 対開発効率のコストパフォーマンスに優れる. 例えば, C 言語などの他の言語で 10 行程度の処理が Perl では 1 行で書けるものもある.
5. 強力な正規表現を利用できる.
6. さまざまな言語の長所をあわせ持つ.
7. さまざまな環境で利用可能.
8. UNIX で用いられる C や sed などに類似した書式であるため, UNIX に慣れている場合は違和感が少ない.

短所
1. コンパイル言語よりは処理速度は劣る.
2. パズル的なプログラムでは, 処理内容がわかりにくくなる.

c. Perl を使ったプログラムの作成
Perl の基礎
文法・書式
Perl スクリプトを構成し, 実行指示を行うコマンドの基本単位は文. 文には単純文と複合文がある. 文は, 関数や式, コマンドが組み合わされて構成される. 文の末尾には; (セミコロン) を置かなければならない. 単純文とは, 代入や関数の評価など副作用を得るための式のこと. 複合文とは, 文のならびを{ }で囲み, ひとまとまりにしたブロックから構成される. if 文や while 文などの制御構造で用いられることが多い. ブロックの最後の文に限り;を省略できる. 慣例で

{の後と，}の前は改行することが多い.

変数
スカラー～Perl の基本的なデータ型の一つ.
スカラー変数～$で始まる．$識別子で変数となる．識別子は英文字または下線（_）で始まり，英文字，数字，下線の組み合わせで構成される.

$A　　スカラー変数（値が一つしか代入されない変数）
例.　足し算
```
1:　　$A = 1; #スカラー変数 A に 1 を代入.
2:　　$B = 2; #スカラー変数 B に 2 を代入.
3:　　print $A + $B;　#A と B の和を標準出力へ表示する.
```

演算子
算術演算子

記号	細分類	機能
+	加法演算子	加算
-	加法演算子	減算
*	乗法演算子	乗算
/	乗法演算子	除算

代入演算子 =
　左辺値に式の値を代入する.

代表的な関数（命令）
print　出力
例.　入力したファイルをそのまま出力
```
1:　　while (<>) {　　#行がなくなるまで行の読み込みを繰り返す.
2:　　　　print;　#読み込んだ行を標準出力へ表示する.
3:　　}
```

while　繰り返し制御
例.　3 までカウントする.
```
1:　　$a = 1; #スカラー変数 a に 1 を代入する.
2:　　while ($a < 4) { #変数$a が 4 より小さい間括弧内を繰り返す.
3:　　　　print $a, "¥n";　#変数$a の値と改行コードを標準出力へ出力する.
4:　　　　$a = $a + 1;
5:　　}
```

<>　　ファイルから行を読み込む
例.　print の例を参照.

reverse 逆順に並び替える
例.　逆さに読む
```
1:　　$_ = <>;　　#標準入力から 1 行入力し，変数$_に代入する.
2:　　$a = reverse $_; #逆順にしたものを，変数$a 代入する.
3:　　print $a;　　#標準出力に表示する.
```

length　スカラー（文字列）の長さを調べ，代入する
例.　文字列の長さを調べる
```
1:　　$_ = <>;　　#標準入力から 1 行入力し，代入する.
2:　　$a = length $_;　#長さを変数に代入する.
3:　　print "length = ", $a;　　#標準出力に表示する.
```

s///g　置換演算子
例. 配列内のすべての a を A に置き換える.

```
1:      $_ = <>;        #標準入力から 1 行入力し，代入する.
2:      s/a/A/g;        #含まれるすべての a を A に置き換える.
3:      print;   #標準出力に表示する.
```

正規表現～文字列のパターンを表現する方法. 通常の文字および特殊な意味をもつ文字（メタ文字）で構成される.

メタ文字

文字	意味
.	任意の 1 文字（改行文字は除く）
*	パターンの 0 回以上の繰り返し
?	パターンが 0 回または 1 回現れる
¥*N*	右隣のメタ文字を意味するエスケープ文字
[*ABC*]	括弧内に含まれる文字のいずれか（文字クラス）（*A* か *B* か *C* のいずれかと一致する）

Perl の利用

Perl を利用する場合, Perl がインストールされている必要がある. Unix 環境では多くの場合, 標準で利用できることが多い. Windows では ActivePerl などをインストールする必要がある. Windows ではコマンドプロンプト上で動作する. このため, MS-DOS の知識が多少必要となる. また, スクリプトはエディタで作成するが, エディタプログラムか MS Word などの文章入力のためのプログラムも必要となる. できあがったスクリプトは, コマンドラインから実行する.

準備

まずはじめに, Perl をインストールする. 正式にインストールした場合, パスを通しておくとよい. インストール終了後, コマンドプロンプトを起動させ, 以下のコマンドを入力し,

```
> perl -e "print 'Hello, world¥n'"
```

コマンドラインから Hello, world!と返ってくればインストールされている. または,

```
> perl -v
```

でバージョンが返ってくればインストールされている.

Perl を使ったプログラム実行の流れ

1. フローチャートの作成
2. プログラムの作成
3. デバッグ
4. プログラムの完成
5. 実行

フローチャート（流れ図）の作成

作成するプログラムの大まかな流れを図の形であらわしたもの. プログラム言語によって処理が異なり, そのままの状態ではプログラムに変換できない場合もあるが, プログラムを作る際参考になるので, コーディング前に一度作るとよい.

プログラムの作成

エディタを用いて, スクリプトを作成する. MS Word などでも代用できる. ただし, テキスト形式で保存し, 保存したファイルの拡張子を*.txt から*.pl に変更しておいてもよい. また, 入力の際半角文字で入力すること.

デバッグ

プログラムが正常に作動するように, 誤りを修正していく作業. まず, 簡単なデータファイルを作り, それを用いて正常に動作するかテストする. 各行の後に, print 文を入れて変数の値を表示すると, 理解が容易になる.

プログラム（スクリプト）の実行

ソース
↓
インタプリタまたはコンパイラ
↓
翻　訳
↓
実　行

Perl スクリプトの実行

スクリプト名が *chotto.pl*，データファイル名が *tatoeba.txt* の場合．perl.exe にパスが通っている状態で，以下のコマンドを実行する．
> perl *chotto.pl* < *tatoeba.txt*
結果をファイル *deta.txt* に出力したい場合は，
> perl *chotto.pl* < *tatoeba.txt* > *deta.txt*

Perl スクリプトの終了

実行中の Perl スクリプトは，^C（コントロール + C）[1]で終了できる．

スクリプトの例

UNIX の場合は各スクリプトのはじめに，
```
1:      #!/usr/local/bin/perl
```
を入力すること．Windows の場合は不要．

スクリプト 1．入力ファイルをそのまま出力するスクリプト
```
1:      while(<>){
2:              print $_;
3:      }
```

スクリプト 2．入力ファイルの文字を置き換えるスクリプト
```
1:      while(<>){
2:              s/[nN¥?]/N/g;
3:              print $_;
4:      }
```

スクリプト 3．入力ファイルから ATGCN 以外の文字を除去するスクリプト
```
1:      #!/usr/local/bin/perl
2:      while(<>){
3:              s/[nN¥?]/N/g;
4:              s/[^aAtTgGcCN]//g;
5:              print $_;
6:      }
```

練習

1. 入力した塩基配列から ATGCN 以外の文字を除去し，その長さを出力するスクリプトを作成せよ．
2. 塩基配列を読み込み，その相補鎖を 5'から 3'の向きで出力するスクリプトを作成せよ．入力ファイルから ATGCN 以外の文字を除去する機能も付加すること．

参考文献

1. 伊藤和人．1997．入門 Perl -Perl for beginner's-．271 pp．秀和システム，東京．

[1] コントロールキー（CTRL，Control などと表記）を押さえたまま，C のキーを押さえる．途中で離さないこと．

(ISBN 4-87966-751-X)
2. Schwartz, R. L. (近藤嘉雪訳) 1995. 初めての Perl. xxxii + 305 pp. ソフトバンク, 東京. (ISBN 4-89052-678-1)
3. Wall, L., T. Christiansen & J. Orwant. 2000. Programming Perl, 3rd Ed. O'Reilly & Associates, Sebastopol. (ISBN 0-596-00027-8)
4. 藤田郁・三島俊司. 1999. CGI & Perl ポケットリファレンス. 463 pp. 技術評論社. (ISBN 4-7741-0755-7)

課題

以下の課題をレポートにあわせて提出すること.

1. 以下の Perl スクリプトがある. このスクリプトは以下の入力ファイルのように, ATGC のみからなる塩基配列を入力した場合, 一定の規則のもとに値を出力する. このスクリプトにより, 何が求められるか. また, 各行で行われる処理を説明せよ. さらに, 例で挙げられている塩基配列を入力した場合に得られる値を求めよ.
ヒント. PCR 反応で必ず必要なものは？ DNA 合成酵素と…

入力ファイル
CAATGCATGCAGTTATTGAC

スクリプト

```
1:      #!/usr/local/bin/perl
2:      $_ = <>;
3:      s/[^AaTtGgCc]//g;
4:      s/[AaTt]/2/g;
5:      s/[GgCc]/4/g;
6:      $i = length $_;
7:      $count = 0;
8:      $sum = 0;
9:      while ($count < $i) {
10:             $j = chop $_;
11:             $sum = $sum + $j;
12:             $count = $count + 1;
13:     }
14:     print "Tm = ", $sum;
```

補足
chop　文字列の最後の文字を切り落とし, 切り落としたものを変数に代入する

2. 生物学の実験で必要な計算をさせるオリジナルの Perl スクリプトを 2 つ作成せよ. 実習中に用いていない関数を利用してもよい. どうしても思いつかない場合は, スクリプト 4 または課題 1 のスクリプトを別の方法に書き換えたり, CG 含量を出力させるなど改変し, 機能を付加してもよい. いずれの場合も, スクリプトと実行例を示すこと.
例 1. リン酸バッファーを作る際, 混合する 2 つの液の比率と得られる pH.
例 2. DNA データベースからアミノ酸をコードする DNA 領域の配列を得たとき, アミノ酸配列を類推する.