■解説（VISION Vol. 22, No. 2, 123–130, 2010）

近年の PC 向けアーキテクチャを利用した視覚刺激の提示：Psychlops における実装

細川 研知 * ・丸谷 和史 ** ・佐藤 隆夫 ***

* 東京大学 インテリジェントモデリングラボラトリー
〒 113–0033 東京都文京区弥生 2–11–16
** NTT コミュニケーション科学基礎研究所
*** 東京大学 大学院人文社会系研究科

1. はじめに

現在の視覚の心理物理学実験では，コンピュータを用いて，ディスプレイの上に刺激を表示し，キーボードや反応ボックスを使って被験者がそれに対する応答を行うというやり方が一般的に用いられている．この方法で刺激を描くには，自分が望んでいることをプログラムという形でコンピュータに命令する必要がある．PsychToolbox や，Visage といった近年のいくつかのツールの登場によって，現在のプログラミングは一昔前と比べるとずいぶん簡単になっている．たとえばプログラミングにおいて一番面倒なのは，コンピュータのハードウェアやオペレーションシステムに自分の命令を聞かせるための準備をさせることである．これらの手続きは実験の本質からはほど遠いにもかかわらず煩雑で，すべてを理解するのは大変であるが，現在ではユーザはこの手続きを理解する必要はほとんどない．この準備は，いつもほとんど同じ手順，手続きで行われるので，これをまとめておけば細かいことを意識せず使うことができる．実際に，心理物理学でよく使われるツールセットの内の大部分で，これらの手続きはツール側が代行していて，ユーザは一連の手続きを呼び出すための短い命令を書くか，単にダブルクリックをしてアプリケーションを立ち上げさえすればよいようになっている．このように，プログラミングのためのツールは日々進歩していて，ユーザがより簡単に自分のやりたいことをできるようになってきている．

しかし，依然として実際の実験準備の段階において，プログラミングが完全に不要になったわけではない．心理物理学実験の初心者にとっては，このプログラミングが実験を行うときの大きな壁であり続けている．読者の方々にも，初めて研究室でガイダンスを受けたときに，プログラミングを習得しなければならないと聞いて，憂鬱な気分になったことがある人がいらっしゃるのではないだろうか．初学者が克服するべき最も大きな壁は実際のプログラミング技術習得の難解さによるというよりは，「プログラミング＝難しそうだ」というイメージなのではないだろうか？

このプログラミングに対するネガティブイメージの原因の一つとして，必要とする情報が整理された形で初学者に与えられていないことがある．一般的なプログラミング教本には，視覚実験に必要なプログラミング技法のすべてが含まれているわけではなく，また余分な内容も多い．実際に視覚実験を組むために必要な基本命令はそれほど多いわけではない．組み合わせまで考慮しても，いくつかの基本形を知っていれば，たいていの実験をプログラミングすることができる．この基本形は開発者たちによって公開されていることがほとんどで，実際にはこの雛形

2010 年冬季大会チュートリアル講演．

を少し書き換えるだけで実験プログラムを完成させることができる．ここでは，典型的な心理物理学実験の状況を取り上げ，これをコンピュータ上で実現するためのプログラミングの基本について，筆者らの開発しているPsychlopsというフリーのツールを例にとって説明したい．

2. プログラムの前の準備

Psychlopsを使って実験プログラムを書くときの最も簡単な方法は，Psychlops-wiki（http://psychlops.l.u-tokyo.ac.jp/）やVisiome（http://visiome.neuroinf.jp/）などのウェブサイトにアクセスして，自分が使いたい実験プログラムの雛形を見つけることから始まる．これらのウェブサイトには，Psychlopsを用いたデモプログラムが数多くアップロードされている．これらのサイトからユーザがサンプルコードを手に入れることはできるし，ここで，手に入れたプログラムを闇雲に試してみることもできるだろう．

しかし，より効率よくプログラムを完成させるためには，それらのどれが，実際に今自分が望んでいる実験プログラムに適しているか，最も近いかを判断できるとよいだろう．これには，刺激と実験計画とプログラムのイメージがあらかじめ明確に対応する必要がある．これさえできていれば，ユーザはほとんどプログラミングを考える必要はなく，基本的なコードをサンプルプログラムからコピー＆ペーストし，ごくわずかな修正をするだけでプログラムを完成させることができる．極端な場合，自分の提示したい刺激に対応したルーチンをコピーして，パラメタを指定するだけで，実験を施行することが可能である．

刺激と実験計画とプログラムのイメージを対応させる方法は，実験演習などで求められる刺激と実験計画について簡単なレポートにまとめるときのことを考えると理解しやすい．典型的なレポートの「方法」は「刺激」と「手続き（実験計画）」から構成されている．もう少し詳しくこれを書けば以下のようになる．

2.1 刺激をデザインする

実験刺激をつくるには，まずどのような刺激をつくるかを決める．刺激の模式図をつくる場合，基本的には時間ごとにどんな画像を提示したかを描く．典型的には，刺激の一試行は時間ごとにブランク提示，マスク提示，ターゲット提示，反応待ちなどのいくつかのセクションに分けることができる．このそれぞれのセクションに対して，どのような絵を出すかを記述すればレポートの「刺激」の部分が完成するだろう．

どのような絵を出すかということの記述は，どんな図形をどんな色でどの位置にどの時間提示するかを定義することで行われる．たとえばブランク刺激であれば定義すべきことはどのような色をどの程度の時間提示したかである．正方形を提示したのであれば色と一辺の長さ，縞刺激であれば平均輝度と振幅，波長，位相を定義し，それが画面上のどこにあったかを決める．複数の要素図形から構成される絵であれば，それぞれの要素図形がどんな色でどの位置にあったかおのおの記述する．運動刺激であれば，図形が時間を追うごとにどう変化するかを定義することで刺激を決めることができる．

2.2 実験計画を設計する

一般に心理物理学実験では，物理刺激の条件（独立変数）を決まったやり方で変え，それにより被験者の反応（従属変数）が変わるかを測定する．したがって，条件の変え方や反応の取り方も記述する．条件の変え方については，恒常法や上下法などのような方法で何回試行を繰り返すか定義する．また刺激を提示し，被験者の反応をどう取得するか，強制二肢選択法か調整法か，実験実施時に具体的にどのキーを使うかといったことを決める．

これと同じように，刺激と実験計画を分けて考え，それぞれに対応するプログラムのサンプルパーツを組み合わせれば雛形は完成する．現代のプログラミング言語ではオブジェクト指向といわれるプログラミングの方式が主流となっている．このオブジェクト指向を採用したツールでは，パーツはバラバラに集めてきても，一

定の手続きに沿ってこれを配置すればプログラムが動作するように設計が行われている．逆に，ウェブ上にあるサンプルプログラムはこの手続きに沿って定型化されたプログラミングが行われていて，その定形を知ればどこからどこまでが刺激描画を担当する行であるか，どこが手続きを担当する行であるかを簡単に読むことができる．Psychlops もオブジェクト指向のプログラミングが可能であるような設計が行われていて，論文やレポートに日本語や英語で書いている内容を，オブジェクトを並べ直し，セクションを構成してプログラムに「翻訳」すれば，コンピュータで刺激提示ができる．以下ではそのオブジェクトやセクションの扱い方について簡潔に説明する．これを知ることで，ユーザはサンプルプログラムから自分のプログラムに移植可能なパーツを切り出すことができる．

3. 設計した刺激をプログラムにする

図 1 に，プログラムで刺激を記述する最も典型的な例を示す．プログラムは大まかに前半の刺激描画部と後半の実験計画部に分かれている．まずは，前半の刺激描画部について説明する．刺激は時間ごとに動画提示，反応受付などいくつかのセクションに分かれており，個々のセクションはそれぞれの時間にどんな図形を提示するかというパーツから構成されている．このような階層構造の中で，自分の実験に合わせて個々のパーツを作成し入れ替えてプログラムを作成できる．これらのセクションは，刺激の模式図のセクションにそれぞれ対応する．

刺激の基本は一枚の絵を描くことである．まず背景輝度を決め，いくつかの図形を大きさや場所を決めて並べ，すべての図形を配置し終えたらそれをまとめて表示することで一枚の絵が完成する．静止画であればその絵を一定時間表示し，動画であれば時間を変数として図形を書き換えていくことで，時空間的な刺激の定義，言い換えれば一つのセクションが完成する．

以上の流れはほとんどの図形描画に共通する定型文であり，図形を定義するパーツを組み合わせてセクションとなるパーツを構成し，セクションパーツを組み合わせて刺激描画パーツを構成し，刺激描画パーツと実験計画パーツを組み合わせて実験プログラムが完成するという階層構造を持つ．次に，自分の実験に合わせてパーツをカスタマイズできるよう，Psychlops における個々のパーツの詳しい取り扱い方について説明する．

3.1 セクションを仕切る

実験刺激の模式図では，時系列でセクションを区切ってどんな絵を提示したか示す．これをコンピュータで再現するには，どの時間にどんな絵を出すかということを記述することで一つのセクションができあがる（**図 2**）．セクションが複数ある場合にはセクション記述も複数記述する．

コンピュータは映画やテレビと同じように，1 秒間に 60~120 枚の絵（コマ，フレーム）を繰り返し表示する．言い換えれば，コンピュータで刺激提示を時系列で配置するには，どのコマに何コマ連続して出すか決める．「何回もコマを出す」ことを示すのが for 文で，表示するコマ数を丸括弧の中に書き，その後に続く波括弧の中にそこで表示する図形パーツを組み込んでいくことでセクションを区切る．提示時間の秒数を決めるには，1 コマあたりの秒数とコマ数を掛ければよい．OS の画面解像度設定でリフレッシュレートを調べ（通常は 60 Hz 程度），その逆数が 1 コマあたりの秒数になる．図 2 の例では，「表示コマ数の指定」でこのセクションを 60 コマ表示し，式に「t」と書けば時間の変数となるよう指示している．このとき，for(int t=0; t<60; t++) という書式は一字一句正しく書き，提示コマ数を指定する「60」の部分だけを変える．もし他の部分を操作すると，実行できなかったり永久にセクションから抜けないことがあるので，よく注意してコピー＆ペーストする必要がある．ただし，反応受付（図 1 の「反応受付セクション」）やデモを表示する場合にはコマ数制限を省いたり while 文を用い，キーを押すまで無制限に表示させることもある．

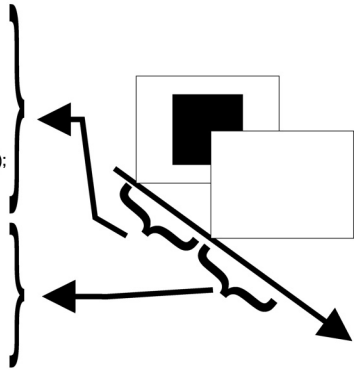
事前準備(定型)

```
#include <psychlops.h>
using namespace Psychlops;
Canvas cnvs;
```

刺激描画部

論文の「刺激」に相当

```
void trial() {
    動画セクション
    for(int t=0; t<60; t++) {
        cnvs.clear(Color(1.0, 1.0, 1.0));
        図形パーツ1
        Psychlops::Rectangle patch(100,100);
        patch.centering().shift(t, 0).draw(Color(0.0, 0.0, 0.0));
        cnvs.flip();
    }
    反応受付セクション
    for(;;) {
        cnvs.clear(Color(1.0, 1.0, 1.0));
        cnvs.flip();
        if(Keyboard::esc.pushed()) { break; }
    }
}
```



実験計画部

「実験手続き」に相当

```
void psychlops_main() {
   cnvs.set(1024, 768, Canvas::window);
   trial();
}
```

図 1　Psychlops による刺激記述の典型的な例．プログラムは刺激描画部と実験計画部，刺激のセクションやセクション中で描画する図形パーツなどを組み立てた構造になっており，それぞれのパーツが論文や模式図の一部分と対応する．

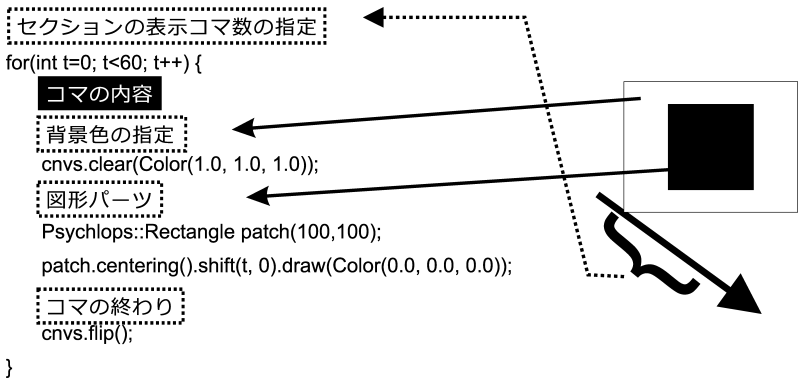


図 2　Psychlops による刺激描画セクションの定義方法．何コマ表示するか，それぞれのコマの中でどのような絵を表示するか定義している．

それぞれのセクション内部には図形パーツを配置してどのような絵を出すか記述するが，どんな図形を描くにせよ，基本的にはまず背景の色を決める必要がある．Psychlops では for 文の次，セクションの冒頭に clear 命令を書くことでそのセクションの背景輝度を指定する（図 2 の「背景色の指定」）．clear 命令の後の丸括弧の中に Color（赤，緑，青）という書式で色を

指定する．セクションの終わり，閉じ波括弧の直前には flip 命令を入れ，コマの終わりを明示してそこまでに配置した図形パーツを表示するようにする（図２の「コマの終わり」）．セクション内で提示する図形の定義は，clear 命令と flip 命令の間で行う（図２「図形パーツ」）．次に，その図形の定義方法について説明する．

3.2 各セクションに図形を配置する

セクションの始点を for 文と clear 命令，終点を flip 命令と閉じ波括弧で仕切ったら，次はその中に，そのセクションで提示したい図形パーツを必要な数だけ配置する．図形パーツの詳細として，図形の種類，個々の図形種に固有のパラメタ（円なら半径など），位置，色などを記述する（**図 3**）．

図形パーツのはじめに，まず図形種を決めその図形にその場での「名前」を付ける．どのような図形があるかについては**表 1** に示した．それぞれ図形種固有のパラメタについては，サンプルコードやドキュメントを参考に記述できる．位置を指定するには，図形の各要素が画面上のどの座標にくるかを記述する．Psychlops では，「画面の特定の座標にセンタリングする」「そこから相対的に動かす」という指示を組み合わせてこれを行う．Psychlops の座標系は古典的な行列座標（水平軸 x は右ほど大きく，垂直軸 y は下ほど大きく，原点は画面左上）に，1 画素分の距離を 1 として指定する．色を指定するには，図形配置時に draw 命令の引数として Color（赤，緑，青）という書式で記述する．Psychlops はオブジェクト指向を活用し，どんな図形を指定しても位置や色の指定方法を変えずにすむよう設計している．

このように，図形の種類，位置，色を指定して配置することが一塊となって図形パーツを構成する．逆に，希望の図形パーツをサンプルコードから切り抜き，自分のプログラムに貼って色や位置を変えることで，論文用の刺激記述と対応させることができる．サンプルコードの中から自分の提示したい刺激に対応した図形描画ルーチンを切り出す際には，目的のパーツを最初から最後まで切り出せばよい．個別の図形は，種類の指定から始まって draw( )命令で終わるまでが一つの完結したパーツになる．

(a)

```
Psychlops::Rectangle patch(100,100);
                  図形の種類
patch.centering().shift(t, 0).draw(Color(0.0, 0.0, 0.0));
        図形の位置                  図形の色
```

(b)

```
Psychlops::Line patch(10,10, 200,200);
patch.centering().shift(t, 0).draw(Color(0.0, 0.0, 0.0));
```

図 3　（a）図形パーツの記述内容．図形の種類と，その種類でのパラメタ，位置，色を記述する．（b）図形の種類を長方形から直線に変更した場合でも，位置や色の記述方法は変わらない．

表 1　Psychlops で使うことができるさまざまな図形の種類と，その固有のパラメタ．「名前」の部分にはアルファベットで任意の名前を付け，括弧の中の各項目には任意の数字や数式を入れる．これらの図形パーツごとのサンプルコードは Psychlops のウェブサイトからダウンロードできる．

図形の種類	Psychlopsでの記法
長方形	Rectangle 名前(幅, 高さ);
楕円	Ellipse 名前(幅, 高さ);
線	Line 名前(始点x座標, 始点y座標, 終点x座標, 終点y座標)
多角形	Polygon 名前; 名前.append(頂点x座標, 頂点y座標) …(頂点の数だけ続ける)
文字	(1) Letters 名前(L"表示する文字");
	(2) Letters 名前(L"表示する文字", Font(L"フォント名"), 文字サイズ);
画像	(1) Image 名前("ファイル名"); （自然画像の場合）
	(2) Image 名前(幅, 高さ); （縞図形などの場合）

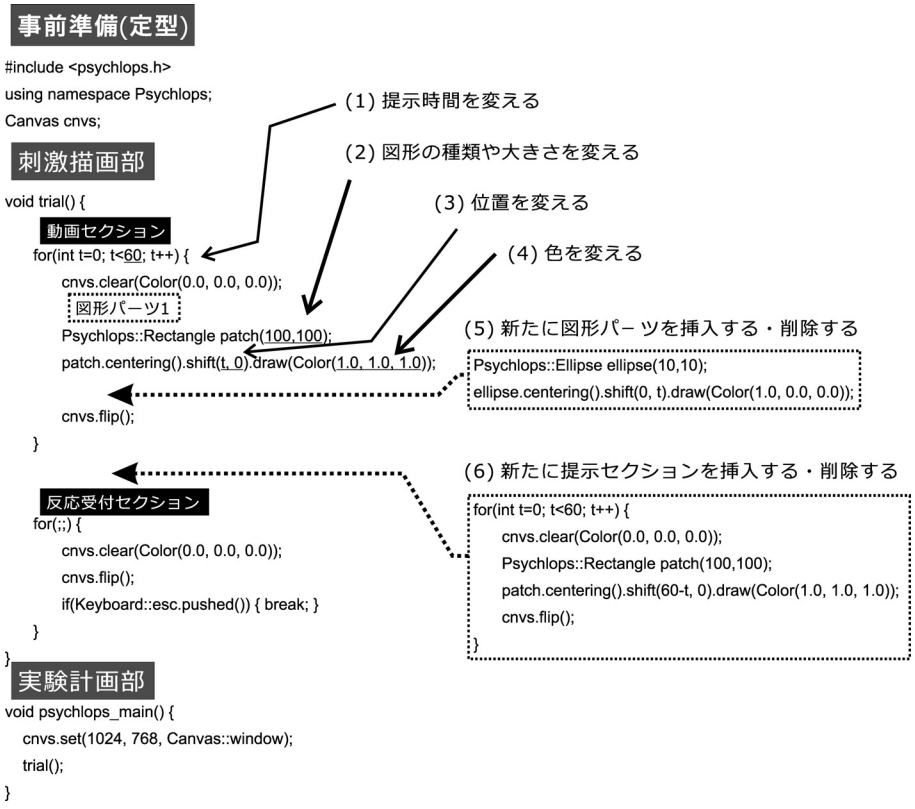


図 4　刺激描画部で実験に合わせて変える部位．図形の種類や大きさ，位置，色，セクションの提示時間や背景色，セクション中に含まれる図形パーツの挿入・削除，セクションの挿入・削除を行って自分の実験を定義する．

パーツ単位で抜き出して自分の実験プログラムのセクションに組み込めば図形を移植することができる．個々のセクションは，画面の背景をclear()で決めることから始め，そのコマにある個々の図形パーツすべてを記述し，flip()命令で締めて一つのパーツとして完結する．そのセクションと同じものを再現したい場合は，clear()命令からflip()命令までを囲む繰り返し文を探して抜き出せばよい．

3.3 図形描画部のカスタマイズできるポイント

図形描画部では，時間ごとに動画提示，反応受付などいくつかのセクションに分け，個々のセクションでどの程度の時間どんな図形を提示するかというパーツを構成することを説明した．言い換えると，個々の実験に合わせて書き換える内容はそれら 6 つの要素に分類できる（図 4）．個々の図形の種類，位置，大きさを変えたり，セクションに図形を挿入・削除したり，セクションの提示コマ数を変えて提示時間を変えたり，新たにセクションを挿入したりできる．

3.4 実験計画

ここまでで，図形を定義するパーツを組み合わせてセクションとなるパーツを構成し，セクションパーツを組み合わせて刺激描画パーツを構成する方法までを説明した．この刺激を用いて実験を行うには，試行を何度か繰り返しながら，物理刺激の条件（独立変数）を決まったやり方で変え，被験者の反応（従属変数）を取得する．これが後半の実験計画パーツである．実験計画は，まず独立変数と従属変数が何か定義するセクション，各試行で反応を取得するセクション，恒常法や上下法など独立変数の操作方法を決めるセクションの大きく 3 つに分かれる．

実験計画の第一のセクションとして，刺激ご

とに操作される独立変数と従属変数を抽出し，それについては刺激の記述前に定義する．この部分は基本的に実験ごとに異なるが，独立変数と従属変数の名前を列挙して定義する（図 5）．ここで定義した独立変数は刺激描画部で用いることができるが，その方法は後述する．

次に，反応取得セクションについて説明する．一般的に，被験者の反応は試行ごとに計測する．そのため，コンピュータの実験プログラムでは，反応取得セクションは個々の試行について書く刺激描画部の最後に追加される（図 5「反応取得セクション」）．反応取得セクションは，被験者に反応を促す視覚刺激を定義する部分に加え，被験者からの回答を取得するパーツから構成される．視覚刺激部分の構成は本章の前半で説明したとおりで，ここでは Psychlops における回答取得パーツの書き方について説明する．回答取得パーツでは「被験者からのキー押し回答があれば回答結果を従属変数として記録し，セクションを終了する」という手続きを記述する．回答に使用するキーを変えたり，変数列挙パーツで名前を挙げた従属変数に結果を記録する．強制二肢選択法のように複数のキーによって回答を受け付けるには，キーの数だけ回答取得パーツを並べる．一般的には反応取得セクションは被験者からの反応があるまで無制限に継続する．そのため，for 文丸括弧内での表示コマ数の指定を消去してある．そのために for(;;) としてあるが，反応取得を無制限に待つためにはこれも一字一句守って書く必要がある．

最後に，恒常法や上下法などの独立変数操作セクションを追加する．このセクションの内容は独立変数の操作法によって大きく異なり，パーツごとに分解して組み合わせるといった仕組みにはあまり適していない．そのためこのセクションに関しては，Psychlops では恒常法や上下法といった方法ごとにほとんどすべてコピー＆ペーストで完結するサンプルコードを用意しており，あらかじめ定義した独立変数と従属変数をそれに当てはめれば自動的に実験が遂行されるよう設計している．この方法については別途 Psychlops のウェブサイトに説明がある．

```
#include <psychlops.h>
using namespace Psychlops;
Canvas cnvs;
変数列挙パーツ
double velocity = 1.0, result = 0.0;   独立変数・従属変数に適当な名前を付けて列挙する
刺激描画部
void trial() {
    for(int t=0; t<60; t++) {
        cnvs.clear(Color(0.0, 0.0, 0.0));
        Psychlops::Rectangle patch(100,100);
        patch.centering().shift(t*velocity, 0).draw(Color(1.0, 1.0, 1.0));
        cnvs.flip();          独立変数に応じた位置設定をする
    }
    反応取得セクション
    for(;;) {
        cnvs.clear(Color(0.0, 0.0, 0.0));
        cnvs.flip();
        if(Keyboard::left.pushed()) { result = 1; break; }
        if(Keyboard::right.pushed()) { result = 2; break; }
    }      反応キーを指定      従属変数に結果を記録
}
実験計画部
void psychlops_main() {
  cnvs.set(1024, 768, Canvas::window);
  trial();
}
```

図 5　実験計画の定義方法．独立変数と従属変数を刺激描画する前に列挙し，刺激を独立変数に応じて変えるようにする．また，反応取得では，キーボードの反応があったら従属変数に結果を記録する定義を行う．この例では，左キーが押されたら従属変数 result に 1 を，右キーが押されたら result に 2 を記録するよう定義している．反応取得セクションの clear 命令と flip 命令の間に図形パーツを組み込むこともできる．

4. ありがちなミス

プログラムには特有の文法があり，マニュアルのとおり書いたつもりでも実行ができないことがある．たとえば，プログラムは融通がきかず，一字一句正しくなければ動かないことが多い．図形の種類や図形の名前，命令の名前については大文字小文字含めて正確に写す必要があり，セクションを区切る波括弧や関数に付随する丸括弧も正しい位置に正しい数だけ配置しなければならない．また，Psychlops では文（行）の終わりにセミコロン（;）を入れなければなら

ず，これを入れ忘れると動かないことがある．

5. おわりに

視覚実験には一般的な構造があり，その構造をプログラムで再現することで視覚実験を記述することが可能になる．本稿では，プログラム構造の基本形の一つを解説し，その基本形を構成するパーツの扱い方について説明した．この後は，サンプルプログラムからパーツをコピー&ペーストし，ごくわずかな修正をすれば，基本的な実験を施行することが可能である．本稿がプログラミングによる実験実施の一助になれば幸いである．

謝　辞　ライブラリをより使いやすいものにするため中嶋豊，池宮城匡両氏の協力に感謝する．