高松大学紀要，36．21～53

# 心理学実験プログラムの開発と教育場面への活用

西　浦　和　樹
伊　藤　克　浩

# A Development of computer program designed for introductory psychological experiment and application to introductory psychological education.

**Kazuki Nishiura**
**Katsuhiro Ito**

Abstract

The purpose of this study was to develop a computer program designed for psychological experiment “ Span of Attention ” and to apply it to an introductory psychological education．We attempted to describe it by a programming language “ Visual Basic 6 ” for Windows 2000，and to open the source code．

The features of the program are (1)realization of control of rapid display by DirectX7 SDK ；(2)use of high-resolution multimedia timer in order to conduct for time critical tasks；(3) programming for some experimental procedures：for example，randomization of a set of stimulus．

The findings of its application to class “ introductory psychological experiment ” describe how to troubleshoot frequently asked questions；(1)conflict problem with power management program；(2)usefulness of Windows API；(3)memory problem with AGP graphic card．

## １．はじめに

電子メール，WWWなどネットワークを介した情報技術の利用が進むにつれて，基本ソフトウェア（OS：Operating　System）の世代交代が進展しつつある。ここ数年，パーソナルコンピュータが必要とするOSは，Microsoft社の独占状態にあり，1995年のWindows 95発売以降，MS-DOSからWindowsへの移行がかなりの速度で進んできている。これに伴って，ストレスなくアプリケーションを動作させることのできるハードウェアへの入れ替えも進展しつつある。Windowsマシンの普及とともに，Windowsをベースとしたプログラム

開発も比較的容易になってきている。

しかしながら，2000年度日本心理学会における知覚・認知・記憶領域における学会発表資料を見る限りにおいて，AVタキストスコープ，パーソナルコンピュータを利用した心理学実験が実施されているにもかかわらず，依然として，Windows上での心理学実験はほとんど行われていない。この最大の理由として，学会員のほとんどがコンピュータ以外を専門とする心理学者であることが挙げられる。

以下では，このような状況を打破すべく，Windows上での心理学実験プログラム開発のための指針を整理しておく。

## ２．心理学実験プログラム開発にあたって

ここでは，心理学教育場面への適用を踏まえて，心理学実験プログラム開発のための指針を明らかにしておく。

まず，心理学教育場面への適用を考えると，大学で心理学実験を受講するユーザはワープロ，メール，WWWなどの比較的利用頻度の高いアプリケーションを使うものの，毎日利用することが少ない初心者ユーザであることが想定される（西浦・中條，2000）。このようなユーザが実験者や被験者として実験プログラムを利用することを考えると，実験プログラムはユーザビリティに配慮した設計でなければならない。

このようなユーザの視点に立つと，利用者が多いWindowsを利用した心理学実験プログラムは，次のような長所と短所を併せ持つ。

**長所**

(1) インターフェースが普段から使い慣れているWindowsで統一されており，ユーザにとって使い勝手がよい。

(2) Windowsマシンは普及しているので，手軽に実験を行える。

**短所**

(1) プログラムを記述する上で，画面サイズやウィンドウの大きさなどを記述しなければならず，プログラムが煩雑になる。

(2) Windowsのシステムイベントやその他の処理に対処しなければならず，ハードウェアに関する新しい知識が必要になる。

上述した短所は，主にWindows上での心理学実験プログラムの作成するプログラマの悩みである。このような短所を知らずして，一度Windowsプログラミングに挑戦してみると，

その煩雑さに気づくことになる。例えば，心理学実験では，高度な画面制御，タイマ制御，画像メモリ制御が要求されるため，Visual C++からDirectXのようなWindows APIを利用することは，非常に困難を伴う作業となる。

ところが，Visual BasicやDirectXのバージョンアップに伴い，Visual Basic 6（VB6）からDirectXを制御することが可能になり，プログラミング環境が一変した。VB6は，Windows上でプログラム開発するためのプログラミング言語であり，Visual C++ほどの柔軟性はないが比較的容易にアプリケーションを開発することができる。

### DirectX**とは**

DirectXは，DirectX1.0が1995年の秋に公開されて以来，1999年にはDirectX7がリリースされ，そのバージョンが上がるにつれて，多機能になり，多くの周辺機器に対応するようになってきた。

DirectXの特徴は，ハードウェアに実装されたアクセラレーション機能（HAL：Hardware Abstraction Layer）と，ソフトウェアによって実装されたエミュレーション機能（HEL：Hardware Emulation Layer）によって，1つのプログラムがどのような環境でも動作するように設計されていることである。

本稿で紹介するDirectX7には，DirectX7ランタイムライブラリ（DirectXの機能をDLLに集約したライブラリ）とDirectX7 SDK（Software Development Kit，DirectXを使用したプログラムの開発に必要な材料や資料を詰め込んだもの）という２つのパッケージがある。単にプログラムを実行するだけであれば，DirectX7ランタイムライブラリで事足りる。しかし，VB上でDirectX7を制御しようとすると，DirectX7 SDKが必要になる。

また，DirectX7 SDKには，RetailとDebugの２つの種別がある。Retailは余分な情報が入っていないため，Debugよりも高速に動作する。DirectX7 SDKのインストール時にSDKの種別を選択しなければならない。ここではRetailを選択することとする。

### DirectX**7の機能**

DirectX7の機能には，点，線，文字，画像などの２次元の描画を行うDirectDraw，３次元の描画を行うDirect3D，音の再生を行うDirectSound，キーボード，マウス，ジョイスティックからの情報を取得するDirectInputがある。

心理学実験では，数msから数百msの範囲で，文字や画像を高速呈示する必要が生じる。

このような画面制御を実現するためには，Windowsが行っている多くの無駄な処理を無くし，心理学実験プログラムの処理だけを実行できるようにしなければならない。

このような実験プログラムは，DirectDrawのフルスクリーンモードを利用することで実現できる。このため，プログラム開発者は，文字や画像の高速描画の基礎となるDirect Drawのフルスクリーンモードを理解しなければならない。

## ３．本研究の目的

本研究は，高度な画面制御を必要とする心理学実験プログラムの開発を試み，教育場面で活用することを目的とする。ここでは，VBからDirectXを利用することで，どのようにして画面制御を実現するのかを例示するため，心理学実験「注意の範囲」で利用した実験プログラムの解説を試みる。また，実験プログラムの活用事例を示す。

## ４．実験プログラムの開発

本実験プログラムは，心理学のための実験マニュアル（利島・生和，1993）に従って，心理学基礎実験「注意の範囲」を実施するために開発した。ここでは，実験の概要とプログラム開発手順を示す。

### 実験の概要

注意の範囲（span of attention）は，知覚の範囲（span of perception）と呼ばれることもあり，人間が一目で見えるのはどの程度であるのかを確認するための実験である。注意の範囲実験の特徴は，全体報告法と部分報告法という実験手続きを用いたところにある。

(1) 全体報告法（whole report method）

全体報告法では，被験者に高速に出現した呈示文字全体の報告を求める。実験の手続きは，①凝視点の呈示，②刺激の瞬間呈示，③信号音の呈示，④ブランクの呈示，⑤反応の記録であり，Figure 1に示す通りである。

この実験の結果，注意の範囲は‘７’を超えないという結論が得られている。しかし，被験者は，報告できた文字よりもたくさんの文字が見えていたこと，また報告している間に文字を忘れてしまうと内省報告した。これらの点を考慮し，被験者が瞬間にどの程度の文字が見えていたのかを調べるために，Sperling（1960）は部分報告法を開発した。

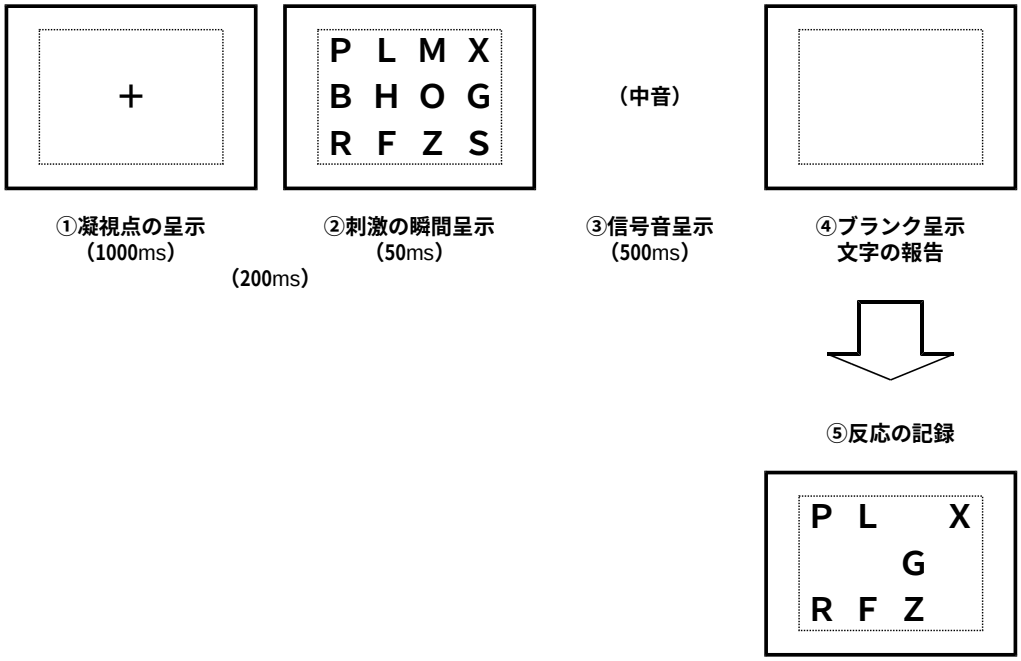


Figure 1　全体報告法における実験の流れ

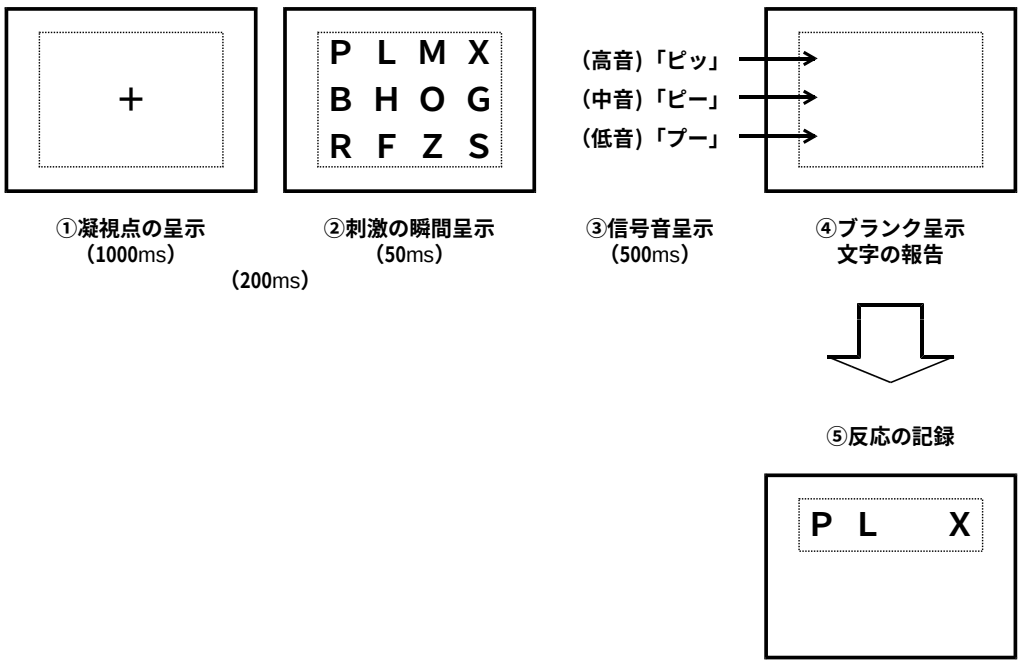


Figure 2　部分報告法における実験の流れ

(2) 部分報告法（partial report method）

この部分報告法では，報告の限界（あるいは短期記憶の限界）を測定するのではなく，実際に知覚されている（利用可能な）視覚情報を調べるための工夫が見られる。実験の手続きは，①凝視点の呈示，②刺激の瞬間呈示，③信号音の呈示，④ブランクの呈示，⑤反応の記録の順に実施される。ここでの被験者の課題は，高速に出現した呈示文字の中から信号音で指示された行の文字を報告することである。

この実験の結果，部分報告法で測定された利用可能な文字数は，全体報告法で報告された文字数より多かったことがわかっている。

**実験プログラム開発手順**

上述の実験手続きをVBにて再現する際に，VBからDirectXを利用するための設定が必要になる。VBの設定は，「プロジェクト」メニューから，「参照設定」を選択し，「参照可能なライブラリファイル」リストボックスから「DirectX7 for Visual Basic Type Library」をチェックしなければならない。詳しくは，Visual Basic DirectXプログラミング（山崎，1999）の35ページに記載されている。

実験プログラムは，プロジェクトSpanOfAttentionとし，このプロジェクトはfrmForm1.frm, frmSplash.frm，frmDialog.frmから構成されている。frmSplash.frmは実験プログラム開始直後に出現するフォームであり，Figure 3に示す開発画面となる。frmDialog.frmは被験者名，年齢，性別，実験条件の入力用フォームであり，Figure 4に示す開発画面となる。本実験のプログラムリストfrmForm1.frm，frmSplash.frm，frmDialog.frmは，本稿末尾のリスト１，リスト２，リスト３に示した。

以下では，実験のメインプログラムであるリスト１について解説する。

リスト１は，(1) DirectX宣言部，(2) 呈示する刺激セットのランダム化，(3) 被験者データの記録，(4) DirectXの初期化，(5) 刺激の呈示，(6) 呈示した刺激の記録，(7) DirectXの解放，の順に記述されている。

**(1) DirectX宣言部**

DirectXの宣言部では，画像呈示のためにどのような方法（実際には，ビデオメモリ上に作られるオブジェクトの利用方法）を使うのかを宣言する。この実験プログラムでは，ビデオメモリ上にプライマリサーフェスobjDDSPrimaryとセカンダリサーフェスobjDDSSecondaryを使用している。

プライマリサーフェスは，実際にディスプレイに表示される表画面であり，ビデオメモリ上に確保されるものである。また，セカンダリサーフェスは，ディスプレイ表示中に刺激の描画を行うことのできる裏画面であり，同様にビデオメモリ上に確保される。サーフェスのことをバッファと呼ぶこともあるので，この２つのサーフェスを利用する手法は，ダブルバッファリングと呼ばれている。

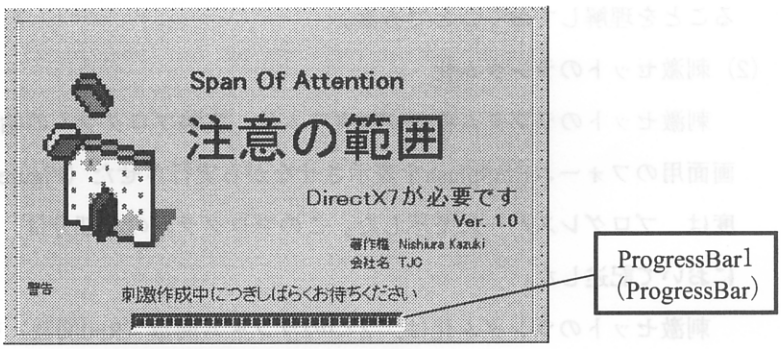


Figure 3　フォームfrmSplash.frmのデザイン。
デザインの説明は，オブジェクト名（オブジェクト）を指す。
プログラムリストはリスト２に示した。

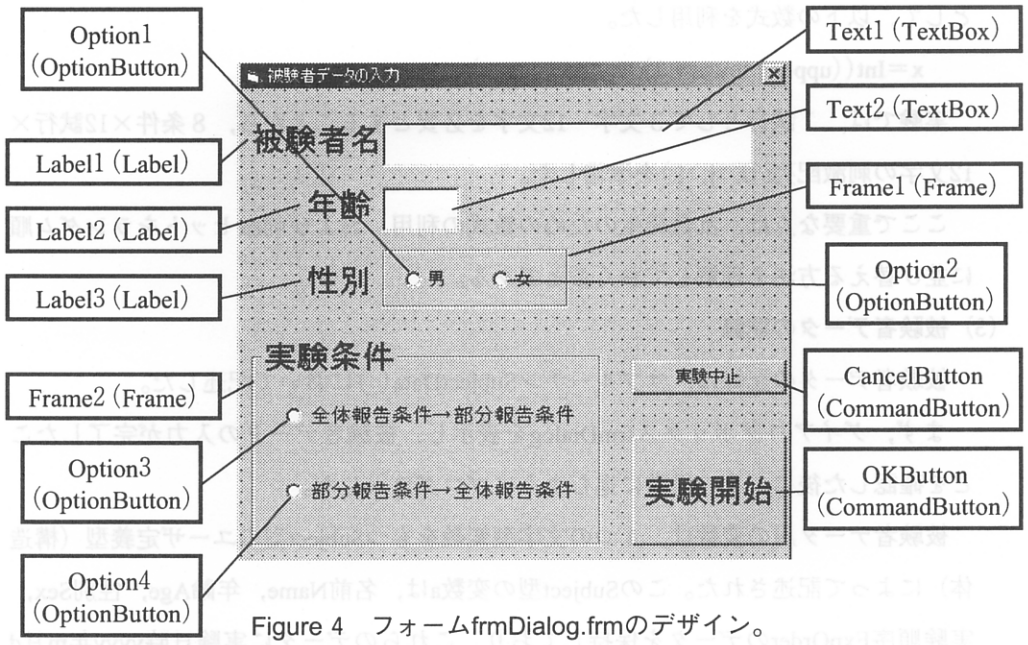


Figure 4　フォームfrmDialog.frmのデザイン。
デザインの説明は，オブジェクト名（オブジェクト）を指す。
プログラムリストはリスト３に示した。

そして，セカンダリサーフェスに刺激を描画し，その後，セカンダリサーフェスとプライマリサーフェスを瞬時に切り替えることで，ディスプレイ表示時のちらつきをなくし，刺激系列の高速呈示を可能にする。このような２つのサーフェスの切り替え処理は，フリップと呼ばれている。

ここで重要な点は，高速呈示を必要とする実験では，２つ（あるいはそれ以上）のサーフェスを確保し，これらのサーフェスをフリップすることで高速呈示が可能になることを理解しておくことである。

**(2) 刺激セットのランダム化**

刺激セットのランダム化のプログラムは，実験プログラムの始まりを告げる見出し画面用のフォームfrmSplashを表示させながら実行させた（Figure 1）。その実行の進度は，プログレスバーにて示した。このプログラムの大部分は，サブルーチンShuffleにおいて記述した。

刺激セットのランダム化は，２つのランダム関数（Rnd関数）を用い，50msの間，26枚並べた刺激カードの任意のx番目と任意のxx番目を入れ替えることで行った。ちょうど，トランプのカードをシャッフルする要領で行うこととなる。

ここで，最小値lowerから最大値upperの間で変化する数xを発生させるための手法として，以下の数式を利用した。

x＝Int((upper－lower＋1) * Rnd＋lower)

実験では，１試行として３文字～12文字を必要とすることから，８条件×12試行×12文字の刺激配列s(i，j，k)を準備した。

ここで重要な点は，乱数発生のための数式の利用，および刺激セットをランダム順に並び替える方法を理解しておくことである。

**(3) 被験者データの記録**

被験者データの記録は，サブルーチンSubjectData( )において記述した。

まず，ダイアログボックスfrmDialogを表示し，被験者データの入力が完了したことを確認した後に，次の段階に進むようにプログラムした。

被験者データ用の変数は，４つの文字型変数をもつSubject型のユーザ定義型（構造体）によって記述された。このSubject型の変数aは，名前Name，年齢Age，性別Sex，実験順序ExpOrderのデータを保持しており，これらのデータに実験日時yyyy年m月d日hh時nn分ss秒を加えて，ファイル名ExpData.txtに保存された。

また，被験者名，年齢，性別，実験条件の入力がない場合は，エラーメッセージを表示し，入力が行われるまで先に進まないようにプログラムした。

### (4) DirectXの初期化

ここでは，実際に刺激の呈示に必要なDirectX初期化，とりわけDirect Draw初期化の手続きについて解説する。

DirectXの初期化は，DirectDrawオブジェクトの作成，DirectDraw協調レベルの設定，ディスプレイモードの変更，プライマリサーフェスの作成，セカンダリサーフェスの作成という手順で行う。この手順は，DirectXを利用するときの約束事として理解しておくとよい。

DirectDrawオブジェクトの設定では，DirectDrawCreateメソッドを呼び出し，宣言部で宣言したobjDDというビデオメモリ上の入れ物に格納することとなる。

DirectDraw協調レベルの設定では，実験プログラムがフルスクリーンで動作するかウィンドウモードで動作するかを選択する。SetCooperativeLevelメソッドの引数は，排他モードDDSCL_EXCLUSIVEによって，ディスプレイモードの変更やパレットの変更などを可能とし，フルスクリーンモードDDSCL_FULLSCREENと組み合わせて利用することとなる。

ディスプレイモードの設定では，実験で利用する画面の解像度を指定することができる。SetDisplayModeメソッドの引数は，横640ドット×縦480ドット，８ビット256色，100Hz，通常モードに設定した。

プライマリサーフェスの作成手順は，ユーザ定義型のDDSURFACEDESC2として変数ddsdを作成し，プライマリサーフェス情報の設定を行った後に，プライマリサーフェスの取得を行うこととなる。

プライマリサーフェス情報の設定では，サーフェスの２つの能力を設定するため，.lFlagsにおいて，DDSD_CAPSとDDSD_BACKBUFFERCOUNTのOR値を指定する。つまり，.ddsCaps.lCapsにおける引数について，DDSCAPS_PRIMARYSURFACEは，このサーフェスがプライマリサーフェスであり，DDSCAPS_FLIPは，セカンダリサーフェスとフリップを行うことができ，DDSCAPS_COMPLEXは，複数のサーフェスを使うという意味がある。設定終了後，CreateSurfaceメソッドを用いてプライマリサーフェスの取得を行う。

セカンダリサーフェス情報の設定では，バックバッファとして利用するため，

ddcaps.lCapsにてDDSCAPS_BACKBUFFERを指定している。設定終了後，GetAttached Surfaceメソッドを用いてプライマリサーフェスに関連付けられたセカンダリサーフェスの取得を行う。

これらサーフェス情報の設定は，非常に複雑であるため，ダブルバッファリングによる心理学実験は，とりあえずこのような設定になるものと理解しておく方がよいだろう。

**(5) 刺激の呈示**

刺激呈示の手続きは，それぞれのプログラムが煩雑になるためサブルーチン化して記述した。

PrConditionでは，刺激の呈示列数の設定，および音の鳴り分けによる報告行の設定を行った。全体報告条件では，中音のみ利用するが，部分報告条件では，高音，中音，低音の中からランダム順に音が鳴ることで，報告行を被験者に知らせるようになっている。このため，各試行ごとに，音の鳴り分けがランダムな順序で行えるように，１～３の数字を配列Soundに格納した。

また，全体報告条件，部分報告条件のそれぞれにおいて，各ブロックごとに呈示列数が１～４列となるよう，１～４の数字を配列Columnに格納した。

PrBlockでは，各ブロックで行われる実験条件の保存と呈示を行った。これら実験条件の呈示が各ブロックの最初に行えるように，反復処理を行った。

また試行ごとに，キーを押すと実験開始PushKey，凝視点の描画Fixation，ブランクの呈示Blank，文字刺激の呈示Update，ブランクの呈示Blank，Beep音の呈示Sound，刺激データの保存DataSaveの順にサブルーチンを呼び出すようにプログラムを記述した。

凝視点や文字の描画は，DrawTextメソッドを使い，一度セカンダリサーフェスに書き込んだ後に，Flipメソッドを使ってセカンダリサーフェスの内容をプライマリサーフェスに転送した。Flipメソッドの引数Nothingは次に位置付けられたセカンダリサーフェスとのフリップを行い，またDDFLIP_WAITはハードウェアの描画終了を待ってからフリップを行うことを意味する。この記述は，垂直同期を取るという点で，心理学実験プログラムでは重要な意味を持つ。

Soundでは，Beep APIを利用して信号音の決定を行っている。Beepの引数は，高音の場合2500Hz，中音の場合650Hz，低音の場合250Hzをそれぞれ500ms間鳴り続ける

ように指定した。こうすることで，本実験における信号音による音の鳴り分けを可能にする。

**(6) 呈示した刺激の記録**

刺激の記録は，サブルーチンDataSaveにおいてプログラムした。

全体報告条件では，呈示したすべての刺激を記録したが，部分報告条件では，呈示した刺激の中から報告を指示した行の刺激のみを記録した。

実験終了時に，実験の終了時間yyyy年m月d日hh時nn分ss秒を記録した。

**(7) DirectXの解放**

実験プログラムが終了し，Windowsに戻る際に，ディスプレイの状態を元に戻さなければならない。また，DirectXが作成したオブジェクトを解放する必要がある。実際のプログラムでは，ディスプレイモードの復元，協調レベルの復元，オブジェクトの解放の順に行う。

これらの手続きは，DirectXを利用した後の決まり事として，終了処理には必要なものだと理解しておくとよい。

## 5．教育場面への活用

実験プログラムの開発終了後，教育場面への活用を試みた。授業名は「心理学実験」であり，(1) 授業の概要を説明し，(2) 心理学実験を行い，(3) そのデータをもとにレポートを作成するというものである。

授業の実施方法は，以下に示す通りであった。

(1) 授業の概要説明

①コンピュータを起動する。

②インストラクターと一緒に実験の練習を行う。

③課題の実施順序を決定する。

はじめに，「注意の範囲」実験の概要を理解させるために，実験のデモンストレーションを行った。実験のデモンストレーションでは，全体報告法における被験者は呈示された文字すべてを報告しなければならないこと，さらに部分報告法における被験者は信号音を聞き分け，指示された行の文字を報告しなければならないことを確認した。なお，すべての受講生が部分報告法における信号音の聞き分けを理解するまでデモンストレーションを行った。

受講生８名が実験に参加した。受講生は，全体報告法から部分報告法を実施する群，部分報告法から全体報告法を実施する群，というように課題の実施順序を二通り設定しカウンターバランスをとったため，各群４名ずつに分かれて実験を行うこととなった。また，すべての受講生が実験者と被験者を体験できるように配慮した。

(2) 心理学実験「注意の範囲」実施

①実験者は被験者をコンピュータの前に着席させる。

②実験者は被験者の隣に着席する。

③実験者は実験プログラムを起動させ，被験者名，課題の実施順序などを入力する。

④実験を実施する。

実験場所は，テーブル上にパーソナルコンピュータを設置し，被験者がゆとりを持って実験に参加できるよう配慮した。

実験の教示は，被験者が全体報告課題と部分報告課題の二通りの課題があることを理解させるため，以下のように行った。

**教示**　これから，コンピュータの画面に瞬間的に呈示される文字を報告するという課題を行ってもらいます。呈示される文字はアルファベットの大文字です。文字が呈示される前には，画面の中央に「＋」の形をした凝視点が呈示されますので，その凝視点を見ていてください。凝視点が出て，消え，文字刺激が瞬間的に出て，信号音が鳴ります。信号音が鳴り次第，呈示された文字が何だったかを実験者に報告してください。実験者が記録したら確認の合図をしますので，スペースキーを押して次の試行に移ってください。

また，報告の仕方には２通りあります。

全体報告課題：呈示された文字すべての中から憶えているものを報告する課題。

部分報告課題：コンピュータの信号音によって指示された行に含まれる文字だけを報告する課題。

なお，最初の２試行は練習，その後の10試行は本試行です。

(3) レポート作成

①被験者すべてのデータを集計し，結果の整理を行う。

②受講生は各自でレポートを作成する。

まず，実験で得られたデータに基づいて受講生に結果の整理を行わせた。次に，実験レポートの作成にあたり，目的，方法，結果，考察，引用文献の記述の仕方について解説し，実験で得られたデータについては補助資料として添付するよう受講生に求めた。なお，レポートの提出期限は１週間，再提出に１週間の期間を設定した。

## ６．今後の課題

以下では，現在のところ把握している実験プログラム開発及び利用上の注意点について述べておく。

まず，実験プログラムの活用場面において，Windowsの電源管理プログラムなどの常駐プログラムとの相性問題がわかっている。本実験プログラムを作成中に，原因不明のエラーが発生したこともあったが，常駐プログラムを終了させることでエラーが大幅に減少した。このことからも，本実験プログラムは，可能な限り常駐プログラムを終了させて利用することを推奨する。

第二に，APIについて：APIに関する知識が不可欠である。今回のプログラムでは，Windows98とWindows2000で動作の異なるAPI（Beep API）を使用した。このため，本実験プログラムは，Windows98において動作を保証していない。これは，Beep音の高さを制御する必要があったためである。単に実験の合図のためにBeep音を制御したいというのであれば，MessageBeep　APIを利用することでWindows上での互換性を確保することができよう。

また，時間の計測に用いたタイマ関数の精度の問題がある。今回は，Windows標準APIのGetTickCount関数ではなく，ライブラリの追加が必要なtimeGetTime関数を使用した。この理由は，timeGetTime関数がマルチメディアの制御を目的としているのでGetTickCount関数より精度が高いからである。心理学実験では時間の精度の高さが要求されるため，timeGetTime関数を使用することを強く推奨する。

第三に，グラフィックカードについて，AGPバスを使ったビデオカードを使うことを推奨する。心理学実験では，たくさんの画像情報を連続して呈示することがある。このような場合，たくさんの画像情報をビデオメモリ上に格納する必要が生じる。DirectXは，AGPで足りないメモリ分をメインメモリから拝借することができるように設定可能である。ただし，ビデオカードによってはかなりスピードが落ちる場合もあるので，注意が必要である。

以上のような点を念頭に置きつつ，本論文で紹介した実験プログラムを多くの実験場面に活用して頂ければと期待している。例えば，心理学実験では，画像ファイルの呈示，および反応時間の計測といったプログラムは必要不可欠になってくる。これらのプログラミングは，DirectDrawを活用して，オフスクリーンサーフェスへの画像ファイルの読み込み，およびDirectInputを活用して，キーボードバッファでのイベント発生時刻（システム時間）の取得により可能となる。機会があれば，改めて公開したい。

最後に，DirectXを利用した心理学実験プログラムが少ない現状では，これらのプログラミング手法を公開し，蓄積していくことが必要であろう。そうすることで，多くの教育場面での活用が期待されるばかりか，プログラミング手法の共有も可能となり，多くの成果が生み出されることが期待できる。

以下，実験プログラム開発に役立つ参考文献を挙げておく。開発の一助になればと思う。

**参考文献**


1．Visual Basic DirectXプログラミング　山崎由喜憲　ソフトバンクパブリッシング　1999年　3,200円　ISBN4-7973-1054-5

DirectXの解説書。DirectXインストールから開発までをわかりやすく解説している。本稿の刺激呈示プログラムの開発に大いに役立った。巻末にDirectX7主要メソッド一覧が掲載されているので，メソッドの引数の意味を知りたいときに参考になる。

2．Visual Basic 6.0入門：基礎編　笠原一浩・山本美孝　ソフトバンクパブリッシング　1998年　2,500円　ISBN4-7973-0734-X

Visual Basic 6.0の入門書。Visual Basicでできることを一通り解説している。本稿では，被験者データの入力フォームの開発に役立った。Windowsで表示されるフォームのデザインとその動作などVisual Basicの基本事項を学ぶのに参考になる。

3．Visual Basic コースウェア⑤　Visual Basic 6.0中級テクニック編　河西朝雄　技術評論社　1999年　2,280円　ISBN4-7741-0806-5

Visual Basic 6.0の入門書。上述のVisual Basic 6.0入門の解説を補う形で使用されたい。また，Windows APIとAPIビューアの使い方を解説しているので，標準APIにはないtimeGetTime関数やBeep関数を使用するときに参考になる。

4．Visual Basic コースウェア⑥　Visual Basic 6.0上級編　河西朝雄　技術評論社　1999年　2,180円　ISBN4-7741-0807-3

Visual Basic 6.0の入門書。上述のVisual Basic 6.0中級テクニック編の解説を補う形で使用されたい。VBからWindowsへの制御を渡すDoEventsステートメントの使い方を解説しているので，ループ処理を用いるときに参考になる。

５．DirectX7 For Visual Basic：はじめるゲームプログラミング　藤田伸二　翔泳社　1999年　2,800円　ISBN4-88135-811-1

DirectXの解説書。Visual Basic DirectXプログラミングを補う形で使用されたい。また，巻末に掲載されているテキストの表示とフォントの指定のメソッド解説が参考になる。

## 引用文献


西浦和樹・中條和光　2000　情報活用能力育成のためのカリキュラム開発　広島大学教育学部紀要第一部（心理学），**48**，43－51.

Sperling，G．1960 The information available in brief visual presentations．Psychological Monographs：General and Applied，**74**，１－29.

利島保・生和秀敏（編著）　1993　心理学のための実験マニュアル：入門から基礎・発展へ　北大路書房

**《リスト１（frmForm1.frm）》「注意の範囲」のプログラムリスト**

```
Option Explicit

Private Declare Function timeGetTime Lib "winmm.dll" ()As Long
Private Declare Function Beep Lib "kernel32" (ByVal dwFreq As Long, ByVal dwDuration As Long)As Long
Private Declare Function ShowCursor Lib "user32" (ByVal bShow As Long)As Long

'これがすべての始まり、DirectX7オブジェクトを作成します
Dim objDX As New DirectX7

'DirectDrawオブジェクト
Dim objDD As DirectDraw7

'プライマリサーフェス
Dim objDDSPrimary As DirectDrawSurface7

'セカンダリサーフェス
Dim objDDSSecondary As DirectDrawSurface7

'被験者データを構造体で記述
Private Type Subject
    Name As String
    Age As Single
    Sex As String
    ExpOrder As String
End Type
Dim a As Subject

'呈示条件を構造体で記述
Private Type ExpCondition
    Sound As Integer
    Column As Integer
End Type
Dim b(1 To 8, 1 To 12)As ExpCondition

Dim block As Integer, Trial As Integer
Dim upper As Integer, lower As Integer

Dim Stimulus(1 To 26) As String '刺激セットＡからＺを格納用の配列
Dim s(1 To 8, 1 To 12, 1 To 12)As String '刺激呈示格納s(４条件，12試行，12文字）
Public bExitLoop As Boolean
Public bExitLoopUpdate As Boolean
```

```
Public bExitLoopWait As Boolean

Dim i As Integer，j As Integer，k As Integer 'ループ用変数

'キーを押したときのループ処理用変数
Private Sub Form_KeyDown(KeyCode As Integer，Shift As Integer)
    bExitLoopWait = False
End Sub

Private Sub Form_Load()
    'タイトル表示
    frmSplash.Show

    'ＡからＺまでの刺激をシャッフルする
    Call Shuffle

    'タイトル消去
    frmSplash.Visible = False

    '被験者データの入力＆保存
    Call SubjectData

' ═══════════════════════════════════
'   DirectXの初期化
' ═══════════════════════════════════
'DirectDrawオブジェクトを作成
    Set objDD = objDX.DirectDrawCreate("")

'DirectDraw協調レベルを設定
    Call objDD.SetCooperativeLevel(Me.hWnd，DDSCL_EXCLUSIVE Or DDSCL_FULLSCREEN)

'ディスプレイモードを変更（リフレッシュレート100Hzに固定）
    Call objDD.SetDisplayMode(640，480，8，100，DDSDM_DEFAULT)

'プライマリサーフェスを作成
    Dim ddsd As DDSURFACEDESC2

    'プライマリサーフェスを作成するためのサーフェス情報を設定
    With ddsd
        .lFlags = DDSD_CAPS Or DDSD_BACKBUFFERCOUNT
        .ddsCaps.lCaps = DDSCAPS_PRIMARYSURFACE Or DDSCAPS_FLIP Or DDSCAPS_COMPLEX
        .lBackBufferCount = 1
    End With
```

```
    'プライマリサーフェスを取得
    Set objDDSPrimary = objDD.CreateSurface(ddsd)

'セカンダリサーフェスを作成
    Dim ddcaps As DDSCAPS2

    'セカンダリサーフェスを作成するためのサーフェス情報を設定
    ddcaps.lCaps = DDSCAPS_BACKBUFFER Or DDSCAPS_COMPLEX


    'セカンダリサーフェスを作成
        Set objDDSSecondary = objDDSPrimary.GetAttachedSurface(ddcaps)


' ═══════════════════════════════════
'   ここから刺激呈示アルゴリズム
' ═══════════════════════════════════

'各種条件の設定
ShowCursor 0      'カーソルを消す場合０

block = 8     '全８ブロック
Trial = 12   '全12試行
Call PrCondition      '呈示列数、音、
Call FileOpen     '実行パスにファイルオープン

For i = 1 To block

    'データセーブ
    If (frmDialog.Option3 = True And i <= 4) Or (frmDialog.Option4 = True And i >= 5)Then
        Write #1, "全体報告条件　第" & i & "ブロック", "３行" & b(i, 1).Column & "列"
    Else
        Write #1, "部分報告条件　第" & i & "ブロック", "３行" & b(i, 1).Column & "列"
    End If

    '実験条件の呈示
    Call PrBlock(Int(i))
    Wait (5000)

    For j = 1 To Trial

        '何かキーを押すとスタート
        Call PushKey
```

```
            '凝視点の描画
            Call Fixation
            Wait (1000)

            'ブランク
            Call Blank
            Wait (200)

            '文字の描画
            Call Update(Int(i), Int(j), Int(b(i, j).Column))

            '文字の呈示時間
            Wait (50)

            'ブランク
            Call Blank

            'Beep音
            Call Sound(Int(i), Int(j))

            'データセーブ
            Call DataSave(Int(i), Int(j), Int(b(i, j).Column))
            DoEvents
            '報告時間
            Wait (3000)
        Next j
Next i

ShowCursor 1      'カーソルを出す

    '終了処理
    Write #1, Format(Now, "終了時間yyyy年m月d日hh時nn分ss秒")
    Close #1
    Unload Me
    End
End Sub

' ═══════════════════════════════════════
'   何かキーを押すとスタート
' ═══════════════════════════════════════
Private Sub PushKey()

    Dim rcRect As RECT
    With rcRect
```

```
        .Top = 0
        .Left = 0
        .Right = 0
        .Bottom = 0
    End With

    Call objDDSSecondary.BltColorFill(rcRect, 255)

    '描画フォント
    objDDSSecondary.SetFont Me.Font
    objDDSSecondary.SetForeColor (RGB(0, 0, 0))

    objDDSSecondary.DrawText 50, 230, "キーを押して下さい", False
    'サーフェスをフリップする
    Call objDDSPrimary.Flip(objDDSSecondary, DDFLIP_WAIT)
    bExitLoopWait = True

    'キーを押すと先に進む
    Do While bExitLoopWait
        DoEvents
    Loop

End Sub

' ═══════════════════════════════════
'  実験条件の描画
' ═══════════════════════════════════
Private Sub PrBlock(i As Integer)
    Dim rcRect As RECT
    With rcRect
        .Top = 0
        .Left = 0
        .Right = 0
        .Bottom = 0
    End With

    Call objDDSSecondary.BltColorFill(rcRect, 255)

    '描画フォント
    objDDSSecondary.SetFont Me.Font
    objDDSSecondary.SetForeColor RGB(0, 0, 0)

    If (frmDialog.Option3 = True And i <= 4) Or (frmDialog.Option4 = True And i >= 5)Then
        '全体報告条件のとき
```

```
        objDDSSecondary.DrawText 100，130，"第" & i & "ブロック開始"，False
        objDDSSecondary.DrawText 100，210，"全体報告条件"，False
        objDDSSecondary.DrawText 100，290，"３行" & b(i，1).Column & "列"，False
    Else
        '部分報告条件のとき
        objDDSSecondary.DrawText 100，130，"第" & i & "ブロック開始"，False
        objDDSSecondary.DrawText 100，210，"部分報告条件"，False
        objDDSSecondary.DrawText 100，290，"３行" & b(i，1).Column & "列"，False
    End If

    'サーフェスをフリップする
    Call objDDSPrimary.Flip(objDDSSecondary，DDFLIP_WAIT)

End Sub

' ═══════════════════════════════════
'  データ保存：ファイル（ExpData.txt）
' ═══════════════════════════════════
Private Sub DataSave(i As Integer，j As Integer，PrColumn As Integer)
    If（frmDialog.Option3 = True And i <= 4） Or（frmDialog.Option4 = True And i >= 5） Then
        '全体報告条件のとき
        Select Case PrColumn
            Case 1
                Write #1，s(i，j，1)，s(i，j，2)，s(i，j，3)
            Case 2
                Write #1，s(i，j，1)，s(i，j，2)，s(i，j，3)，s(i，j，4)，s(i，j，5)，s(i，j，6)
            Case 3
                Write #1, s(i, j, 1), s(i, j, 2), s(i, j, 3), s(i, j, 4), s(i, j, 5), s(i, j, 6), s(i, j, 7), s(i, j, 8), s(i, j, 9)
            Case 4
                Write #1, s(i, j, 1), s(i, j, 2), s(i, j, 3), s(i, j, 4), s(i, j, 5), s(i, j, 6), s(i, j, 7), s(i, j, 8), s(i, j, 9), s(i, j, 10), s(i, j, 11), s(i, j, 12)
        End Select
    Else
        '部分報告条件のとき
        Select Case PrColumn
            Case 1
                Call Save1(i，j) 'データ保存１
            Case 2
                Call Save2(i，j) 'データ保存２
            Case 3
                Call Save3(i，j) 'データ保存３
            Case 4
                Call Save4(i，j) 'データ保存４
        End Select
    End If
```

```
End Sub

' ═══════════════════════════════════════
'  データ保存１（ExpData.txt）
' ═══════════════════════════════════════
Private Sub Save1(i, j As Integer)
    If b(i, j).Sound = 1 Then
        Write #1, s(i, j, 1)
    End If
    If b(i, j).Sound = 2 Then
        Write #1, s(i, j, 2)
    End If
    If b(i, j).Sound = 3 Then
        Write #1, s(i, j, 3)
    End If
End Sub

' ═══════════════════════════════════════
'  データ保存２（ExpData.txt）
' ═══════════════════════════════════════
Private Sub Save2(i, j As Integer)
    If b(i, j).Sound = 1 Then
        Write #1, s(i, j, 1), s(i, j, 2)
    End If
    If b(i, j).Sound = 2 Then
        Write #1, s(i, j, 3), s(i, j, 4)
    End If
    If b(i, j).Sound = 3 Then
        Write #1, s(i, j, 5), s(i, j, 6)
    End If
End Sub

' ═══════════════════════════════════════
'  データ保存３（ExpData.txt）
' ═══════════════════════════════════════
Private Sub Save3(i, j As Integer)
    If b(i, j).Sound = 1 Then
        Write #1, s(i, j, 1), s(i, j, 2), s(i, j, 3)
    End If
    If b(i, j).Sound = 2 Then
        Write #1, s(i, j, 4), s(i, j, 5), s(i, j, 6)
    End If
    If b(i, j).Sound = 3 Then
        Write #1, s(i, j, 7), s(i, j, 8), s(i, j, 9)
```

```
    End If
End Sub

' ═══════════════════════════════════════
'  データ保存４（ExpData.txt）
' ═══════════════════════════════════════
Private Sub Save4(i, j As Integer)
    If b(i, j).Sound = 1 Then
        Write #1, s(i, j, 1), s(i, j, 2), s(i, j, 3), s(i, j, 4)
    End If
    If b(i, j).Sound = 2 Then
        Write #1, s(i, j, 5), s(i, j, 6), s(i, j, 7), s(i, j, 8)
    End If
    If b(i, j).Sound = 3 Then
        Write #1, s(i, j, 9), s(i, j, 10), s(i, j, 11), s(i, j, 12)
    End If
End Sub

' ═══════════════════════════════════════
'  ファイル（ExpData.txt）オープン
' ═══════════════════════════════════════
Private Sub FileOpen()
    Dim fname As String
    If Right$(App.Path, 1) <> "¥" Then
        fname = App.Path & "¥ExpData.txt"
    Else
        fname = App.Path & "ExpData.txt"
    End If
    Open fname For Append As #1
End Sub

' ═══════════════════════════════════════
'  信号音の決定
' ═══════════════════════════════════════
Private Sub Sound(i, j As Integer)
    Select Case b(i, j).Sound
        Case 1
            Beep 2500, 500   '高音2500Hz 500ms
        Case 2
            Beep 650, 500    '中音650Hz   500ms
        Case 3
            Beep 250, 500    '低音250Hz   500ms
    End Select
```

```
End Sub

' ═══════════════════════════════════════
'  呈示条件の決定
' ═══════════════════════════════════════
Private Sub PrCondition()
        '報告条件　frmDialog.Option3 = True（全体→部分），False（部分→全体）
        '行 b(i,j).Sound = 1，2，3
        Randomize
        If frmDialog.Option3 = True Then
            For i = 1 To 4
                For j = 1 To 12
                    b(i, j).Sound = 2
                Next j
            Next i
            For i = 5 To 8
                For j = 1 To 12
                    b(i, j).Sound = Int((3－1＋1) * Rnd＋1)
                Next j
            Next i
        Else
            For i = 1 To 4
                For j = 1 To 12
                    b(i, j).Sound = Int((3－1＋1) * Rnd＋1)
                Next j
            Next i
            For i = 5 To 8
                For j = 1 To 12
                    b(i, j).Sound = 2
                Next j
            Next i
        End If

        '呈示列数　b(i,j).Column = 1，2，3，4
        Dim Num(1 To 8) As Integer
        Dim x As Integer: Dim x1 As Integer
        Dim xx As Integer: Dim xx1 As Integer
        Dim ShuffleTime As Long
        Dim Shuffle As Integer: Dim Shuffle1 As Integer
        Dim lLastTime As Long
        Dim lPastTime As Long
        Dim lNowTime As Long
```

```
        For i = 1 To 4
            Num(i) = i: Num(i + 4) = i
        Next i

        ShuffleTime = 100
        Randomize
        lLastTime = timeGetTime()
        lPastTime = 0
        Do While lPastTime < ShuffleTime
            lNowTime = timeGetTime()
            lPastTime = lNowTime－lLastTime

            '１～４ブロックのシャッフル
            x = Int((4－1＋1) * Rnd＋1)
            xx = Int((4－1＋1) * Rnd＋1)

            '５～８ブロックのシャッフル
            x1 = Int((8－5＋1) * Rnd＋5)
            xx1 = Int((8－5＋1) * Rnd＋5)

            Shuffle = Num(xx): Shuffle1 = Num(xx1)
            Num(xx) = Num(x): Num(xx1) = Num(x1)
            Num(x) = Shuffle: Num(x1) = Shuffle1
            'イベント処理
            DoEvents
        Loop
        For i = 1 To 4
            For j = 1 To 12
                b(i, j).Column = Num(i): b(i + 4, j).Column = Num(i + 4)
            Next j
        Next i


End Sub


' ═══════════════════════════════════
'  凝視点の描画
' ═══════════════════════════════════
Private Sub Fixation()
    Dim rcRect As RECT
    With rcRect
        .Top = 0
        .Left = 0
```

```
        .Right = 0
        .Bottom = 0
    End With

    'セカンダリサーフェスを白で塗りつぶす
    Call objDDSSecondary.BltColorFill(rcRect, 255)

    '描画フォント
    objDDSSecondary.SetFont Me.Font
    objDDSSecondary.SetForeColor (RGB(0, 0, 0))

    objDDSSecondary.DrawText 290, 210, "＋", False

    'サーフェスをフリップする
    Call objDDSPrimary.Flip(objDDSSecondary, DDFLIP_WAIT)

End Sub

' ═══════════════════════════════════════
'  ブランクの描画
' ═══════════════════════════════════════
Private Sub Blank()
    Dim rcRect As RECT
    With rcRect
        .Top = 0
        .Left = 0
        .Right = 0
        .Bottom = 0
    End With

    'セカンダリサーフェスを白で塗りつぶす
    Call objDDSSecondary.BltColorFill(rcRect, 255)

    'サーフェスをフリップする
    Call objDDSPrimary.Flip(objDDSSecondary, DDFLIP_WAIT)

End Sub

' ═══════════════════════════════════════
'  文字の描画
' ═══════════════════════════════════════
Private Sub Update(block As Integer, Trial As Integer, PrColumn As Integer)

    Dim rcRect As RECT
```

```
    With rcRect
        .Top = 0
        .Left = 0
        .Right = 0
        .Bottom = 0
    End With

    'セカンダリサーフェスを白で塗りつぶす
    Call objDDSSecondary.BltColorFill(rcRect, 255)

    '描画フォント
    objDDSSecondary.SetFont Me.Font
    objDDSSecondary.SetForeColor (RGB(0, 0, 0))

    '呈示列数の決定
    Select Case PrColumn
        Case 1
            objDDSSecondary.DrawText 290, 130, s(block, Trial, 1), False
            objDDSSecondary.DrawText 290, 210, s(block, Trial, 2), False
            objDDSSecondary.DrawText 290, 290, s(block, Trial, 3), False
        Case 2
            objDDSSecondary.DrawText 255, 130, s(block, Trial, 1) & s(block, Trial, 2), False
            objDDSSecondary.DrawText 255, 210, s(block, Trial, 3) & s(block, Trial, 4), False
            objDDSSecondary.DrawText 255, 290, s(block, Trial, 5) & s(block, Trial, 6), False
        Case 3
            objDDSSecondary.DrawText 225, 130, s(block, Trial, 1) & s(block, Trial, 2) & s(block, Trial, 3), False
            objDDSSecondary.DrawText 225, 210, s(block, Trial, 4) & s(block, Trial, 5) & s(block, Trial, 6), False
            objDDSSecondary.DrawText 225, 290, s(block, Trial, 7) & s(block, Trial, 8) & s(block, Trial, 9), False
        Case 4
            objDDSSecondary.DrawText 190, 130, s(block, Trial, 1) & s(block, Trial, 2) & s(block, Trial, 3) & s(block, Trial, 4), False
            objDDSSecondary.DrawText 190, 210, s(block, Trial, 5) & s(block, Trial, 6) & s(block, Trial, 7) & s(block, Trial, 8), False
            objDDSSecondary.DrawText 190, 290, s(block, Trial, 9) & s(block, Trial, 10) & s(block, Trial, 11) & s(block, Trial, 12), False
    End Select

    'サーフェスをフリップする
    Call objDDSPrimary.Flip(objDDSSecondary, DDFLIP_WAIT)

End Sub

' ═══════════════════════════════════
'  時間待ち用関数
' ═══════════════════════════════════
Function Wait(n As Long) As Long
```

```
    '垂直回帰の判定（リフレッシュレート100Hz固定，10ms以上で使用）
    Dim VBS As Long
    n = n / 10

    Do While VBS <= 2 * n
        If objDD.GetVerticalBlankStatus() = 0 Then
            Do While objDD.GetVerticalBlankStatus() = 0
            Loop
        Else
            Do While objDD.GetVerticalBlankStatus() <> 0
            Loop
        End If
        VBS = VBS + 1
    Loop

End Function

' ═══════════════════════════════════════
'  実験時にマウスボタンをクリックすると中断
' ═══════════════════════════════════════
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, x As Single, Y As Single)
    ShowCursor 1
    Unload Me
    End
End Sub

' ═══════════════════════════════════════
'  アプリケーション終了時
' ═══════════════════════════════════════
Private Sub Form_Unload(Cancel As Integer)
    'ディスプレイモードを復元
    Call objDD.RestoreDisplayMode

    '協調レベルを復元
    Call objDD.SetCooperativeLevel(Me.hWnd, DDSCL_NORMAL)

    DoEvents

    'オブジェクトを開放
    Set objDDSSecondary = Nothing
    Set objDDSPrimary = Nothing
    Set objDD = Nothing
    Set objDX = Nothing
```

```
End Sub

' ═══════════════════════════════════════
'  被験者データの入力
' ═══════════════════════════════════════
Private Sub SubjectData()
    'ダイアログボックスに被験者データを入力
    frmDialog.Show

    'ダイアログボックスに入力が完了するとループを抜ける
    bExitLoop = True
    Do While bExitLoop: DoEvents: Loop   '入力待ちループ


    a.Name = frmDialog.Text1.Text
    a.Age = frmDialog.Text2.Text
    If frmDialog.Option1 = True Then
        a.Sex = "男"
    Else
        a.Sex = "女"
    End If
    If frmDialog.Option3 = True Then
        a.ExpOrder = "全体報告条件→部分報告条件"
    Else
        a.ExpOrder = "部分報告条件→全体報告条件"
    End If

    'ファイルに保存(ExpData.txt)
    Call FileOpen
    Write #1, Format(Now, "実験日yyyy年m月d日hh時nn分ss秒"), a.Name, a.Age, a.Sex, a.ExpOrder
    Close #1
End Sub

' ═══════════════════════════════════════
'  刺激セットのシャッフル
' ═══════════════════════════════════════
Private Sub Shuffle()

    Dim x As Integer 'Rnd関数の戻り値1
    Dim xx As Integer 'Rnd関数の戻り値2
    Dim Shuffle As String 'シャッフル中の文字を一時的に保持する作業用
    Dim ShuffleTime As Long 'シャッフル時間の設定

    '時間の測定用
```

```
Dim lLastTime As Long
Dim lNowTime As Long
Dim lPastTime As Long

'刺激作成の進行状況を示すプログレスバー
frmSplash.ProgressBar1.Max = 8
frmSplash.ProgressBar1.Min = 0

ShuffleTime = 50
upper = 26      '刺激の個数ＡからＺの26
lower = 1

'８条件(i)×12試行(j)×12文字分の刺激を作成する
For i = 1 To 8
    frmSplash.ProgressBar1.Value = frmSplash.ProgressBar1.Value + 1

    For j = 1 To 12
        '刺激項目を設定する（ＡからＺまで）
        Call SetStimulus

        '刺激項目のシャッフル開始
        Randomize    '乱数ジェネレータの初期化

        'ループ前の時間を取得
        lLastTime = timeGetTime()
        lPastTime = 0
        Do While lPastTime < ShuffleTime
            'ループ中の時間を取得
            lNowTime = timeGetTime()
            'ループ開始からの経過時間を算出
            lPastTime = lNowTime－lLastTime

            x = Int((upper－lower＋1) * Rnd＋lower)
            xx = Int((upper－lower＋1) * Rnd＋lower)
            Shuffle = Stimulus(xx)
            Stimulus(xx) = Stimulus(x)
            Stimulus(x) = Shuffle
            'イベント処理
            DoEvents
        Loop

        '1試行分の刺激を格納
        For k = 1 To 12
            s(i, j, k) = Stimulus(k)
```

```
                    'List1.AddItem S(i, j, k)
                Next k
            Next j
        Next i
        'プログレスバーを0に戻す
        frmSplash.ProgressBar1.Value = 0
        frmSplash.Label1.Visible = False
        frmSplash.ProgressBar1.Visible = False

End Sub

' ═══════════════════════════════════════
'  刺激セット
' ═══════════════════════════════════════
Private Sub SetStimulus()
    'AからZまで
    Stimulus(1) = "A"
    Stimulus(2) = "B"
    Stimulus(3) = "C"
    Stimulus(4) = "D"
    Stimulus(5) = "E"
    Stimulus(6) = "F"
    Stimulus(7) = "G"
    Stimulus(8) = "H"
    Stimulus(9) = "I"
    Stimulus(10) = "J"
    Stimulus(11) = "K"
    Stimulus(12) = "L"
    Stimulus(13) = "M"
    Stimulus(14) = "N"
    Stimulus(15) = "O"
    Stimulus(16) = "P"
    Stimulus(17) = "Q"
    Stimulus(18) = "R"
    Stimulus(19) = "S"
    Stimulus(20) = "T"
    Stimulus(21) = "U"
    Stimulus(22) = "V"
    Stimulus(23) = "W"
    Stimulus(24) = "X"
    Stimulus(25) = "Y"
    Stimulus(26) = "Z"
End Sub
```

## 《リスト２（frmDialog.frm）》「注意の範囲」のプログラムリスト

```
Option Explicit

Private Sub CancelButton_Click()
    End
End Sub

Private Sub Form_Load()

End Sub

Private Sub OKButton_Click()
    If Text1.Text = "" Then
        MsgBox "被験者名が入力されていません", vbQuestion, "エラーメッセージ"
    ElseIf Text2.Text = "" Then
        MsgBox "年齢が入力されていません", vbQuestion, "エラーメッセージ"
    ElseIf Option1.Value = False And Option2.Value = False Then
        MsgBox "性別が選択されていません", vbQuestion, "エラーメッセージ"
    ElseIf Option3.Value = False And Option4.Value = False Then
        MsgBox "実験条件が選択されていません", vbQuestion, "エラーメッセージ"
    Else
        frmForm1.Show

        frmForm1.bExitLoop = False
        frmDialog.Visible = False

    End If
End Sub
```

## 《リスト３（frmDialog.frm）》「注意の範囲」のプログラムリスト

```
Option Explicit

Private Sub Form_KeyPress(KeyAscii As Integer)
    Unload Me
End Sub

Private Sub Form_Load()
    lblVersion.Caption = "ﾊﾞｰｼﾞｮﾝ " & App.Major & "." & App.Minor & "." & App.Revision
    'lblProductName.Caption = App.Title
End Sub

Private Sub Frame1_Click()
    Unload Me
End Sub

Private Sub Label2_Click()

End Sub
```

高 松 大 学 紀 要

第 36 号

平成13年９月25日　印刷
平成13年９月28日　発行

**編集発行**

高　　松　　大　　学
高 松 短 期 大 学
〒761-0194　高松市春日町960番地
TEL（087）841－3255
FAX（087）841－3064