2008 年度基礎ゼミ 127

「情報社会におけるハードウェアとソフトウェアの挑戦」

第 2 回「プログラミング体験」ワークブック

0. プログラミングって何？

（キーワード：プログラム、プログラミング言語）

1. Objective Caml の起動と終了

1-1. Objective Caml を起動してみよう。

・「スタート」→「すべてのプログラム」→「アクセサリ」→「コマンドプロンプト」をクリック

・出てきたウインドウの中で「ocaml」と打って Enter キーを押す（ただしカギ括弧は入力しない。以下同様）

どのような画面が表示されただろうか？（キーワード：バージョン、プロンプト）

1-2. Objective Caml を終了しよう。

・プロンプト「# 」が表示された状態で「#quit ;;」と打って Enter キーを押す（プロンプトの#とは別に、もう一つ自分で#を入力する必要があるので注意）

正常に終了したら、以下のために、また Objective Caml を起動しておこう。なお Objective Caml を実行していてよくわからない状態になったら、Ctrl キーを押しながら c を押せばプロンプトに戻るので、通常は間違えた問題からやり直せば良い。それでもおかしかったら Objective Caml 全体を終了し、

章の始めからやり直す。

2. 整数演算

2-1. 「1 + 2 ;;」と入力して Enter キーを押してみよう。何が表示されたか？（キーワード：式、評価、値、型）

2-2. 「3 * 4 ;;」ではどうか？

2-3. 「1 + 2 * 3 ;;」ではどうか？ 「(1 + 2) * 3 ;;」は？（キーワード：優先順位）

2-4. 「+」や「*」の他に、整数の引き算を表す「-」、整数の割り算の商を表す「/」、整数の割り算の余りを表す「mod」などが使える。それぞれの動作を確認してみよう。負の数ではどうか？

一般に、式を評価するときは、式の後に「;;」をつけて Enter キーを押せば良い。以下では式の後につける「;;」はしばしば省略する。

3. 小数演算

3-1. 「1.2 +. 3.4」「1.2 *. 3.4」「5.0 +. 6.7」をそれぞれ評価してみよう。「+」や「*」の後についている「.」を忘れないこと。

3-2. 「+」の後に「.」をつけ忘れて「1.2 + 3.4」を評価しようとしたり、小数の「5.0」を整数の「5」として「5 +. 6.7」を評価しようとすると、どうなるだろうか？（キーワード：型検査／型チェック）

3-3. 「1.0 /. 3.0」を評価するとどうなるか？ 「0.1 +. 0.1 +. 0.1」ではどうか。（キーワード：浮動小数点数、浮動小数点演算）

3-4. Objective Caml を用いて、半径 1.23 の円の面積と周の長さを求めてみよう。ただし円周率は 3.14159265359 とする。

4. 変数の定義と使用

4-1. Objective Caml を用いて、半径 4.5 の円の面積と周の長さを求めよ。半径 67.8 の円ではどうか。

4-2. 「let pi = 3.14159265359 ;;」と入力し、Enter キーを押してみよう。

4-3. 4-2 を実行した後で「pi」という式を評価するとどうなるか。「1.23 *. 1.23 *. pi」や「2.0 *. 1.23 *. pi」ではどうか。

定義を実行するときも、式の評価と同様に後に「;;」をつけて Enter キーを押せば良い。以下では定義を実行するときにつける「;;」もしばしば省略する。

4-4. 4-2 で定義した変数 pi を使って、4-1 をやり直してみよう。

5. 関数呼び出し

5-1. 「sqrt 2.0」という式を評価してみよう。「sqrt 3.0」や「sqrt 4.0」ではどうか。(キーワード：関数の引数と返値)

注：普通の数学と異なり、Objective Caml では f(x)や g(3)などの括弧を省略して「f x」「g 3」のように書くことができる。ただし引数が複雑なときは括弧が必要（自信がないときは括弧をつける)。

5-2. 平方根関数「sqrt」の他に、三角関数「sin」「cos」「tan」なども使うことができる（引数の単位はラジアン)。「sin (pi /. 3.0)」などの式を評価して、いろいろな角度に対する三角関数の値を計算してみよう。

6. 関数の定義

6-1. 半径 1.2 と 3.45 と 67.89 の円の面積をそれぞれ求めてみよう。

6-2. 「let menseki r = r *. r *. pi ;;」と入力し、Enter キーを押してみよう。型はどう表示されるか？

6-3. 「menseki 1.2」「menseki 3.45」「menseki 67.89」をそれぞれ評価してみよう。

6-4. 同様に円の半径から周の長さを計算する関数を定義し、半径 1.2 と 3.45 と 67.89 の円の周の長さを求めてみよう。

注：もし pi の値を変更したら、menseki の定義も実行し直す必要がある。一般に、変数や関数を定義し直したら、それを用いている変数や関数も定義し直すこと。

7. グラフィックス

7-1. 「#load "graphics.cma"」「open Graphics」をそれぞれ順番に実行してから、「open_graph ""」「draw_circle 100 100 50」をそれぞれ順番に実行してみよう。何が起こったか？（キーワード：ライブラリ、モジュール、unit 値「()」と unit 型）

7-2. 関数 draw_circle の引数をいろいろと変えて試してみよう。どの引数が何を表しているか推測してみよう。

7-2. WWW ブラウザで Objective Caml の Graphics モジュールのマニュアル日本語訳を開こう（「http://ocaml.jp/」を開き、左メニュー中央「マニュアル」の「3.06（和訳版）」→「Part IV Objective Caml ライブラリ」の「The graphics library」→下方「Module Graphics: machine-independent graphics primitives」と辿る。または「http://ocaml.jp/archive/ocaml-manual-3.06-ja/libref/Graphics.html」を開く）。関数 draw_circle の説明を探して読んでみよう。

7-3. 関数「draw_rect」「draw_ellipse」「draw_arc」「fill_rect」「fill_circle」「fill_ellipse」「fill_arc」の説明を読んで、それぞれ実際に使ってみよう。画面がゴチャゴチャしてきたら、「clear_graph ()」を実行すればクリアできる。

7-4. 関数「set_color」に color 型の値「blue」「red」「green」などを与えて実行してみよう。それから draw_circle などを呼び出すとどうなるか？（あらかじめ用意されている色の名前はマニュアルの「Some predefined colors」を見よ。それ以外の色を作りたいときは関数「rgb」を使う。）

7-5. 関数「moveto」と関数「lineto」の説明を読み、実際に使ってみよう。

7-6. 整数 x と y を受け取り、座標(x,y)を左下とする一辺の長さが 100 の正方形を描く関数「square」を定義してみよう。（ヒント：複数の引数があるので、「let square x y = 」のように書き始める。）

7-7. 整数 x と y を受け取り、座標(x,y)を上の頂点とする正三角形を描く関数「triangle」を定義してみよう。ただし、整数を小数に変換する関数は「float_of_int」、小数を整数に変換する関数は「int_of_float」である。また、複数の式を順に実行するには、「式$_1$; 式$_2$; 式$_3$; …」のように書けば良い。

8. 繰り返し／反復／ループ

8-1. 「clear_graph ()」を実行して画面をクリアしてから、

```
for i = 1 to 10 do
  draw_circle 100 100 (10 * i)
done
```

を実行してみよう（Objective Caml のプログラムは、単語の途中以外であればどこでも改行してよい。以下同様)。どんな絵が描けたか？

8-2. 8-1 の数字や関数(draw_circle)をいろいろに変えて、どうなるか試してみよう。

9. 再帰

9-1. 「clear_graph ()」で画面をクリアしてから、次の関数定義を実行しよう。

```
let rec f x =
  if x < 1 then () else
  (draw_circle 100 100 x;
   f (x / 2))
```

それから式「f 100」を実行したらどうなるだろうか？ ただし「if 式$_1$ then 式$_2$ else 式$_3$」は、式$_1$が成り立てば式$_2$を、成り立たなければ式$_3$を評価する、という意味の式である。

9-2. 9-1 の関数定義で、「if x < 1 then () else」の行を忘れて

```
let rec f x =
  (draw_circle 100 100 x;
   f (x / 2))
```

と定義してしまい、「f 100」を実行するとどうなるか？（注：実行を中止したいときは Ctrl キーを押しながら c を押す。）

10. ネットワークプログラミングその 1 (HTTP)

（計算機環境によっては実行できないこともあるので、その場合は省略する）

10-1. 「#load "unix.cma"」「open Unix」を実行して、UNIX モジュール（ライブラリ）をロードおよびオープンしよう。

UNIX とはオペレーティングシステム(OS)の名前である。いわゆるインターネットで使用されている TCP/IP 通信方式は、UNIX を中心に発展してきた。現在では UNIX 以外の OS にも UNIX と同様の TCP/IP 通信機能がある。

インターネットでは、「www.tohoku.ac.jp」のような人間にわかりやすい名前（ホスト名）の他に、「123.45.67.89」のような数字のアドレス（IP アドレス）がある。

10-2. 「let h = gethostbyname "www.tohoku.ac.jp"」および「let a = h.h_addr_list.(0)」を実行しよう。さらに、「string_of_inet_addr a」を評価してみよう。どんな結果が表示されたか？

いわゆるインターネットの「ホームページ」は、HTML という言語で記述されている。

10-3. WWW ブラウザで「http://www.tohoku.ac.jp/japanese/index.html」を開き、右クリックして「ソースの表示」をしてみよう。

10-4. 以下の定義や式を、順に実行ないし評価してみよう。

```
let (ic, oc) = open_connection (ADDR_INET(a, 80)) ;;
output_string oc "GET /japanese/index.html¥r¥n¥r¥n" ;;
flush oc ;;
print_endline (input_line ic) ;;
print_endline (input_line ic) ;;
print_endline (input_line ic) ;;
```

（上の行をしばらく繰り返す。↑キーを押して Enter キーを押せば良い。）

```
close_in ic ;;
```

一般に WWW ブラウザは、このようにしてダウンロードした HTML 文書を解析・描画することにより、ホームページを表示している。

10-5. 10-2 から 10-4 までの「www.tohoku.ac.jp」や「/japanese/index.html」の部分を他のホスト名やファイル名におきかえて、いろいろなホームページの HTML 文書をダウンロードしてみよう。

11. ネットワークプログラミングその 2（SMTP）

（計算機環境によっては実行できないこともあるので、その場合は省略する）

11-1. 以下の定義や式を順に実行ないし評価し、メールを送信してみよう。送信元は東北大学での自分のメールアドレス、送信先は自分の携帯電話のメー

ルアドレスにしよう。(もし携帯電話のメールアドレスを持っていなければ、自分の他のメールアドレスでも良い。他人のメールアドレスを使用しないこと。)

```
let h = gethostbyname "smtp.cs.he.tohoku.ac.jp" ;;
let a = h.h_addr_list.(0) ;;
let (ic, oc) = open_connection (ADDR_INET(a, 25)) ;;
print_endline (input_line ic) ;;
output_string oc "HELO localhost¥r¥n" ;;
flush oc ;;
print_endline (input_line ic) ;;
output_string oc "MAIL FROM: 送信元メールアドレス¥r¥n" ;;
flush oc ;;
print_endline (input_line ic) ;;
output_string oc "RCPT TO: 送信先メールアドレス¥r¥n" ;;
flush oc ;;
print_endline (input_line ic) ;;
output_string oc "DATA¥r¥n" ;;
flush oc ;;
print_endline (input_line ic) ;;
output_string oc "From: 送信元メールアドレス¥r¥n" ;;
output_string oc "To: 送信先メールアドレス¥r¥n" ;;
output_string oc "Subject: test mail¥r¥n" ;;
output_string oc "this is a test¥r¥n" ;;
output_string oc ".¥r¥n" ;;
flush oc ;;
print_endline (input_line ic) ;;
output_string oc "quit¥r¥n" ;;
flush oc ;;
print_endline (input_line ic) ;;
close_in ic ;;
```

11-2. 11-1 において、「送信元メールアドレス」に「dummy@example.com」(実在しない)を使用したらどうなるだろうか?

12. 3Dグラフィックス

（発展的内容なので割愛する場合もある）

Objective Caml を終了し、「ocaml」のかわりに「lablglut」を起動してから、以下の定義や式を順に実行ないし評価しよう。(*と*)で囲まれた部分はコメントなので、入力しなくても良い。

注:「`」はバッククォートという記号で、標準的な日本語キーボードでは、Shift キーを押しながら@キーを押せば入力できる。「'」（クォート）とは異なるので気をつけること。「~」はチルダという記号で、標準的な日本語キーボードでは、Shift キーを押しながら^キーを押せば入力できる。

```
(* 初期化処理 *)
Glut.init Sys.argv ;;

(* ウインドウの準備 *)
Glut.createWindow "kisozemi" ;;

(* ライトを有効化 *)
Gl.enable `lighting ;;
(* 1番のライトを白色の環境光に設定 *)
GlLight.light 1 (`ambient (1.0, 1.0, 1.0, 1.0)) ;;
(* 1番のライトを有効化 *)
Gl.enable `light1 ;;
(* 2番のライトを赤色の点光源に設定 *)
GlLight.light 2 (`diffuse (1.0, 0.0, 0.0, 1.0)) ;;
(* 2番のライトを右方に設置 *)
GlLight.light 2 (`position (2.0, 0.0, 0.0, 0.0)) ;;
(* 2番のライトを有効化 *)
Gl.enable `light2 ;;
(* 3番のライトを青色の点光源に設定 *)
GlLight.light 3 (`diffuse (0.0, 0.0, 1.0, 1.0)) ;;
(* 3番のライトを左方に設置 *)
GlLight.light 3 (`position (-2.0, 0.0, 0.0, 0.0)) ;;
(* 3番のライトを有効化 *)
Gl.enable `light3 ;;
```

```
(* 描画関数 *)
let rec display () =
  (* 画面をクリアするときの色を黒に設定 *)
  GlClear.color (0.0, 0.0, 0.0) ;
  (* 画面をクリア *)
  GlClear.clear [`color] ;
  (* サイズ0.5のティーポットを描画 *)
  Glut.solidTeapot 0.5 ;
  (* バッファをフラッシュ *)
  Gl.flush () ;;

(* 描画関数を登録 *)
Glut.displayFunc display  ;;

(* キーボード処理関数 *)
let keyboard ~key:k ~x:x ~y:y =
  (* ティーポットを回転 *)
  GlMat.rotate ~angle:2.0 ~x:0.25 ~y:0.5 ~z:1.0 ();
  (* 描画関数displayを呼び出して，全画面を再描画 *)
  display () ;;

(* キーボード処理関数を登録 *)
Glut.keyboardFunc keyboard ;;

(* メインルーチンを起動 *)
Glut.mainLoop () ;;
```

以上をすべて実行すると、「kisozemi」ウインドウが出てくるのでスペースキーなど適当なキーを何度も押してみよう。終了するときは「コマンドプロンプト」ウインドウ右上のxボタンでウインドウを閉じれば良い。

課題

Objective Caml の機能を利用して、何か好きな絵を描いてみよう。9 章（グラフィックス）までに出てきたいろいろな機能をできるだけ利用すること。できた絵と、その絵を描くために入力したプログラム全部（式や変数定義・関数定義）を、sumii@ecei.tohoku.ac.jp へメールせよ。

・プログラムはメモ帳（「スタート」→「すべてのプログラム」→「アクセサリ」→「メモ帳」）などに書いてからコピー&ペーストすると良い。（メモ帳でコピーしたい範囲をドラッグして選択し「編集」→「コピー」、コマンドプロンプトで左上のアイコンをクリックして「編集」→「貼り付け」）

・絵をセーブするには、「Caml Graphics」のウインドウで Alt キーを押しながら PrintScreen キーを押し、「スタート」→「すべてのプログラム」→「アクセサリ」→「ペイント」を起動して「編集」→「貼り付け」、「ファイル」→「名前を付けて保存」とすれば良い。

・セーブした絵はメールに添付して送ること。（送り方は口頭で説明します）