コンピュータ基礎実習上級 教材 (#8)

■ 入力、scanf、if、switch.

pinrtf ( ) などによるプログラムからの結果の取り出し（人間に見える形にする）を出力と呼んでいます。これに対してプログラムに値や指示を与えることを入力と呼びます。
ここではそうした方法のうちの一つ、キーボードから文字入力を行う例を示します。

□ scanf による文字入力

右のプログラムを入力して実行してください。

実行するとキーボードからの入力を待つ状態になります。以下のように 230 と入力すると、消費税を計算して結果を画面に出力します。

```
230
total = 241
```

```
#include <stdio.h>
/*
  消費税計算をする 473088 榎田裕一郎
*/
int main() {
  int price, total;

  scanf("%d", &price);
  total=price * 1.05;
  printf("total = %d\n", total);

  return 0;
}
```

`scanf("%d", &price);` で `price` 変数に入力された値（この例では `230` ）が入ります。

書式 :
`scanf(”書式”, &変数 1, &変数 2, … &変数 n);`

”書式”には入力する変数の数だけ `%` に続けて入力する形式を `%d` や `%f` の変換文字列で指定します。変換文字列は `printf` のそれと同じで、`%d` は十進数の整数、`%f` は `float` 型の実数に、`%lf` が `double` 型の実数に対応します。変数の前の `&` については今は説明しません。

つまり二つの変数に値を設定したい場合は、以下のように書きます。
```
scanf("%d %f", &price, &tax);
```

二行に分けて書く方法もあります。
```
scanf("%d", &price);
scanf("%f", &tax);
```

どちらの方法でも、実行時の入力の方法は同じです。二行に分けて入力することもできます。

```
230 1.05
total = 241
```

```
230
1.05
total = 241
```

● 課題 1.

上記の例のように税率を scanf で入力するようにして実行し、動作を確認してください。
（100 円に 1.05 の税率で 104 円と計算したり、若干誤差が出る場合があります。内部的には電卓で 1 ÷ 3 × 3 とすると 1 ではなく 0.999… と出るのと同じことが起きているのです。）

● 課題 2.

先週のプログラムを修正して、小さな四角の進行方向（dx, dy） を入力できるようにしてください。

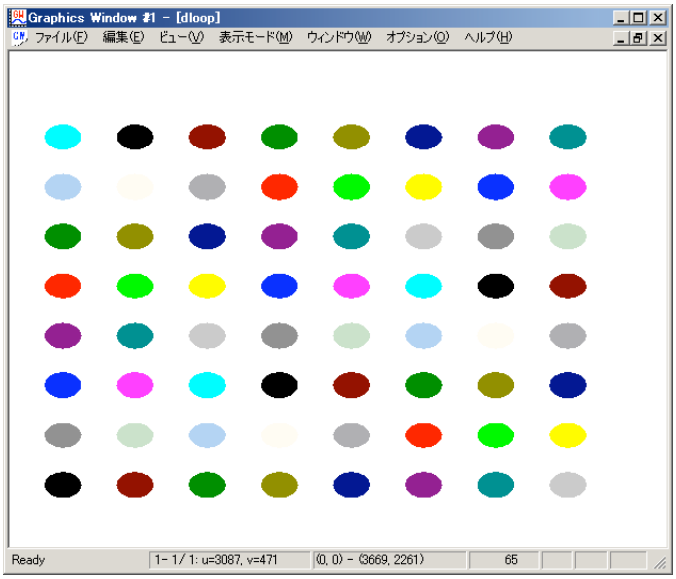
● 課題 3.

入力に応じて処理を変える

先週の小さな四角を描くプログラムに手を加えて、入力に対応して四角ではなく丸を描くように機能追加してください。

実行すると以下のように 1 か 2 のどちらかを入力するように求め、1 ならば四角を、2 ならば円を描くようにしてください。

```
drawing type? (1-2) = 2
```



以下の点に注目してプログラムを修正して下さい。

押さえて欲しいところ：

・printf( ) でプロンプトとなる文字列を出力。（改行なし）
・scanf( ) で描画図形のタイプを入力し、int 型変数 type に格納。
・type の値によって描く図形のタイプを if 文で変更。
・GWsrect( ) で四角形（塗りつぶし）を描く。
・GWscircle ( ) で楕円（塗りつぶし）を描く。

修正サンプル：

```
printf("drawing type? (1-2) = ");
scanf("%d",&type);
..  （中略）..
    if( type == 1 ) {
      /* 塗りつぶした四角形 */
      GWsrect(x, y, x+10.0, y+10.0, c);
    } else {
      /* 塗りつぶした円 */
      GWscircle(x, y, x+10.0, y+10.0, c);
    };
```

GWscircle ( ) 関数の引数は描く楕円の座標位置を示しており、楕円が外接する四角形の左下、右上の座標位置 (x1, y1) – (x2, y2) の順で示します。

```
GWscircle (x1, y1, x2, y2, c);
```

最後の引数 c は色番号です。

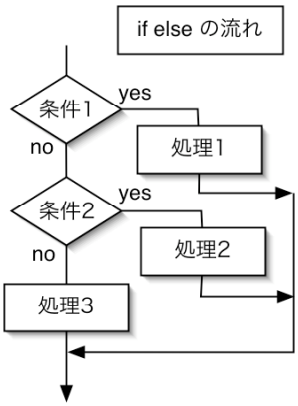
□ さまざまな if 文の書き方

1, 2 以外の値を入力した場合の例外処置を右のように含めました。1 でも 2 でもない場合は描画ウィンドウを閉じてプログラムを途中で終了します。else if 句の使い方に注目して下さい。

```
if( type == 1 ) {
  /* 塗りつぶした四角形 */
  GWsrect(x, y, x+size, y+size, c);
} else if( type == 2 ) {
  /* 塗りつぶした円 */
  GWscircle(x, y, x+size, y+size, c);
} else {
  printf(" 1 か 2 で指定してください。\n");
  GWquitx(0);
  return 0;
};
```

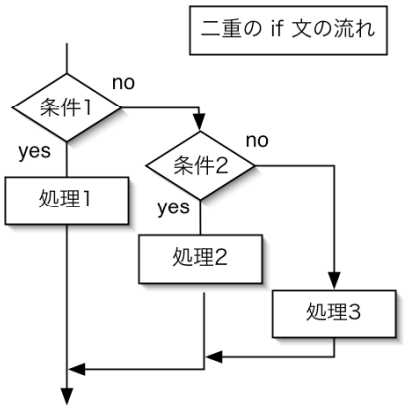
（下の例では一つですが、else if は幾つでも続けて書けます。）

```
if(条件 1) {
  処理 1;
} else if(条件 2) {
  処理 2;
} else {
  処理 3;
};
```



同じ意味の処理を if 文を入れ子にして表現することもできます。入れ子も何重にでもできます。

```
if(条件 1) {
  処理 1;
} else {
  if(条件 2) {
    処理 2;
  } else {
    処理 3;
  };
};
```

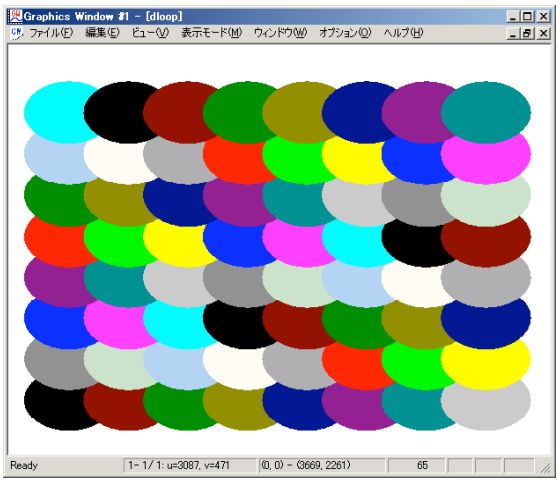


● 課題 4.

・上記の例外処置を組み込み、
・type だけでなく、長さも入力し、
・四角形の辺の長さ、円の径をその長さで描画する。

つまり以下のような実行時の入力によって右図のような結果となるように。（例は 2　30.0 と入力した。）

```
drawing type (1-2) and size = 2 30.0
```



□ switch 文による場合分け

if else を使った条件分岐を、switch 文を使ってより分かりやすく記述できる場合があります。

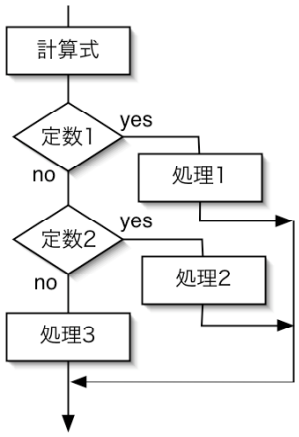
```
switch( type ) {
case 1:
  GWsrect(x, y, x+10.0, y+10.0, c); /* 四角を描く */
  break;
case 2:
  GWscircle(x, y, x+size, y+size, c); /* 円を描く */
  break;
default:
  printf(" 1 か 2 で指定してください。\n");
  Gwquitx(0);
  return 0;
};
```

switch( ) のカッコ内に変数や計算式を書き、その結果がどのケースに該当するかによって処理を分岐させます。

書式：

```
switch( 式 ) {
case 定数 1:
  処理 1;
  break;
case 定数 2:
  処理 2;
  break;
..（略）..
default:
  処理 n;
}
```

switch の流れ



注意点：
・case に続く条件部分には定数または定数式しか書けません。（ x < 10 など条件文は不可。）
・case による分岐は幾つでも書けます。
・どのケースにも該当しなかった場合は default: に続く処理を実行します。
・default は省略できます。（その場合該当しないケースでは何もせずに swtich を通過します。）
・各処理の最後には break; を付けてください(*1)。
・switch の直後の { と、対応する } を忘れないように。

● 課題 5.

サンプルプログラムを加工し、switch を使って描く図形を 4 種類指定できるようにしてください。あと二つの図形には角の丸い四角を描く GWrrect( ) 、扇形を描く GWspie( ) を利用すると良いでしょう。各関数の仕様については<<ガイド 9.3>> を参照。

*1 たとえば一番目の処理の最後に break; が存在しなかった場合、処理 1 が終わると無条件に処理 2 を実行します。なぜこのような言語仕様になっているのかはここでは説明しません。興味のある人はコンパイルされたコードを読むと分かるでしょう。