０．はじめに

さて、今回は関数、命令の扱い方と、ラベルによる操作について説明します。前回やった「mes 命令」と変数に関する操作についても引き続き使用します。

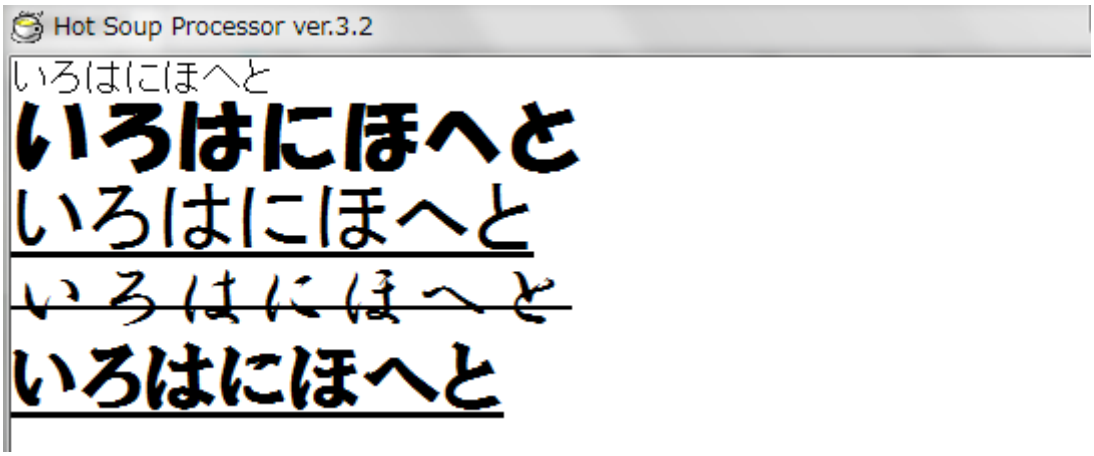
１．命令

前回の「mes 命令」や「stop 命令」のように、その文字列で１つの動作をしてくれるものを命令といって、たくさんの種類があります。ここで、命令とはどんなものかをおおざっぱに理解できればいいかなと思います。ちなみに、命令は水色で表示されます。今回は、その中でもよく使うものを取り上げていくので、すべて覚えなくていいですが、「あぁ、こんなものがあったな」程度に覚えておいてください。

１．１「mes」の補助命令

「mes」には補助命令があります。これらにより文字のフォントや大きさを変えることが可能です。それが「font」です。「font」では文字のフォント（明朝、ゴシック etc.）、大きさを変更できるほか、文字に装飾を施すことができます。

```
untitled *
1   font "MSゴシック",20,0//←装飾
2   //     ↑フォント   ↑文字サイズ
3   mes "いろはにほへと"
4   font "HG創英角ﾎﾟｯﾌﾟ体",40,1//装飾…太字
5   mes "いろはにほへと"
6   font "MS明朝",40,4//装飾…下線
7   mes "いろはにほへと"
8   font "HG行書体",40,8//装飾…打ち消し線
9   mes "いろはにほへと"
10  font "HGP創英ﾌﾟﾚｾﾞﾝｽEB",40,5//装飾…下線＋太字
11  mes "いろはにほへと"
12  stop[EOF]
```



文字サイズ、フォントの変更方法はプログラムからわかるんじゃないかと思います。「font

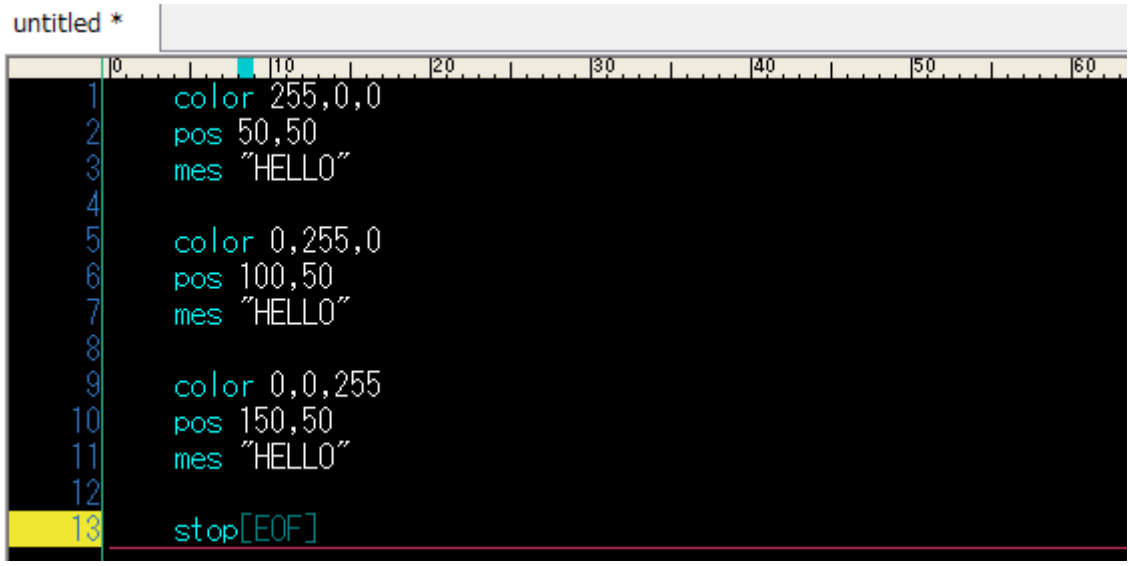
“文字フォント”,文字サイズ,装飾という順番で表記します。さて、文字の装飾についてですが、装飾については次のように振り当て得られています。

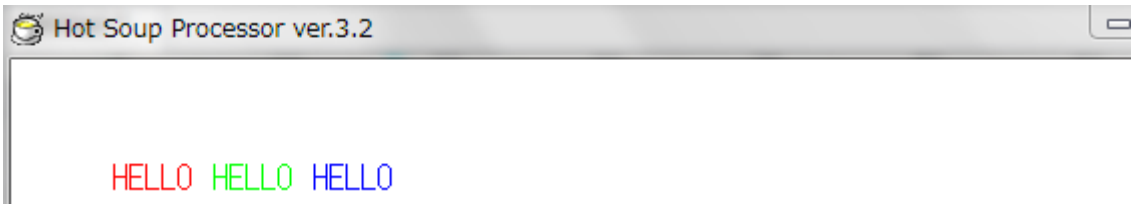
1	太文字
2	イタリック体
4	下線
8	打ち消し線
16	アンチエイリアス

太文字、下線、打ち消し線は上のプログラムで試したのでわかると思います。イタリック体は、文字が斜めになり、アンチエイリアスは境界をぼやかして文字を滑らかにします。実際に自分で試してみるといいと思います。これらを複数使いたい場合、たとえば太字かつイタリック体というときはそれらの数の合計、つまり１＋２＝３を装飾モードとして代入すれば、実装できます。上のプログラムでも最後の表示は太字＋下線の１＋４＝５を代入しています。もちろん、3 つ以上を足して代入することも可能です。

１．２補助的な命令

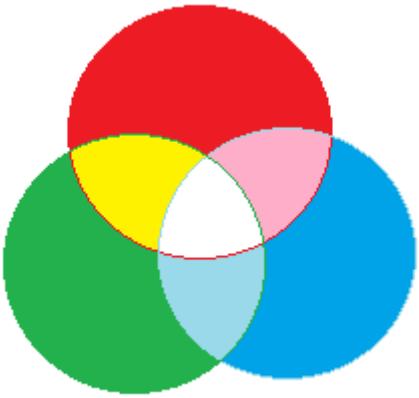
次に汎用性があり、かつ「mes」の補助もできる命令を紹介します。今回はプログラム上で何かを表示する命令に補助として使える「color」「pos」を紹介します。「color」はその名の通り色を、「pos」は position、つまり表示位置を指定できます。





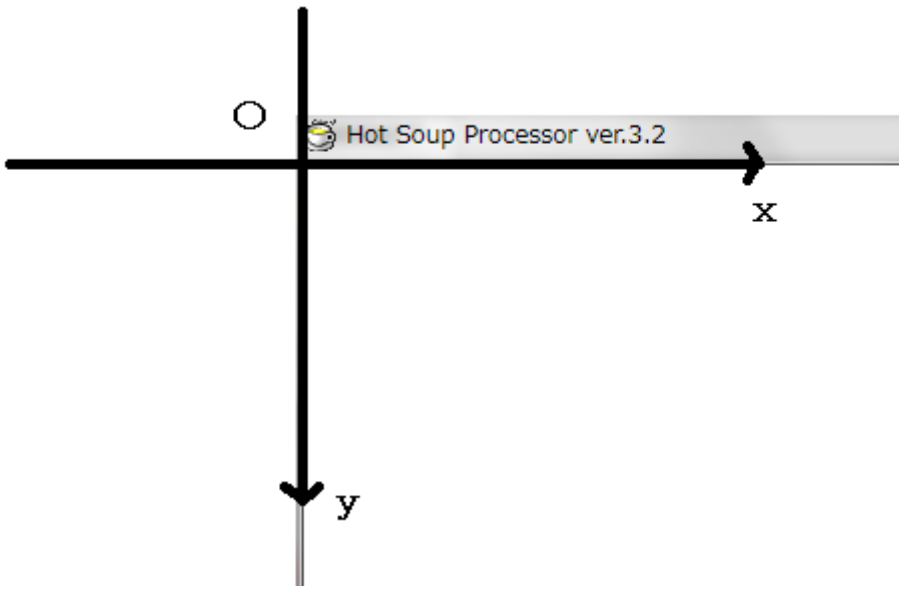
まず、「color」命令から解説します。「color R,G,B」のように表記します。R,G,B には赤、緑、青の光色の輝度を 0～255 の間で選択します。それぞれの混合による色の変化は下のよ

うな感じになります。

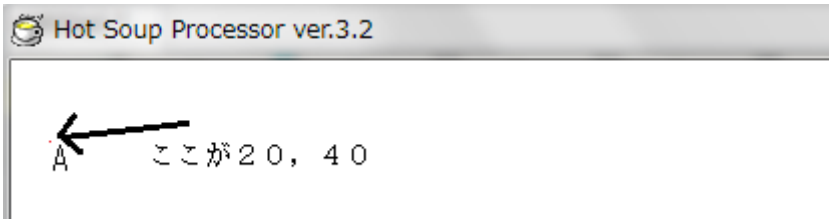


赤＋緑で黄色になり、赤＋青でピンクっぽい色になります。普段の感覚と若干違うので、いろいろ数値をいじって試してみてください。（後で課題１．２，１．３があります）

次に、「pos」です。「pos x,y 」の形式で表記して、表示位置を指定することができます。表示画面の左上を原点として、右方向に x 軸、下方向に y 軸とみなして、座標を指定します。



たとえば、「pos 20,40」とすれば、次のような場所になります。（単位は 1 ドットです）



ちょっと見えにくいですが、赤いドットのあたりになります。mes 命令は pos で指定し

た場合は、その位置を左上の基準点として表示をします。「mes」命令以外にも表示系命令はいろいろありますが、基本的に左上を基準点として、pos 命令で基準点の位置を指定できます。

課題１．１

次のような表示をするプログラムを作りなさい。表示している文字は「イルカ」である。

課題１．２

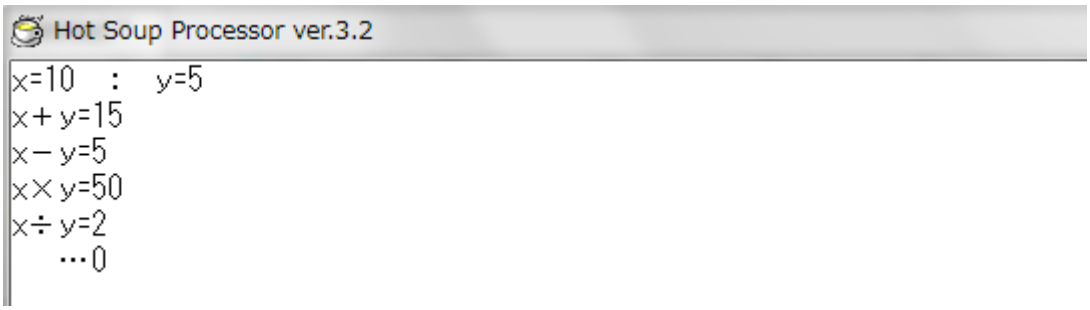
次のような色の文字を表示しなさい。おおよそ同じ色であればいい。

課題１．３

次のような色の文字を表示しなさい。おおよそ同じ色であればいい。

課題１．４

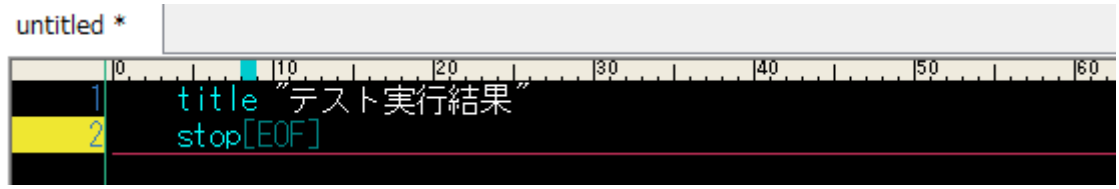
前回の課題１．６と同じなように、変数の計算結果を表示するプログラムを作りなさい。ただし、文字列の足し算を行ってはならず、pos 命令を用いて実装すること。



１．３よく使いそうな命令

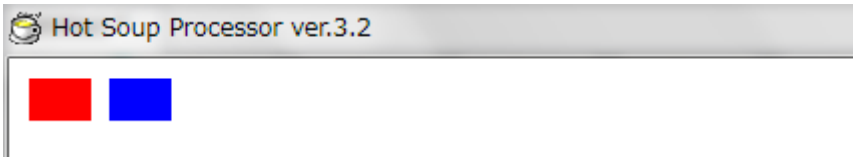
いままでは、mes 命令を補助する命令を紹介してきましたが、2 つほど、mes 命令とは違う画面出力命令を紹介しておきます。１つめは、「title」命令です。使い方としては、mes 命令と同じように「title “文字列”」の形で表記します。実際にどのような命令か、つぎの実

行結果で確認してみてください。



上の実行結果から予想できるように、title 命令ではウィンドウのタイトルを変更することができます。

もう１つは、boxf 命令です。先ほどの color 命令は覚えているでしょうか。この boxf 命令も color 命令で色の変更ができます。実際にどのように使うかを見てみましょう。



boxf命令は、指定した位置に指定した大きさの四角形を描くことができる命令です。「boxf 四角形の上端のｙ座標,左端のｘ座標,下端のｙ座標,右端のｘ座標」のように記述します。座標軸の取り方は先ほどの pos 命令と同じく、ウィンドウの枠内左上端を原点とし右方向にｘ座標、下方向にｙ座標をとります。

課題１．５

次のような実行結果を表示するプログラムを作りなさい。なお、画面内に表示されている四角形の大きさおよび位置は、次の表のとおりである。

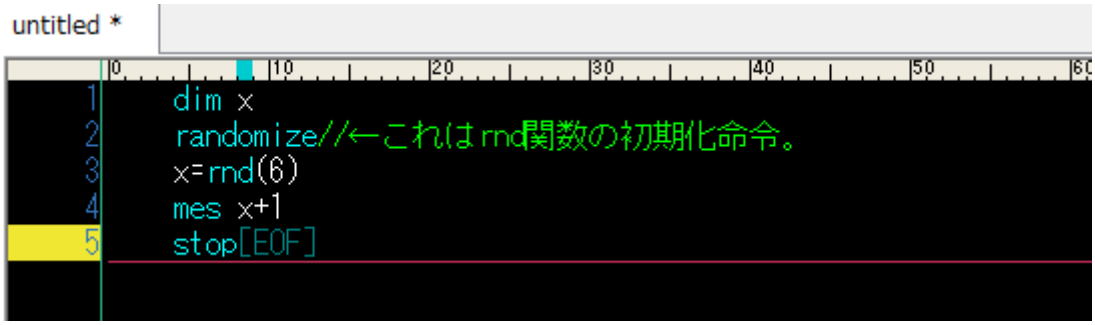
色	位置（四角形の左上の位置）(px)	横幅（px）	高さ(px)
赤	(10,10)	40	40
青	(60,60)	60	40
緑	(130,110)	80	60

２．関数

さて、いままでの命令は、使用すると、決まった処理をして終了するだけでした。関数とは、実行すると、値を返してくる命令のことを言います。関数か命令かは、返り値の有無で判断できます

では、実際に関数の使い方を説明します。今回は簡単なもので、かつ比較的よく目にする２つ紹介します。

１つ目は「rnd」という関数です。これは、「randomize」からきており、その名の通り乱数を発生させます。ランダムに状況を変化させたい時や、戦闘システムのあるゲームではダメージを若干値変化させる補正としてよく使われます。



このプログラムを何回か実行してみてください。実行するたびに１～６の中からランダムに 1 つが表示されるのがわかると思います（これがさいころのプログラム）。2 行目に「randomize」とありますが、これは rnd の補助的な命令で、rnd 関数を使うときは rnd 関数より上に、できればプログラムのなるべく上のほうに書くようにしてください。これがなくなるとどうなるのでしょうか。実際にやってみてください。さて、rnd 本体の説明です。関数は変数と同じようにその文字列自体が数値などの情報を持っています。（これが返

り値です）なので、「変数＝関数」とすることで関数の中身を変数に代入することができます。さて、rnd は「rnd(数値)」という形式で記述することで、0 以上その数値未満の数字のうちランダムに１つを返り値として持ちます。ですので、「rnd(6)」と記述することで、これは０～５のなかのどれか １ つを返り値に持つことになります。ところが、さいころでは１～６を表示させたいため、０～５＋１することで１～６を表示させています。

課題２．１

　コイントスのプログラムを作りなさい。裏を０、表を１として表示し、実行するたびに結果が変わるようにすること。

次にもうひとつだけ、「gettime」関数を紹介します。その名の通り時間を習得する関数です。

```
untitled *
1   dim year
2   dim month
3   dim day
4   dim date
5   dim hour
6   dim min
7   dim sec
8   sec=gettime(6)
9   min=gettime(5)
10  hour=gettime(4)
11  day=gettime(3)
12  date=gettime(2)
13  month=gettime(1)
14  year=gettime(0)
15  mes ""+year+"年"+month+"月"+day+"日"
16  mes ""+hour+"時 "+min+"分 "+sec+"秒"
17  stop[EOF]
```

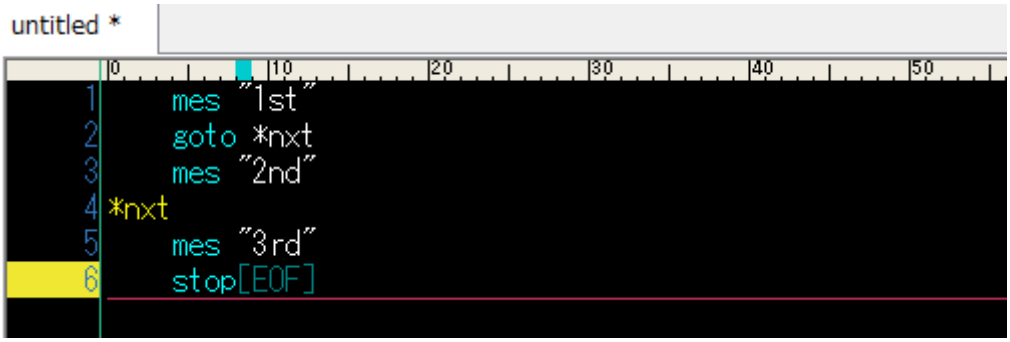
```
Hot Soup Processor ver.3.2
2012年3月9日
0時 27分 4秒
```

　文字列として足し算をさせるために、最初に mes 命令の引数（命令や関数に与える情報）として“”（空文字列）を与えています。「gettime(数値)」では与えた数値によってその時の PC タイムの情報を習得し、返り値とします。年月日、時分秒は上の結果からわかると思います。ほかに、ミリ秒（1/1000 秒）は gettime(7)で、曜日は gettime(2)で習得できますが、この関数は返り値が整数型なので曜日も数字で返ってきます。工夫することでこれを

文字列で表示させることができますが、その方法は第４回の「if 分」で紹介したいと思います。

３．ラベルの操作

さて、ラベルの操作は HSP におけるプログラムの中では最重要項目の中の一つで、かつ HSP でプログラムを描く利点にもなります。しっかり理解するようにしてください。



ラベルとは目印のことで Tab を入れずに「＊ラベル名」の形で書き、黄色で表示されます。今回は「*nxt」というラベルを使っています。では、これを実行させてみましょう。

プログラムは上から順番に処理を行っているはずなのに、mes ”2nd” の処理が飛んでいますね。順番に処理を追っていきましょう。まず、1 行目で mes “1st”は表示されます。次に 2 行目の goto *nxt ですが、ここで新しく「goto」命令が出てきます。これは、「goto *ラベル名」の形で表記し、*ラベル名　で指定したラベルまで処理を移動します。よって、今回は 4 行目のラベル、*nxt まで移動します。処理は 4 行目には関数や命令などはないので 5 行目に行き、mes “3rd”を表示してプログラムが終了する、といった流れです。

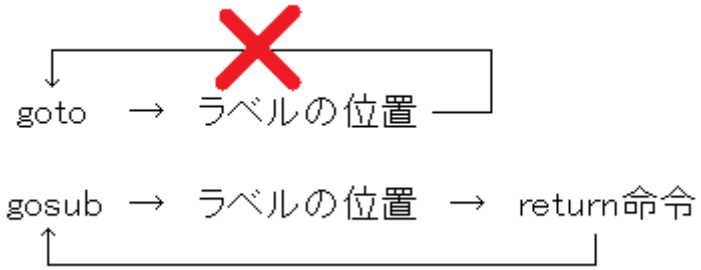
課題３．１

次のような処理をするプログラムを作りなさい。

「変数ｘ、ｙおよびラベル「*a」「*b」の 2 つを作り、プログラム内でｘ、ｙの値と goto 命令でラベル名を指定し、「*a」を指定したなら x とｙの足し算結果を、「*b」を指定したならｘとｙの引き算の結果を表示する」

さて、いまはラベルをつくり、goto 命令でジャンプするといった方法を紹介しましたが、ほかに gosub 命令というのがあります。gosub 命令も goto 命令と同じく、「gosub *ラベル名」の形で使用しますが、goto 命令は一方通行にラベルへジャンプするだけなのに対し、gosub 命令は一度ラベルに飛んだ後、ラベルとセットで配置する return 命令でもう一度元

の場所に返ってくるという特性を持っています。



そして、gosub 命令でジャンプするラベル～対応する return 命令間を「サブルーチン」と呼びます。実際にプログラムを見てみましょう。

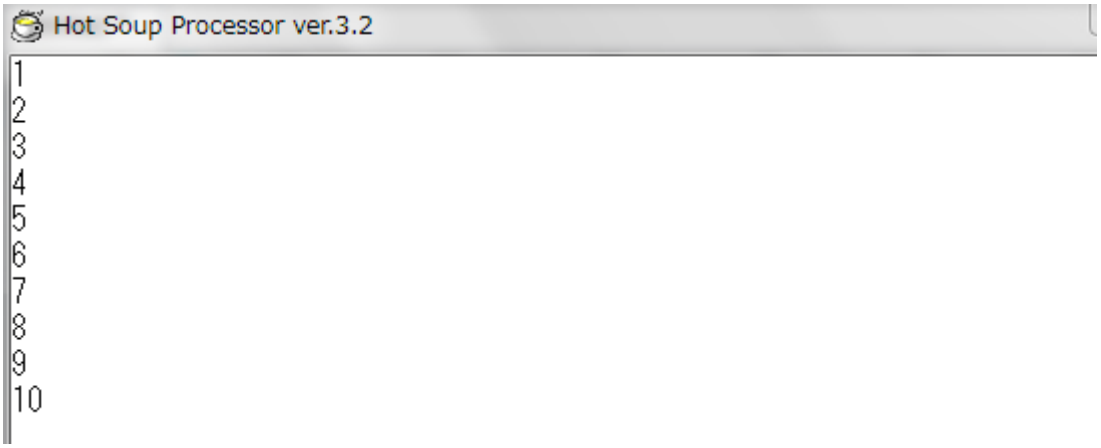
```
untitled *
    gosub *sub
    mes "1st"
    stop

// サブルーチン
*sub
    mes "2nd"
    return

    // 念のため
    stop[EOF]
```

```
Hot Soup Processor ver.3.2
2nd
1st
```

再び、プログラムが実際に処理される順を追って見てみましょう。まず、1 行目で gosub 命令により、指定されているラベル、つまり*sub まで処理をジャンプします。さて、*sub までジャンプしたので、次の行（7 行目）の mes 命令で「2nd」を表示します。次の行に行き、return 命令が来たので、サブルーチンが終了し、ここに飛んできた gosub 命令の次の行、つまり 2 行目に戻ります。ここで、mes 命令により今度は「1st」が表示され、最後に stop 命令で処理がストップする、といった順番になります。最後の行（11 行目）の stop はプログラムがうまくいかなくなった時の念のために入れてあり、処理もされるわけではないので今回は特に気にしなくて大丈夫です。このように、サブルーチンはいったん上から順番に流れる処理から離れて別の場所にプログラムを書くことができるので、プログラムあとから見やすいように書いたり、何度も使うプログラムをサブルーチンとして書くことで、gosub 命令 1 つで同じ処理を繰り返すことが可能になったりします。このあたりについては、後々詳しく触れたいと思っています。また、これを使うことでループ（同じ処理

を繰り返させること）を作ることができるようになりますが、そちらについては第 3 回で話をしたいと思います。

課題３．２　サブルーチン（整数値型変数 x に 1 を加算して x の値を表示する）を作り、変数 x を 1 から１０まで変化させてそのたびに表示するプログラムを作りなさい。なお、使っていい変数は１つまでとし、mes 命令はプログラム中 1 箇所でしか使用してはならない。実行例を以下に示す。



課題３．３　gosub 命令を使わず、goto 命令のみで上と同じ条件下で上と同じ処理をするプログラムを作りなさい。なお「サブルーチン」を作るのでなく、goto 命令により疑似的にサブルーチンと同じ動きをさせることでこの問題を解決しなさい。