０．はじめに

前回のループ同様、今回扱う「分岐」は、プログラムの中で最重要項目の一つです。いくつかのパターンがありますが、用途に合わせて上手に使い分けましょう。今回は、「if 文」と「switch-case 文」の 2 つを紹介します。

１．If 文

分岐の中でも最もオーソドックスなのが、この if 文です。Switch-case 文であってもこの If 文で書き換えることができるので、実質これ 1 つ押さえておけば後でなんとでもなります。

さて、分岐とは条件式が成立するか否かなどにより、処理を分岐させることを言います。では、条件式とはなにか、それについてまずはじめに説明します。

１．１条件式

条件式とは、以下のような記号で結ばれた変数や数値で構成される式のことを言います。

記号	使い方の例	意味
＞	a>b	aよりbが小さいならば成立
＜	a<b	aよりbが大きいならば成立
==	a==b	aとbが同じならば成立
>=	a>=b	aはb以上ならば成立
<=	a<=b	aはb以下ならば成立
!=	a!=b	aとbは異なるならば成立
|	A|B	条件式Aまたは条件式Bのどちらかが成立しているならば成立
&	A&B	条件式Aが成立し、かつ条件式Bも成立しているならば成立

赤色の記号は、青色の記号よりも処理が優先されます。計算順序の×÷と＋－の関係と同じように、青よりも赤が先に検討されます。上の表を見ながら、課題をといてみましょう。

課題１．１

次の条件式が成立か不成立かを答えなさい。それだけではわからないものはわからないと答えよ。

(1) 3<5

(2) 3>5

(3) 4<=9

(4) 8==0

(5) 9!=6

(6) (9<10)|(8>10)

(7) (9<10)&(8>10)

(8) ((a/3)==(a¥3))&((a/4)!=(a¥4))

①変数 a が 12 のとき
②変数 a が 15 のとき
③変数 a が 7 のとき
④変数 a が 4 のとき

(9) a==b が成立しない変数 a,b があるとき、a!=b

(10) a<=b が成立しない変数 a,b があるときの a>b

(11) a<b が成立しない時の変数 a,b があるときの a>b

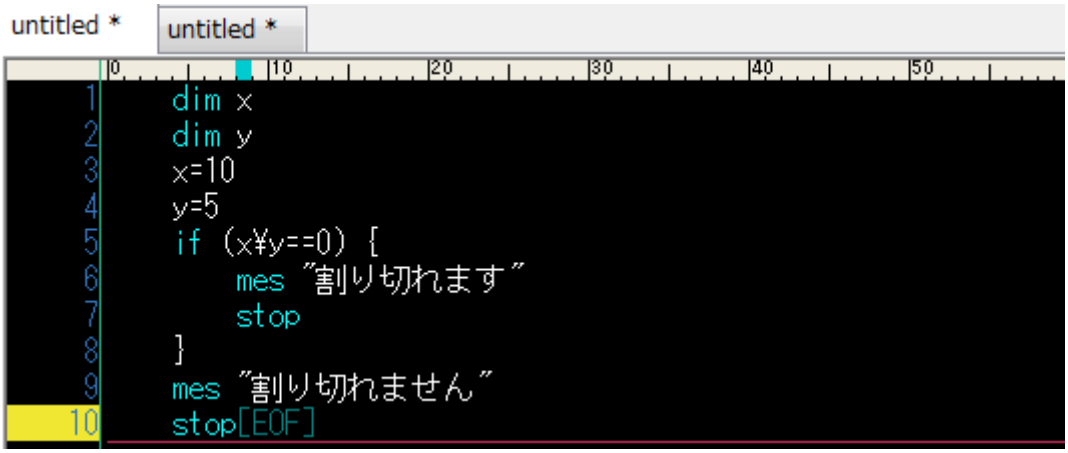
(12) a<b が成立しない時の変数 a,b があるときの a>=b

(13) a<b と a>b のどちらも成立しない変数 a,b があるときの a!=b

(14) a<=b と a>=b のどちらも成立する変数 a,b があるときの a==b

１． 2 if 文

さて、条件式がどういったときに成立するかを練習したので、分岐について説明します。実際に if 文を使ったプログラムをまずは見てみましょう。

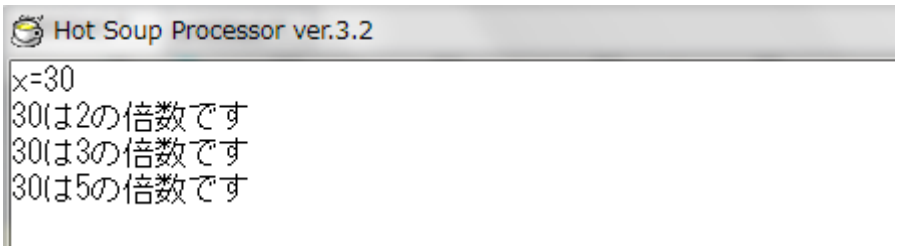


If 文の使い方ですが、if(条件式){ 処理 }の形で書き、条件式が成立した場合、{}で囲った処理を行わせることができます。今回は、x を y で割った余りが 0、つまり割り切れるときは「割り切れます」と表示し、動作停止、そうでなければ if 文の{}でくくった内容は飛ばされ、「割り切れません」と表示して動作を停止します。また、ループと同じく if の｛｝でくくった部分は中身が 1 行で済む場合を除き、Tab を一つ分あけるようにしましょう。1 行で済む場合は、if(条件式) : ~ のように、:を使うことで（）を省略することもできます。

課題１． ２

次のような処理を行うプログラムを組みなさい。

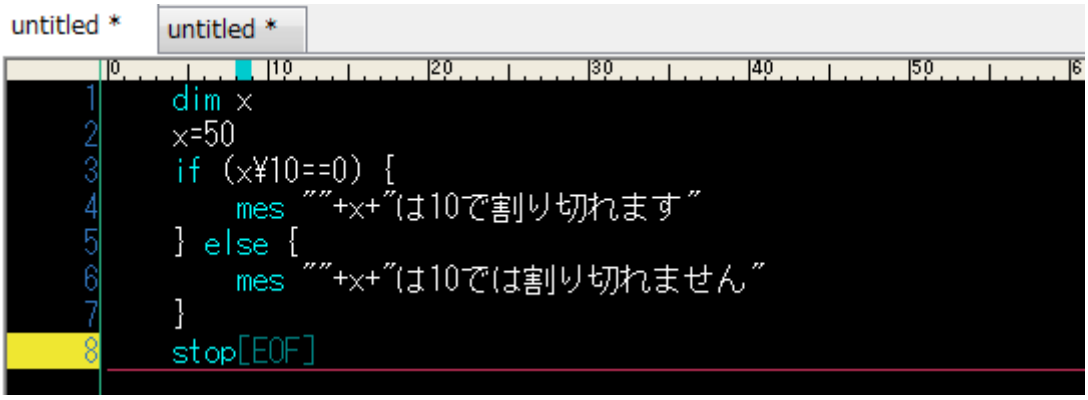
整数型変数ｘをつくり、値を何か代入する。その値が２の倍数、３の倍数、５の倍数、ならばそれぞれ「～の倍数である」と表示する。ｘ＝３０のときの実行結果を以下に示す



なお、ｘがいかなる値であっても正しく動作することを確認すること。

１． 3 if~else 文

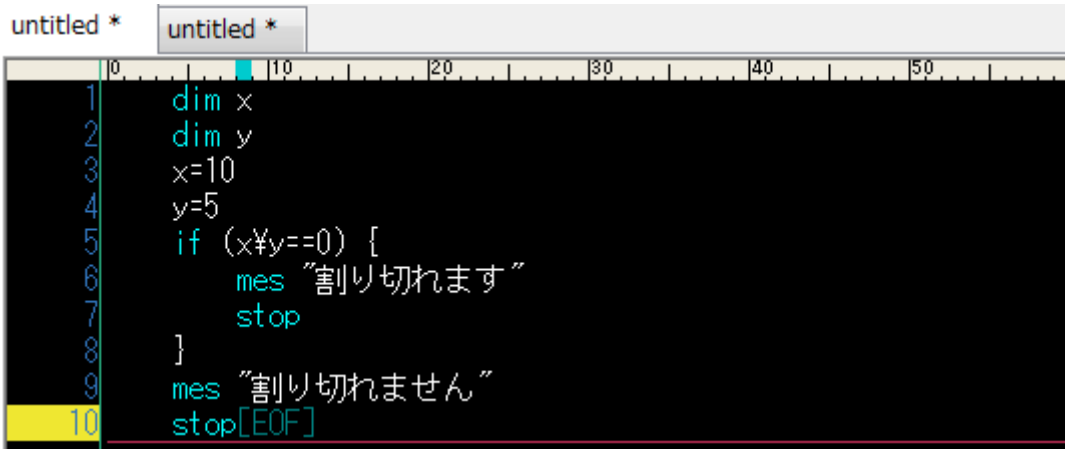
さて、if 文を使用する際、補助的に使える命令があり、それが else 命令です。



If 文の｛｝の後に続けて else {処理}と書くことで、if 文が成立しなかった場合に行う処理を指定できます。つまり、今回場合ならば、もしも x を 10 で割った余りが 0 ならば if に続く｛｝内の処理を、そうでなければ else に続く｛｝の中の処理を行います。このように、if 文が成立しない場合に、といった場合は else 命令を使いましょう。また、if 文同様、else で分岐されるプログラムは Tab を一つあけましょう。

課題１．３

先ほど例に挙げた下のようなプログラムを if-else 文で書き直しなさい。



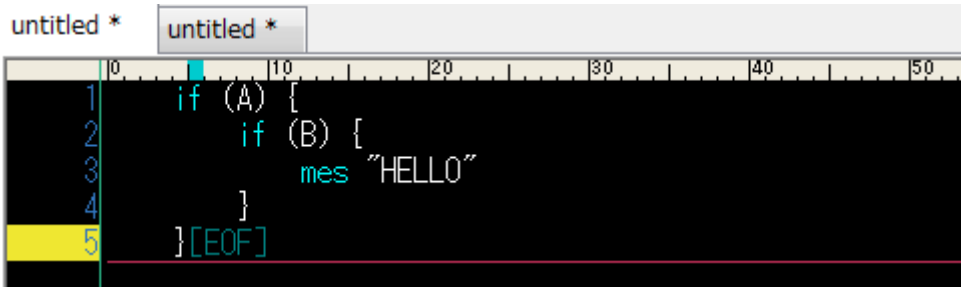
課題１．４

整数型変数 x、ｙの二つを入力し、ｘの方が大きい、ｙのほうが大きい、どちらも同じ、の３つのうちいずれかを判断し、結果を表示するプログラムを作りなさい。

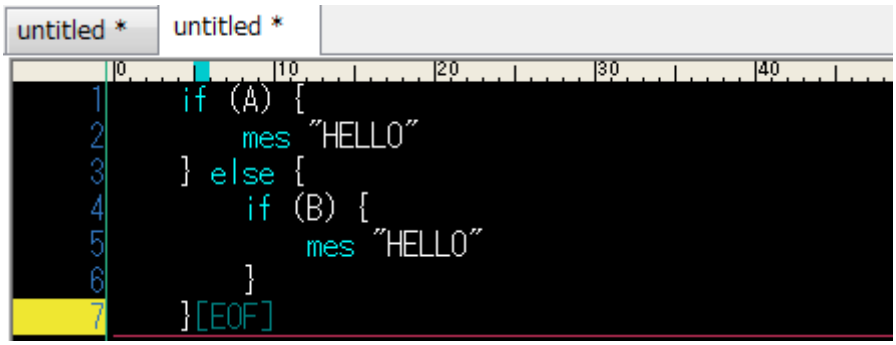
課題１．５

条件式Ａと条件式Ｂがあるとき、次のプログラムを if を使うのが１回だけで済むように簡略化しなさい。（ヒント…&や｜の意味はどうだったか。また、条件式Ａと Ｂはたとえばｘ==0 やｘ！＝０などが入るが、どんな条件式でも成り立つため、仮にＡとＢとおいている）

①



②

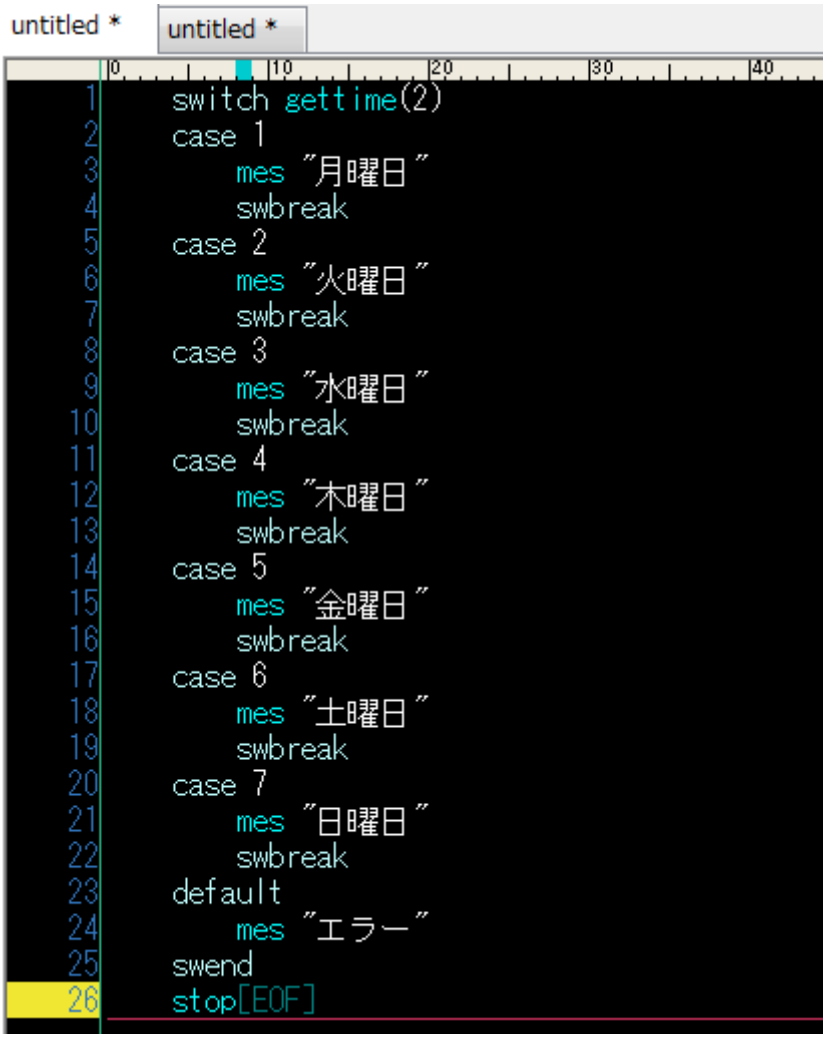


課題１． ６

変数 x を指定し、それが ２ の倍数か３の倍数か、それとも６の倍数か、はたまたそのどれでもないのかを表示するプログラムを作りなさい。なお、６の倍数ならば２の倍数である、もしくは３の倍数である、と表示してはいけない。

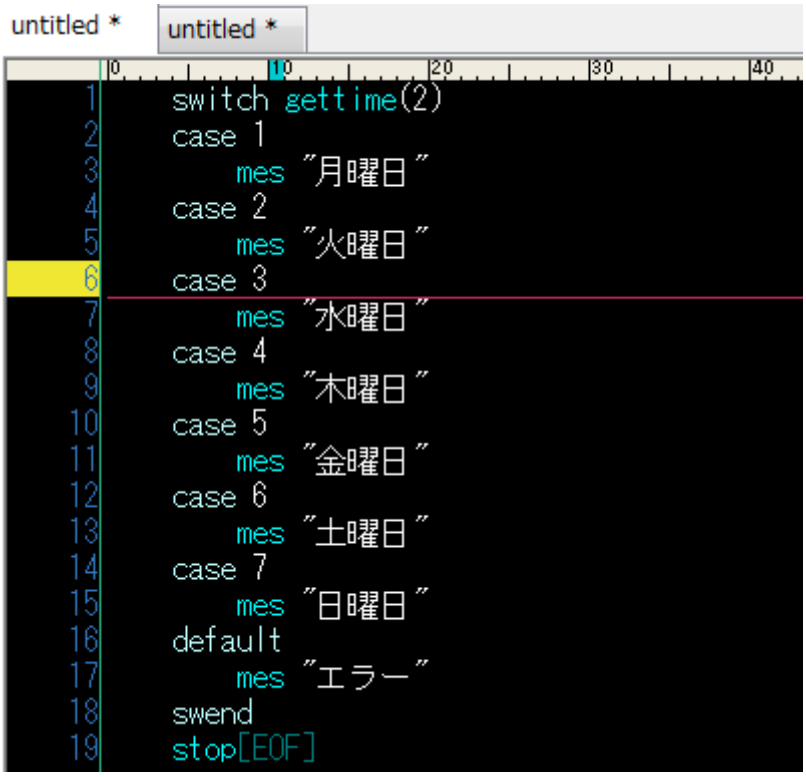
２．Switch-case 文

If 文以外にも分岐はあり、それがこの switch-case 文です。こちらは、指定した変数の値によって処理を変化させるので、if 文と違い汎用性はありませんが、if 文と比べて見やすいため、キーの入力状況を調べるときや、何らかの状況を返す関数や、現在の処理状態を示す変数を扱う場合などには積極的に使っていくとよいでしょう。では、実際のプログラムを見てみます。



Switch-case 文は switch で始まり swend で分岐を終了します。まず switch の後に変数や値を返す関数を置き、その値によって処理を変化させるかを指定します。今回は gettime(2)、つまり何曜日かを返す関数を指定しました。この値によって処理を変化させます。次に switch で指定した変数（関数）と case で指定した値が一致するところまでジャンプします。たとえば、実行したのが木曜日だとしたら、gettime(2)には「４」が入っているので case ４の行、つまり１１行目までジャンプします。そして、そこから処理を再開し、mes 命令で「木曜日」と表示した後、swbreak 命令が出てきます。これは、「switch 命令による分岐から抜ける」という意味なので、switch 命令の終わりを意味する swend までジャンプします。これで、プログラム自体の処理は stop により一時停止し、終了します。月曜、火曜などでも同じように、switch で指定した変数（関数）と一致する数値を指定している case までジャンプし、処理を再開します。Switch-case 文の最後のほうに default とありますが、これは if 文の else と同じような使い方ができます。

これが、switch-case 文の基本的な使い方なのですが、実は if 文だと若干難しくなるが switch-case 文だと簡単に再現できるとっておきの裏技があります。



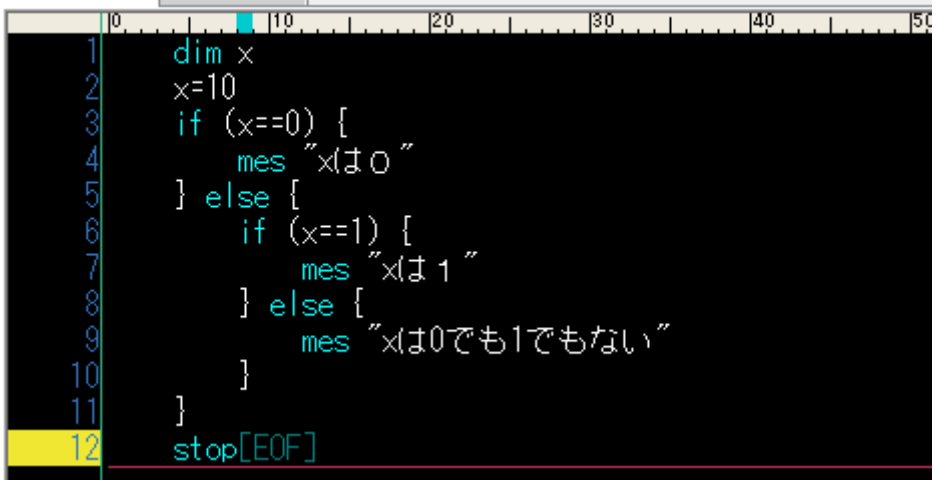
はい、先ほどのプログラムから swbreak をすべて取り除いてみました。これによってどう処理が変化するでしょう。先ほどと同じように実行日が木曜日だと仮定しましょう。木曜日なので gettime(2)の値は 4 になり、case 4 の行までジャンプし、mes 命令で「木曜日」と表示します。ここまでは同じです、さて、次の行に行くと case 5 とあります。case 命令は switch 命令におけるジャンプ先の目印を示しているため、ラベル同様、その行は何の処理も行われません。ですので、次の行に行き「金曜日」と表示しさらに次に行き…と、switch 文の最後の swend までの処理をすべて行います。つまり、swbreak を省略すると以降の処理は swbreak か swend が出るまですべて行われることになります。この特徴は、覚えておくと役に立つかもしれません。

課題２． 1

次の if 文で表記されたプログラムを switch-case 文で書き直しなさい。

untitled *

untitled *



```
dim x
x=10
if (x==0) {
    mes "xは０"
} else {
    if (x==1) {
        mes "xは１"
    } else {
        mes "xは0でも1でもない"
    }
}
stop[EOF]
```