

マイクラフトで プログラミング

第13回 障害物をよけて進ませよう

目標

- 「パスワードもどき」を作ってみよう
- ブロックがあればよけて進むようにしよう

場合によって違う命令をしたい

今回は場合や状況によって違う
命令を実行することを考えよう

例えば



鉄だけほるようにしたいとき、ブロックが鉄のときと鉄以外のときで命令は違うはず

if文

```
1   a = "ohayou"  
2   if a == "ohayou" then  
3       print("ohayou gozai masu")  
4   end  
5   ifから始まるからif文
```

条件によって命令を変えるには
if(イフ)文を使うよ!

if文の成り立ち

```
if ここに条件を書きます then
  ここに命令を書きます
end
```

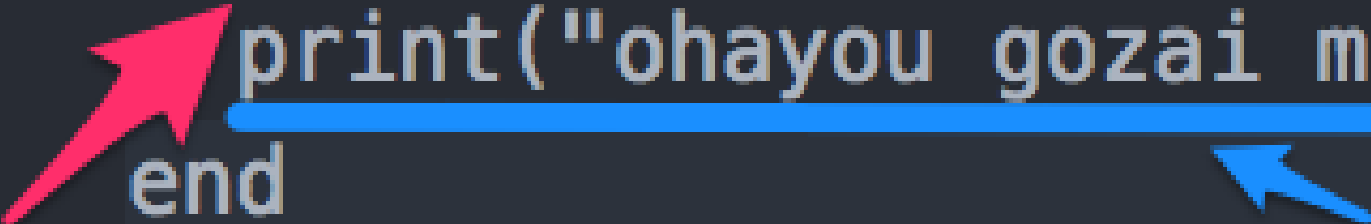
The diagram illustrates the structure of an if statement. It consists of three main parts: the start section (開始部), the command section (命令部), and the end section (終了部). The start section is marked by the 'if' keyword and the condition 'ここに条件を書きます' (Write the condition here), followed by the 'then' keyword. The command section is marked by the command 'ここに命令を書きます' (Write the command here). The end section is marked by the 'end' keyword. Arrows point from the labels to their respective parts: a red arrow from '開始部' to 'if', a blue arrow from '命令部(何行でもOK)' to the command line, and a green arrow from '終了部' to 'end'.

if文は開始部(条件部)と命令部と
終了部の3つでできているよ

if文の動き

```
1  a = "ohayou"  
2  if a == "ohayou" then  
3      print("ohayou gozai masu")  
4  end
```

5 変数aの中身が"ohayou"なら print命令を実行



if文は条件部の条件を満たしているときだけ
命令部に書いた命令を実行するよ

条件式

```
if ここに条件を書きます then  
  ここに命令を書きます  
end
```

ここに条件を書いていくよ

if文には条件式が必要
条件式では特別な記号を使って数を比べたりするよ

【条件式】 数や文字が同じかどうか

```
if a == "ABC" then  
  print("ABC desu")  
end
```

変数aの中身が文字のABCと
同じかどうか比較している

2つの数や文字が同じかどうかは **==** で
確かめることが出来るよ

課題1 パスワードもどきを作る

```
> e13-1  
TechChance  
YES  
>
```


```
> e13-1  
Tech  
> _
```

io.read命令で文字を入力して，入力した文字が
TechChanceの時だけ「YES」と表示しよう

else

```
if a == "ABC" then
    print("ABC desu")
else
    print("ABC janai yo")
end
```

変数aの中身が文字ABC以外するとき
elseより後のprint命令を実行する



else(エルス)を使うとifの条件式を満たすとき
以外の命令を決めることが出来るよ

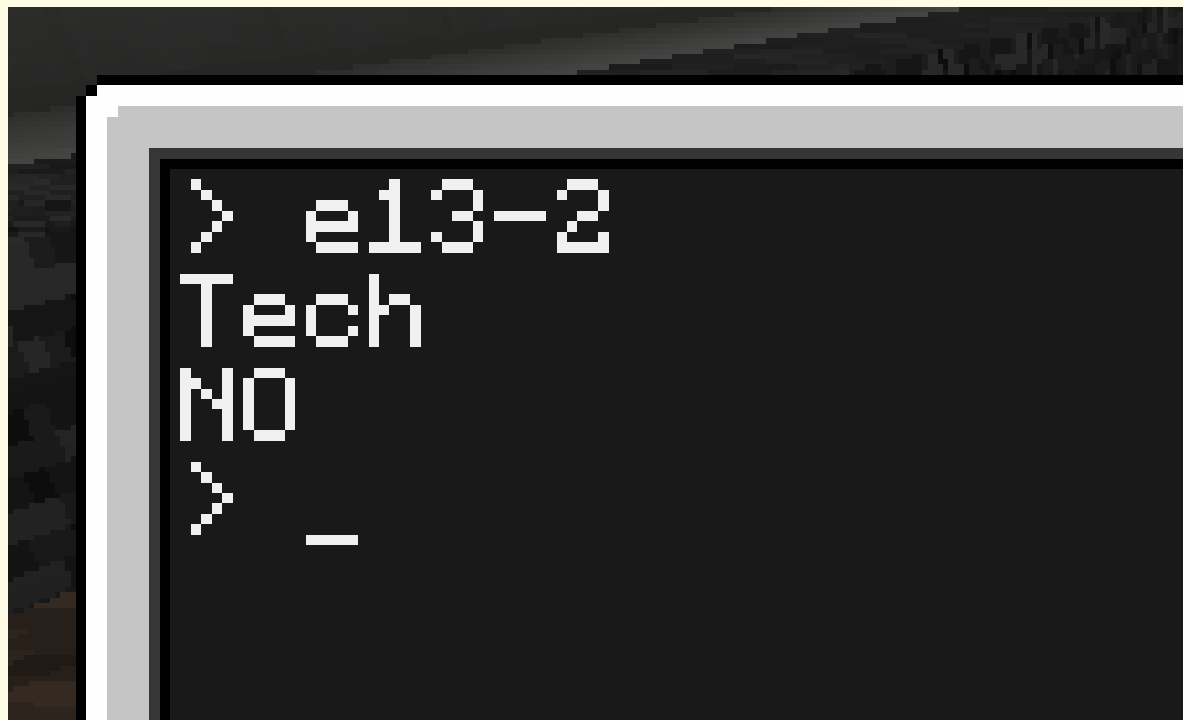
else

```
if a == "ABC" then  
    print("ABC desu")  
end  
else  
    print("ABC janai yo")  
end
```

ifとendの外にelseを書いているからこれはダメ!

elseは必ずifとendの間に書かれなければならないよ

課題2 パスワードもどきの改良



```
> e13-2
Tech
NO
> _
```

課題1を改良して, `else`を使ってTechChance
以外の時は「**NO**」と表示できるようにしよう

trueとfalse

```
> test1  
true  
> _
```

ブロックがあるとき

```
> test1  
false  
> _
```

ブロックがないとき

turtle.detect命令でtrueとfalseが出てきたね
これも判定に使うことができるよ

課題3 turtle.detectの復習

```
> test1  
true  
>  
_
```

ブロックがあるとき

```
> test1  
false  
>  
_
```

ブロックがないとき

`turtle.detect`命令で目の前にブロックがあるかどうかをtrueとfalseで表示しよう

【重要】 trueとfalseをif文に使う

```
a = true
if a == true then
  print("true desu")
end

if a then
  print("true desu")
end
```

どちらも同じ意味だよ!

trueやfalseはそのものが条件式だよ
だから == がなくても使える

課題4 ブロックを調べる1

```
> e13-3  
arimasu  
> _
```

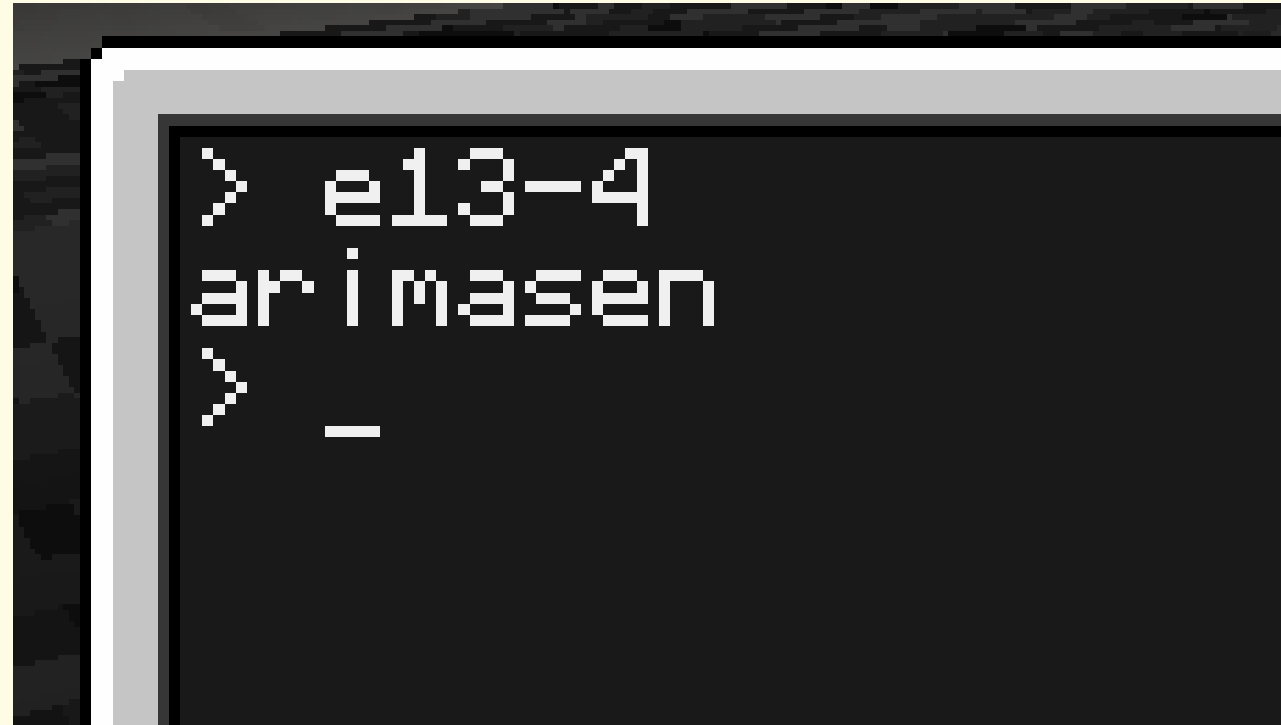
ブロックがあるとき

```
> e13-3  
>
```

ブロックがないとき

`turtle.detect`命令を使って目の前のブロックを調べて、`true`の時は「`arimasu`」と表示しよう

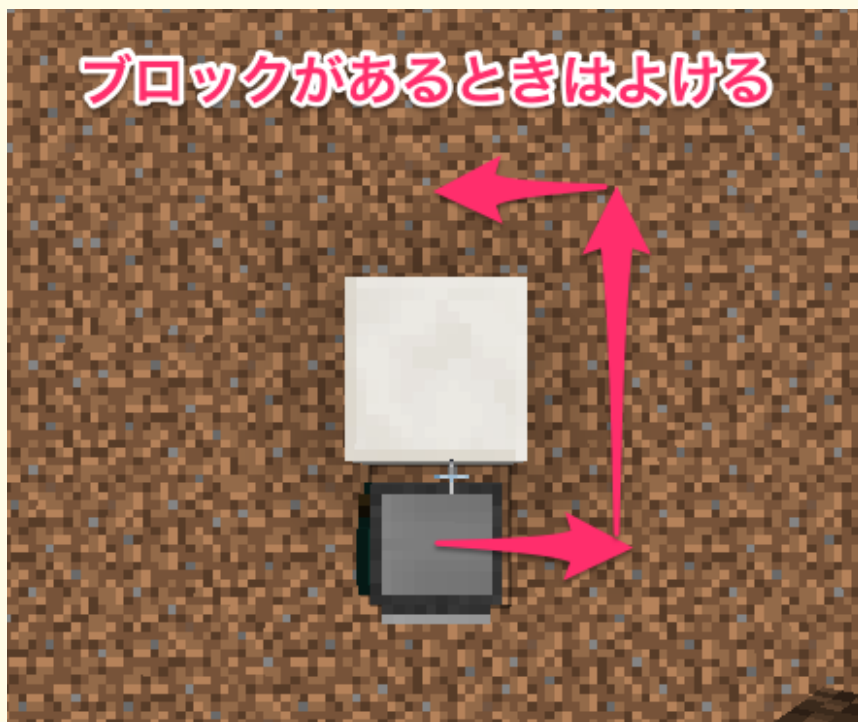
課題5 ブロックを調べる2



```
> e13-4
ar imasen
> _
```

課題3を改良して，目の前のブロックがない時は
「ar imasen」と表示しよう

課題6 ブロックをよけて進む



ブロックがあるときは左か右によけてすすんで、
ないときはまっすぐ進むプログラムを書こう

ブロックをよけたかな？



課題6では1マスのブロックはよけることができる
でも2マスのブロックはできない

課題7 もっとよける



ブロックを2マスまでよけるように
課題6を改良しよう