

マイクラフトで プログラミング

第6回 復習 + 残りねん料やスロットをしゃべらせよう

今日の目標

- 残りのねん料をしゃべらせよう！
- 目の前にブロックがあるかどうか調べよう！

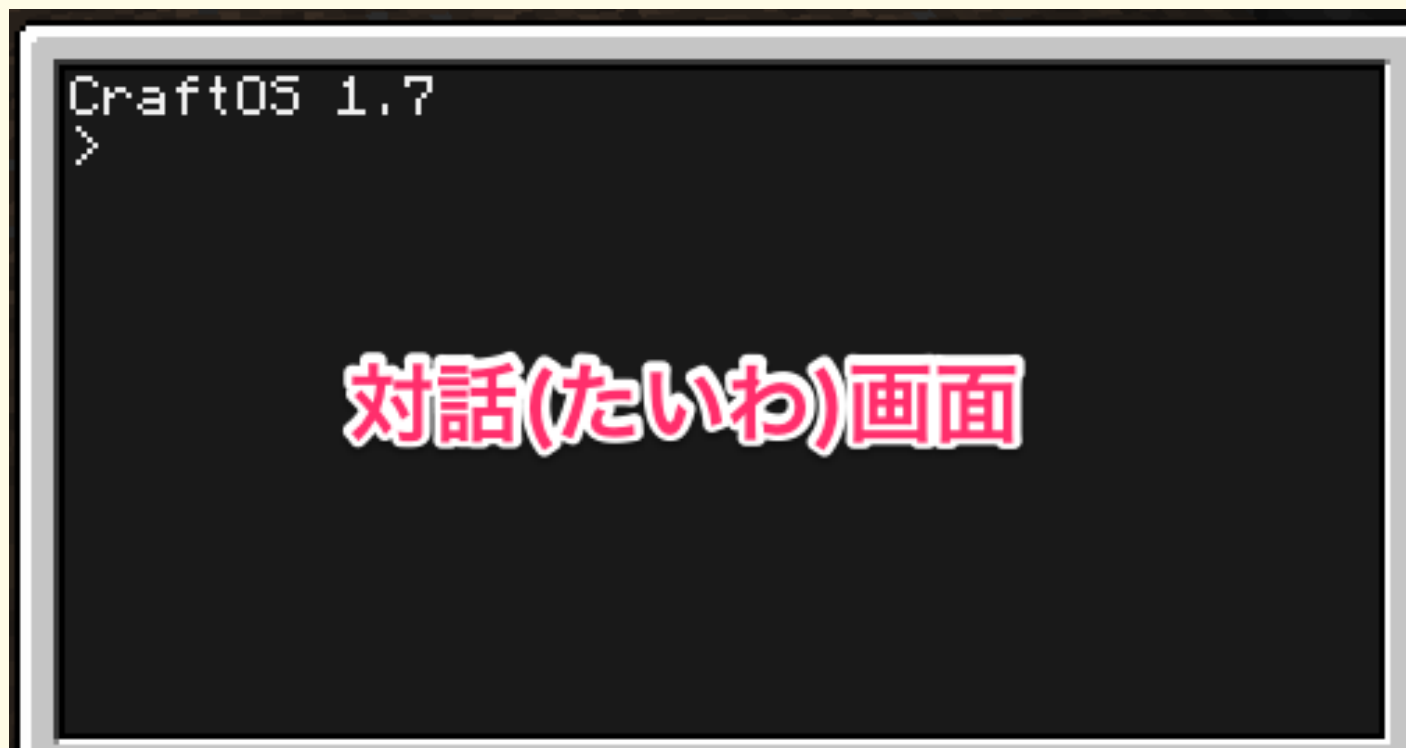
復習をしよう!

今日はまず今まで習ったことの
復習をしていこう!

何を習った？

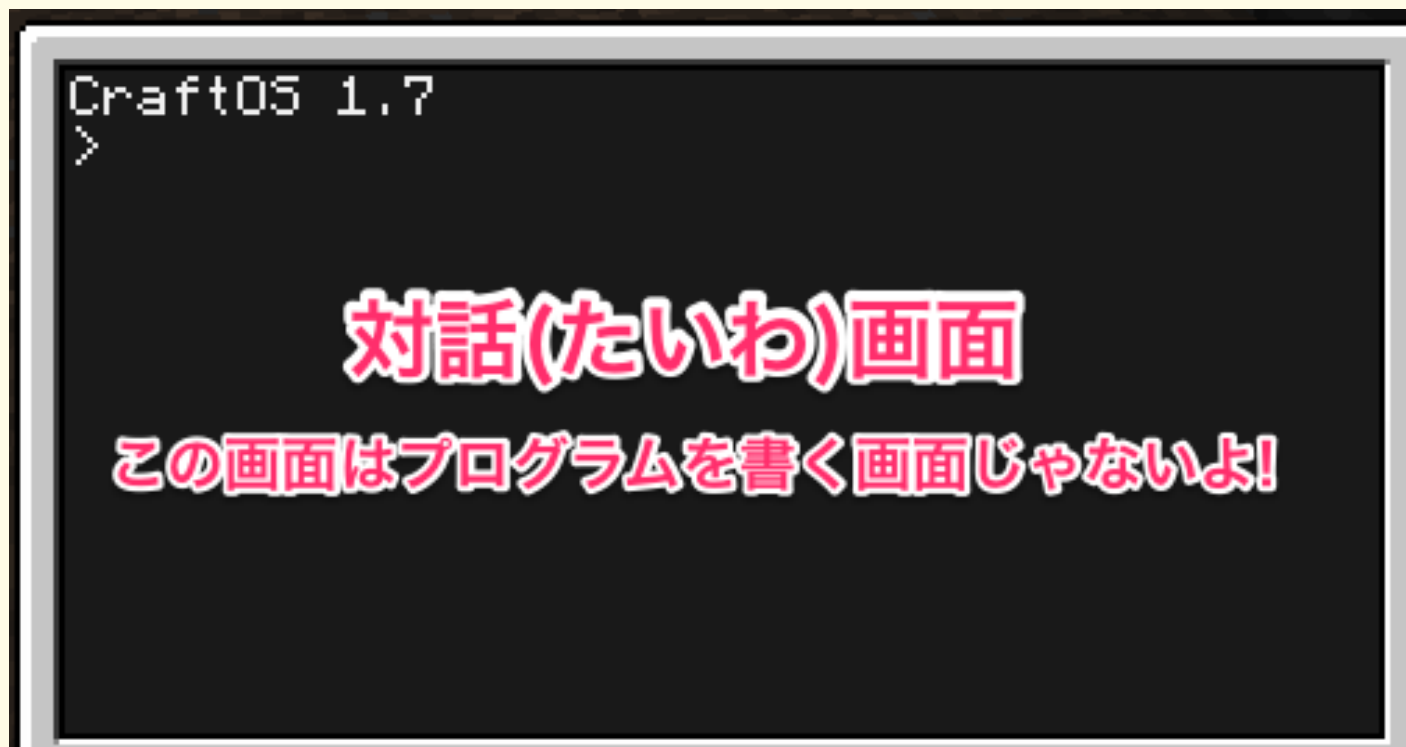
- 対話画面とエディタ画面の違い
- プログラムを動かす方法
- 数字や文字の入る箱「変数(へんすう)」
- いろんな命令

対話(たいわ)画面



ここでタートルとお話するよ
プログラムを実行するのも対話画面だよ

対話(たいわ)画面



ただし、対話画面では
プログラムは書けないよ!

プログラムを編集(へんしゅう)するコマンド



```
CraftOS 1.7  
> edit kadai
```

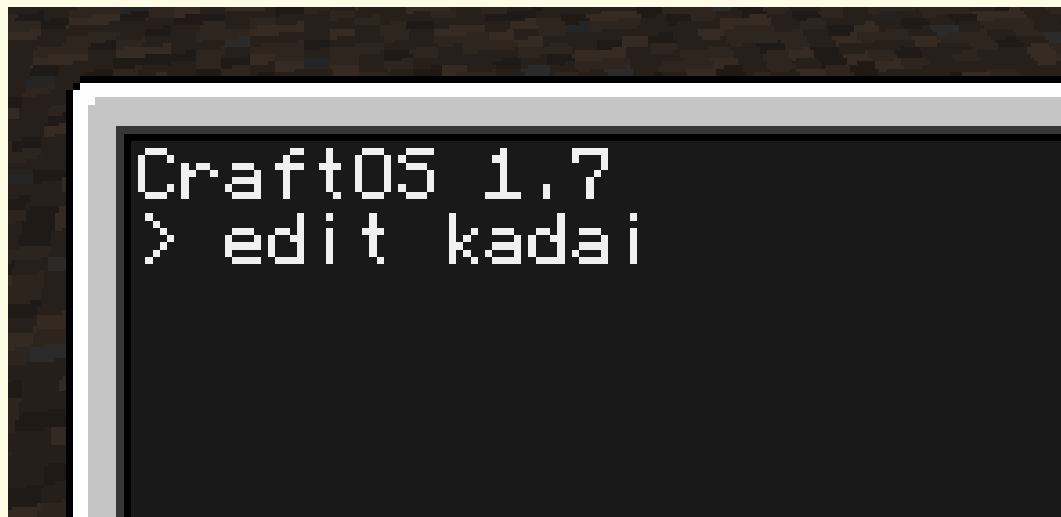
edit プログラム名 で
プログラムを編集できるよ

エディタ画面



edit プログラム名 でエディタ画面になるよ
ここでプログラムを書くよ

editコマンドの動き



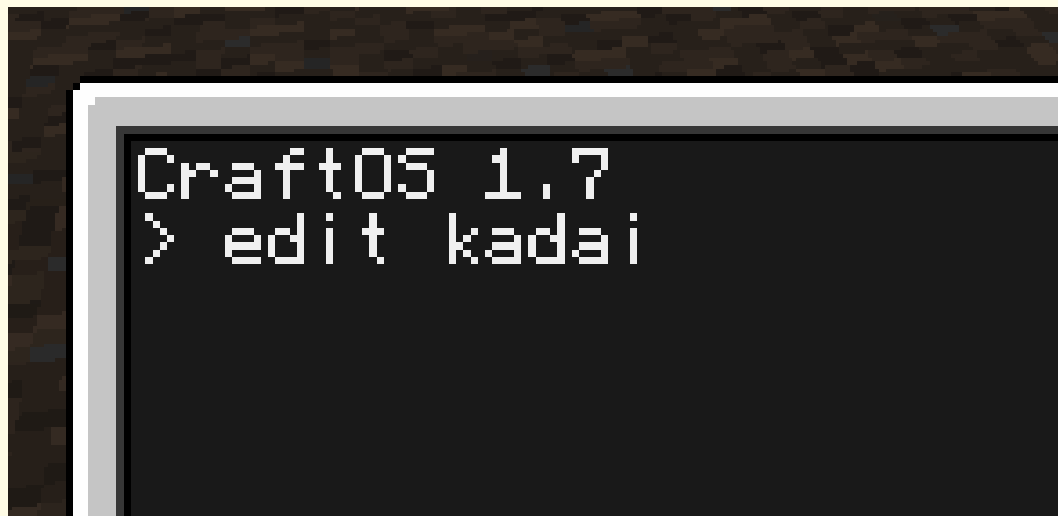
```
CraftOS 1.7  
> edit kadai
```



```
Press Ctrl to access menu Ln 1
```

なければその名前で
新しいプログラムファイルを作るよ

editコマンドの動き



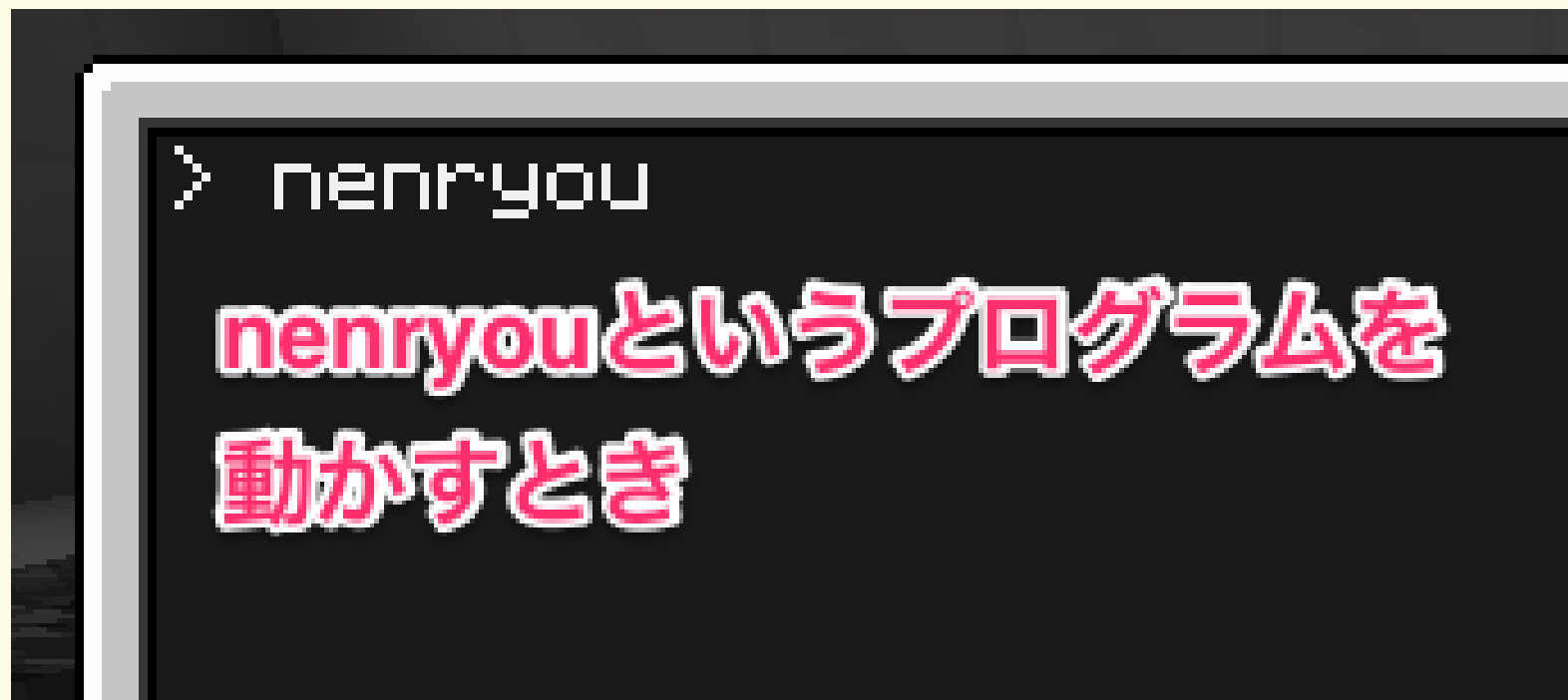
```
CraftOS 1.7  
> edit kadai
```



```
turtle.forward()  
turtle.turnLeft()  
  
Press Ctrl to access menu Ln 1
```

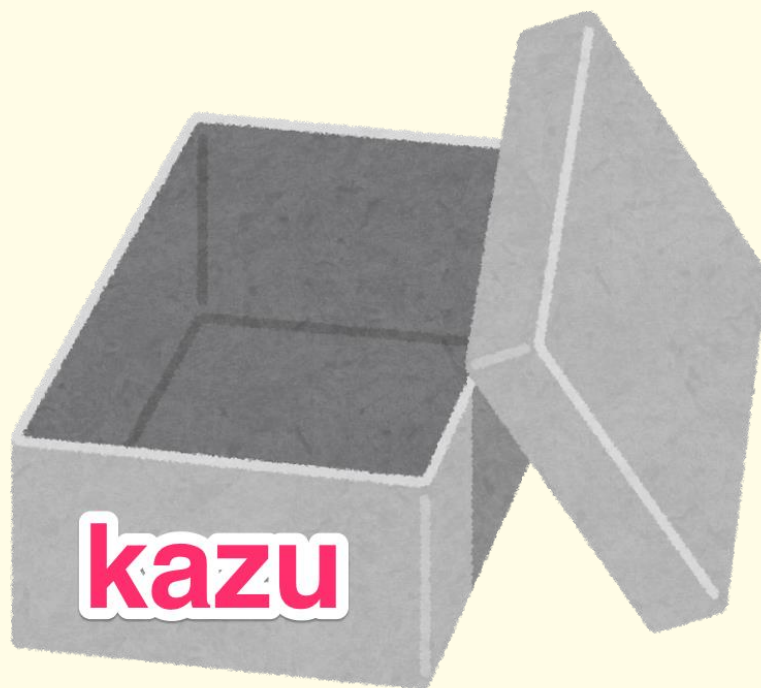
もうすでにプログラムがあれば
それを編集(へんしゅう)するよ

プログラムを動かす方法



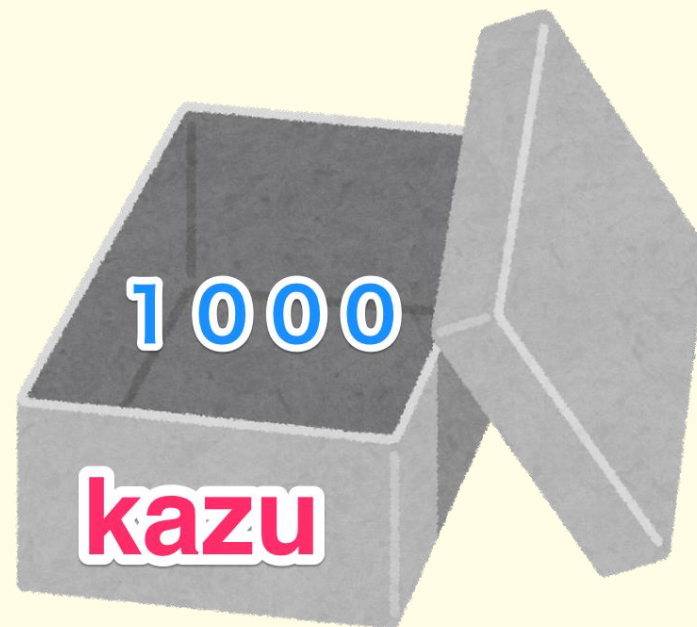
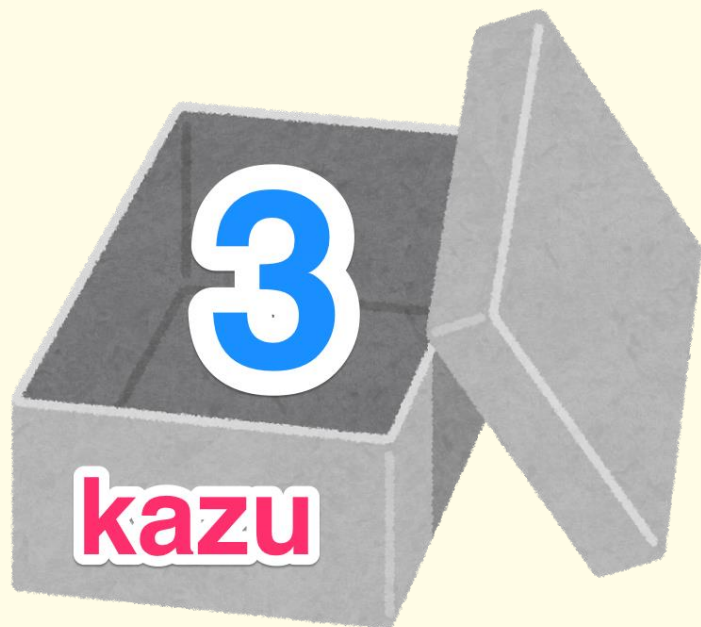
対話画面でプログラム名をうてば実行されるよ!

数字の入る箱



数字が入る名前のついた箱を想像してみよう
たとえばkazuっていう箱

箱のなかの数は変えられる



入ってる数は自由に変えられるとしよう

プログラムで書くと…

数の入る箱

```
kazu = 10
```

さっき考えた数の入る箱を
プログラムで書いてみるとこうなる

変数

```
kazu = 10
```

kazuっていう変数

ここではkazuに10を入れている

これ(kazu)を**変数**っていうよ

変数 = **数**で変数に数を入れることができるよ!

変数の名前

文字から始まるのはOK

```
ichiban = 1
```

```
1ban = 1
```

数字から始まっているのはダメ

変数の名前は自由につけられる
ただし、**頭文字は数字じゃダメ**

変数



変数に = で数字や文字を入れることができるよ
でも、このイコールは算数や数学の = と違うよ!

変数を使う

```
slot = 10  
turtle.select(slot)_
```

slot変数をselect命令に使っている

例えばスロットの選択や ねん料の数に
変数を使えるよ！

いろいろな命令

これまでにいろいろな命令を習ったね
命令リストを見て思い出してみよう！

さっそく

復習はコレで終わり！これから今日の目標

- 残りのねん料をしゃべらせよう
- 目の前にブロックがあるかどうか調べよう

をやっ払いこう！

タートルにしゃべってもらおう



こんにちは!



タートルにしゃべってもらう

```
print(30)  
print("hello")
```

```
> say  
30  
hello  
>
```

`print(しゃべらせたい内容)`でタートルに
しゃべらせることができるよ

なんでもしゃべってくれる

```
print("konbanwa")
```

文字のときは最初と最後に
ダブルクォーテーションをつけよう

文字の時は” を最初と最後につけないとダメ
数字や変数の時は “ を付けなくてOK

変数の中身もしゃべってくれる

```
a = "ohayou"  
print(a)_
```

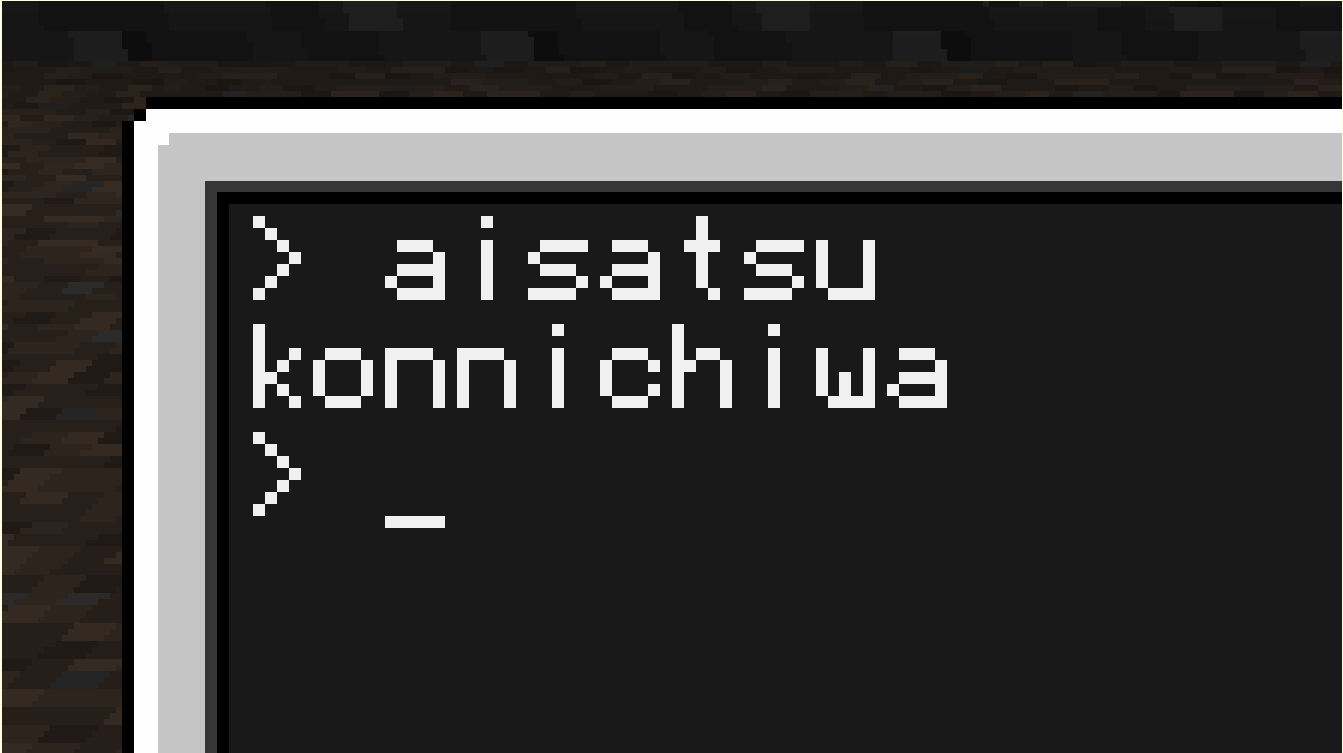
変数をしゃべらせたいときは
かっこの中にその変数を書く

```
> hensuu  
ohayou  
> _
```

変数aの中身を
しゃべってくれた

命令のかっこの中に変数を入れれば
変数の中身をしゃべってくれるよ

課題1



```
> aisatsu  
konnichiwa  
> _
```

`print`命令を使って
konnichiwa としゃべらせよう!

課題2

```
myHensu = "computer craft!"
```

好きな変数をつくって
中身をしゃべらせよう!

タートルに話を聞いてもらう



こんにちは!



タートルに話を聞いてもらう

```
kaiwa = io.read()
```

`io.read`命令でタートルが
話をきいてくれるよ

聞いた内容を変数に入れる

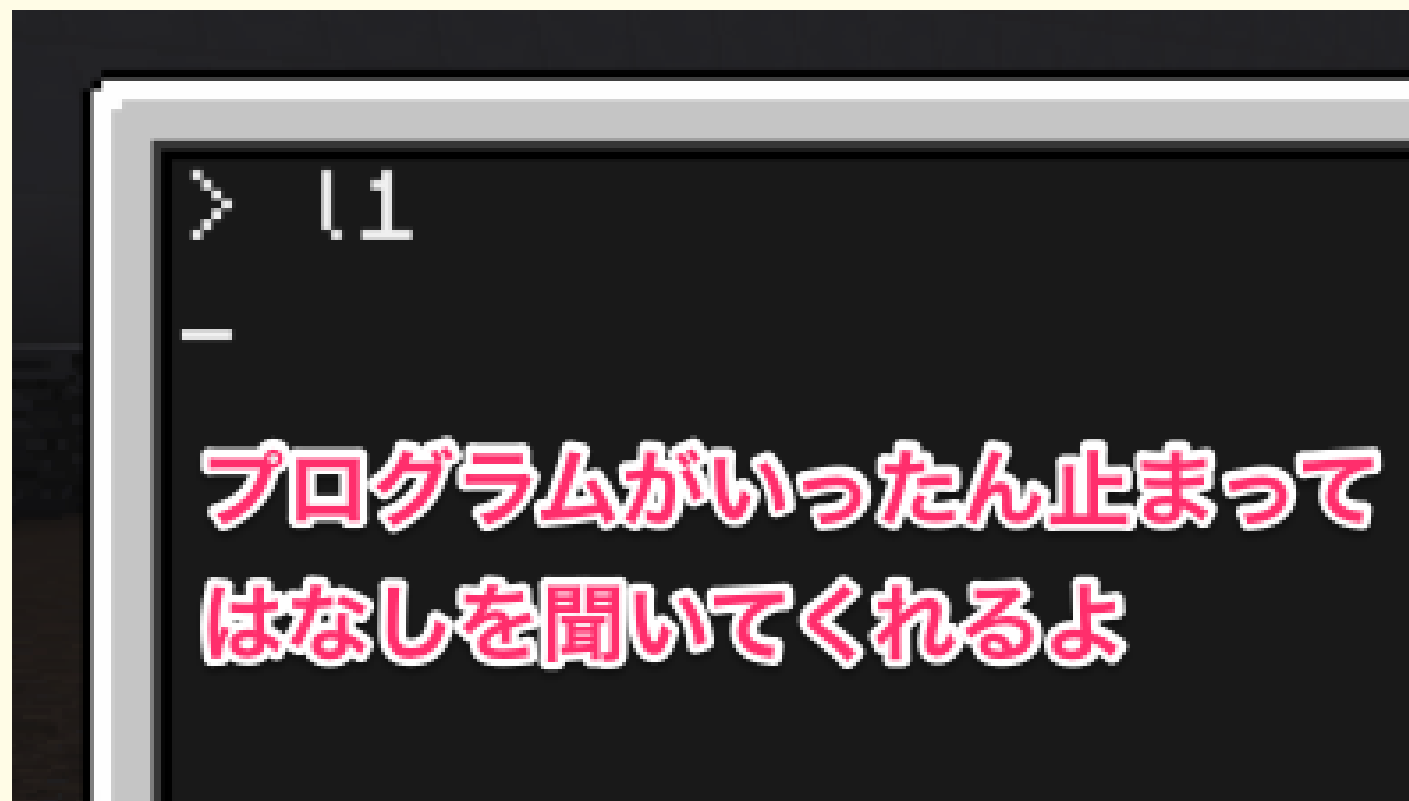
```
kaiwa = io.read()
```

kaiwa変数に

話しかけた文字や数字が入る

変数 = **io.read**命令で聞いた内容を
変数に入れることができるよ

キーボードで話をする



キーボードで文字や数字を打ち込んで
お話できるよ

課題3

```
> l1  
hello_
```

```
> l1  
hello  
hello  
>
```

`io.read`命令を使ってタートルに
しゃべった内容をそのまま`print`命令でしゃべらせよう

残りねん料をしゃべらせる



残りのねん料は?

1000



残りねん料を教えてくれる命令

```
fuel = turtle.getFuelLevel()
```

`turtle.getFuelLevel()` 命令とイコールで
残りねん料を変数に入れることができる

課題4

A terminal window with a dark background and white text. The prompt is a greater-than sign (>). The command entered is 'nokori'. The output is '20000'.

```
> nokori  
20000  
>
```

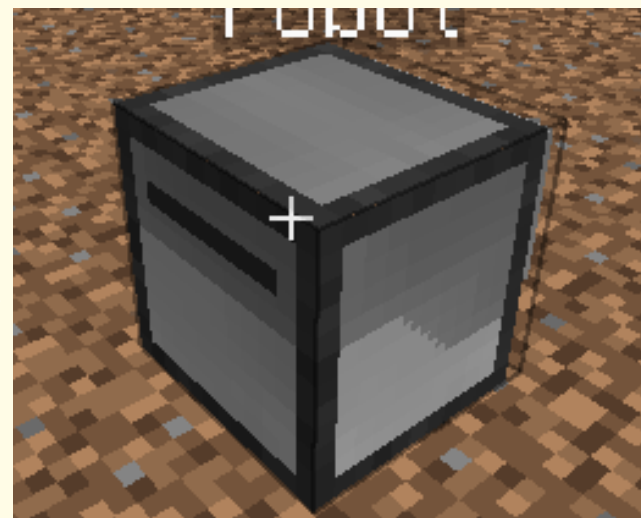
turtle.getFuelLevel命令とprint命令
を使って, 残りねん料をしゃべらせよう

目の前にブロックがあるかしゃべらせる



目の前にブロック
がありますか？

いいえ



目の前にブロックがあるかを調べる

```
block = turtle.detect()
```

`turtle.detect`命令でタートルの目の前に
ブロックがあるかを調べることができるよ

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

目の前にブロックがあるとtrue(トゥルー)
ブロックがないとfalse(フォルス)と教えてくれる

課題5

```
> block  
false  
)  
_
```

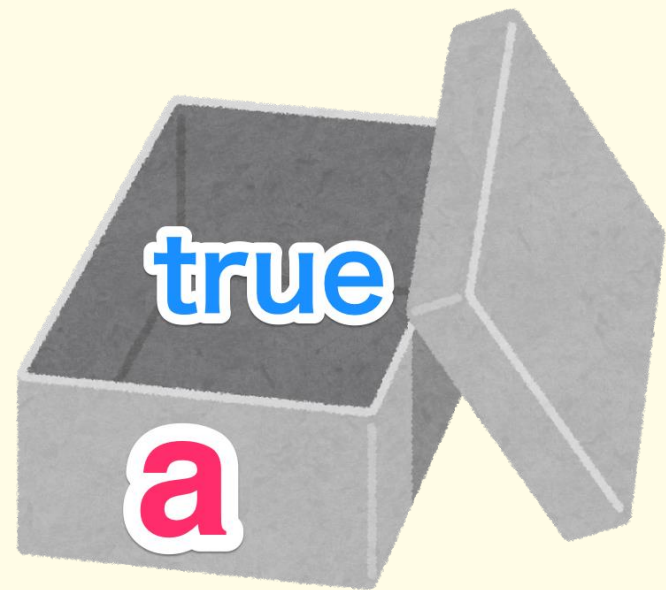
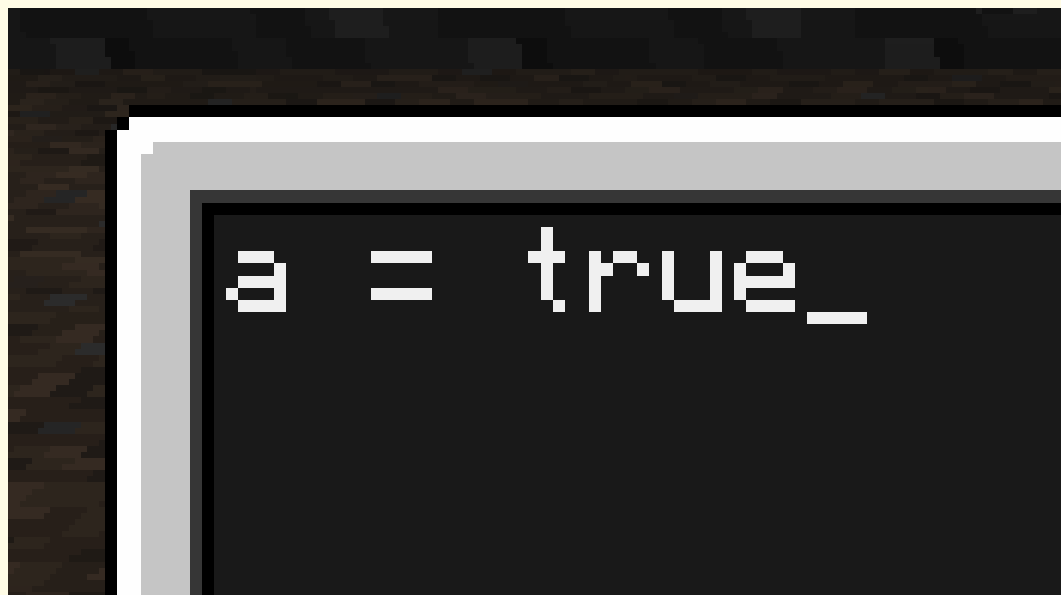
turtle.detect命令とprint命令を使って
目の前にブロックがあるかしゃべらせよう

true と false

trueはyesやはいという意味

falseはnoやいいえという意味

trueとfalseも変数に入る



trueとfalseも **変数に入れることができる**よ
今はあまり使わないけど覚えておこう!

どう使う？



detect命令で目の前にブロックがあるか判別して
ブロックをよけて進むときに使える！

もっとていねいにしゃべらせる



残りのねん料は？

1000



もっとていねいにしゃべらせる



残りのねん料は？

のこりねんりょう
は1000です



文字と変数を組み合わせる

```
namae = "murao"  
print( "namae wa " .. namae)
```

..(ドット2つ)でつなげる

```
> test  
namae wa murao  
> _
```

.. で文字と変数をつなぐことができるよ

文字と変数を組み合わせる

```
a = "ringo"  
b = "banana"  
print(a .. " to " .. b)
```

```
> tsunagu  
ringo to banana
```

**2つの変数と文字を
つなぐことができた**

.. をもっと使えばいくつでも
文字と変数をつなぐことができるよ

課題6

```
> teinei  
A wa 10 desu  
>
```

変数Aの中に10
が入っているとき

変数Aを作ってA wa 変数Aの数 desu
としゃべらせよう

課題7

```
> nokori  
nokori nenryou ha 20000 desu  
> _
```

課題3を改良してていねいにしゃべらせよう