

プログラミング (Python)

第3回 条件分岐 if 文

早稲田大学本庄高等学院 2020年度版

飯島涼



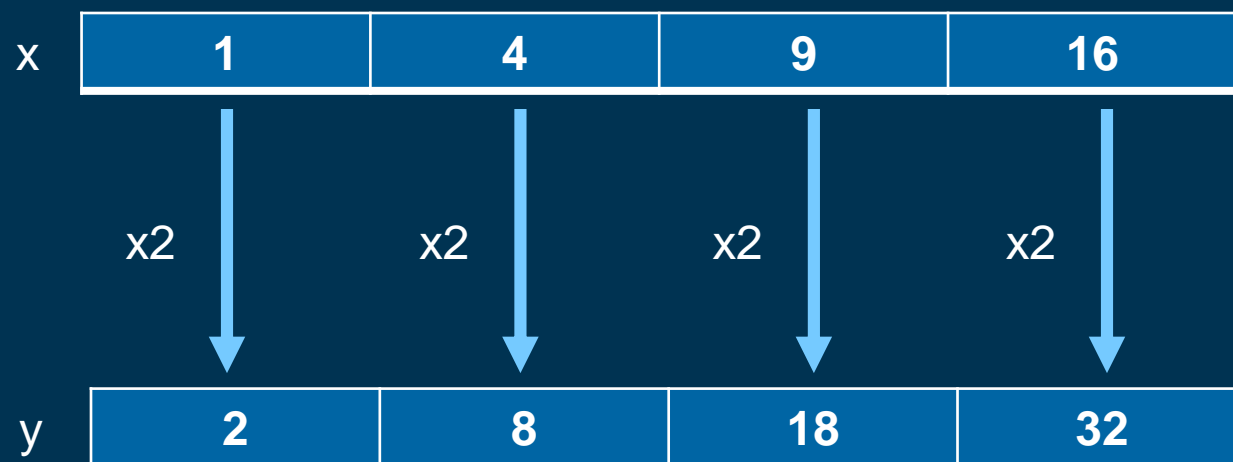
ブロードキャスト計算

- [ベクトル] * a の形で計算した場合、各ベクトルの要素と、aを掛け算した結果がはいったベクトルが作られる。

- 例

```
1 import numpy as np
2 # ベクトルのブロードキャスト計算
3
4
5 x = np.array([1, 4, 9, 16])
6
7 y = 2 * x # 数字*ベクトル の形
8 |
9 print(x)
10 print(y)
```

```
[ 1  4  9 16]
[ 2  8 18 32]
```

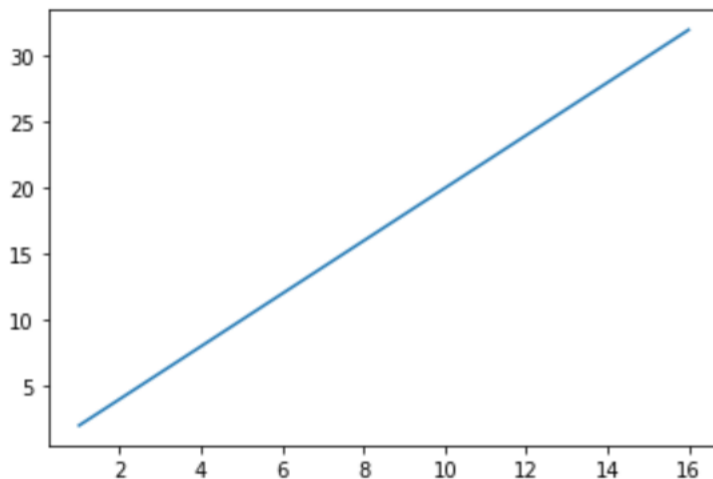


plot()

- plot()はそれぞれ対応する点に点を打ってつないでくれる

```
7 x = np.array([1, 4, 9, 16])
8
9 y = 2 * x # 数字*ベクトル の形
10
11 print(x)
12 print(y)
13 plt.plot(x, y)
14 plt.show()
```

```
[ 1  4  9 16]
[ 2  8 18 32]
```



復習

リスト・モジュール・グラフ

モジュール

- プログラムを便利に書くための機能が詰まっているフォルダ

import [モジュール名] でモジュールを呼び出す

import [モジュール名] as [モジュール名を省略した書き方(自分で好きに決めていい)]

例)

```
3 import matplotlib.pyplot as plt
4 import numpy as np
```

モジュールの使い方

[モジュール名].機能名()

- mathモジュール

math.sqrt(4) = $\sqrt{4}$

=> mathというモジュールの.sqrt(平方根)という機能を使います ということ

```
1 # math モジュール
2 import math
3
4 #平方根
5 x = math.sqrt(4) #mathというモジュールに入ったsqrtという機能を使いますということ
6 print(x)
```

numpy

- ベクトルや行列の計算をするためのモジュール

- `arange(1, 2, 0.1)`

- `range`のベクトル版

- `array([1, 3, 5, 4])`

- リストをベクトルの計算に使えるようにする機能

```
1 # numpy モジュール
2 import numpy as np
3
4 x = np.arange(1, 2, 0.1)
5 print(x)
6 y = np.array([2, 3, 5, 1])
7 print(y)
```

```
[1.  1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9]
[2 3 5 1]
```

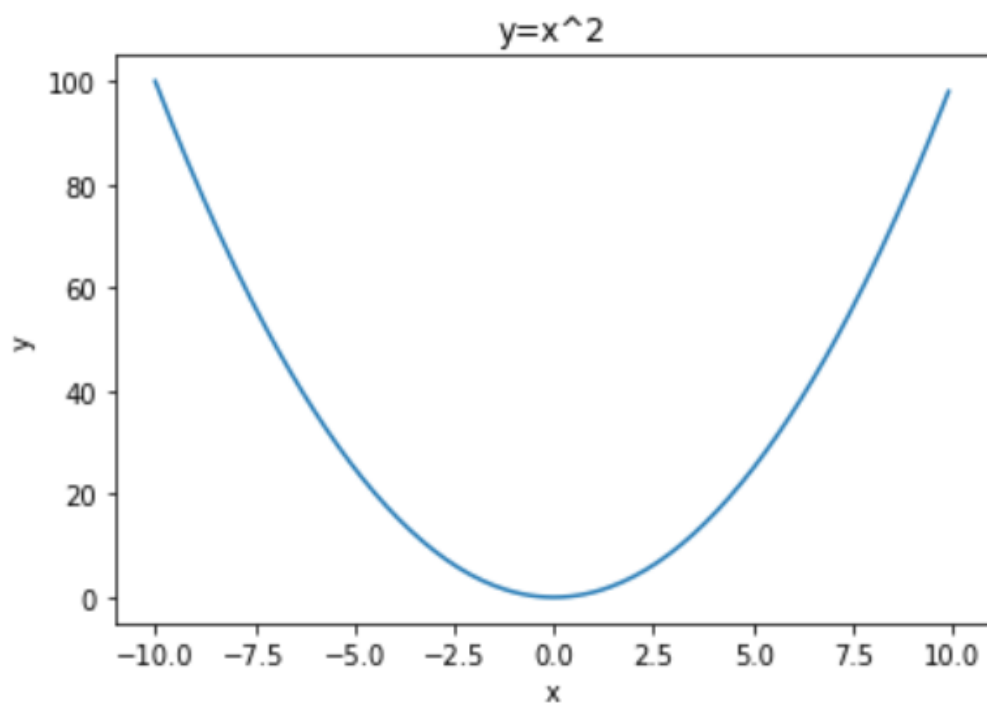
グラフの表示

- とりあえず使ってみる

```
1 # 数学のグラフ表示
2
3 import matplotlib.pyplot as plt
4 import numpy as np
5
6 x = np.arange(-10, 10, 0.1) # rangeと使い方は同じ, 少数の指定ができる
7 y = x**2 # リストのブロードキャストリング
8
9 plt.plot(x, y)
10 plt.show()
```


タイトルや軸ラベルの表示

```
9 plt.plot(x, y)
10 plt.title("y=x^2")
11 plt.xlabel("x")
12 plt.ylabel("y")
13 plt.show()
```

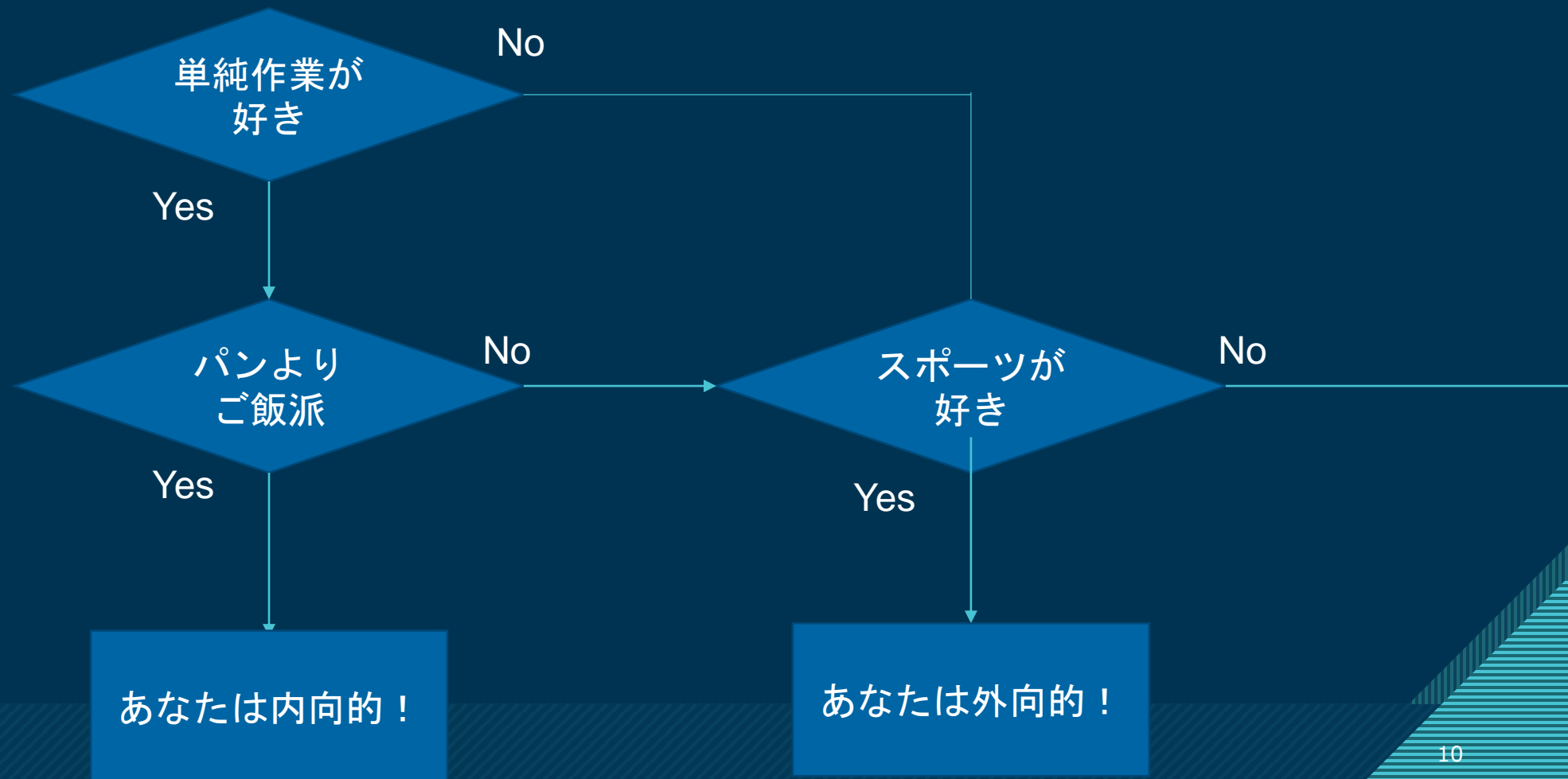


今日の内容

- 条件分岐

身近な例

- 占い
- 性格診断



条件分岐 (if文)

- 条件に応じて、プログラムの挙動を変える
 - たくさん課金していたら、レアアイテムが出やすくする
 - レベル50をこえていたら、先の道へ進める
 - 相手からフォローされていたら、ダイレクトメッセージが遅れる etc… …

if else 文(条件分岐)

- 場合によって実行する内容を変える命令
 - もし…だったら, もしxxxだったら,

書き方

行末にコロンを入れる

```
if (条件):  
    条件を満たした時に実行される内容  
else:  
    条件に合わなかった時に実行される内容
```

インデント

条件分岐で動作させたい内容は、if, elseを書いた行より
決まった数だけ右にずらして書く => Tab キーを押してずらす

行末にコロンを入れる

```
if (条件):  
    条件を満たした時に実行される内容  
else:  
    条件に合わなかった時に実行される内容
```

Tab
キー

Tab
キー

条件の種類

コードの書き方	意味	例
<code>a == b</code>	aとbは等しい	<code>a == 4, s == "ryo"</code>
<code>a < b</code>	aはbより小さい	<code>a < 0.5</code>
<code>a > b</code>	aはbより大きい	<code>a > 0.5</code>
<code>a != b</code>	aはbと等しくない	<code>a != 0</code>
<code>a <= b</code>	aはb以下	<code>a <= 0.5</code>
<code>a >= b</code>	aはb以上	<code>a >= 0.5</code>

条件の注意

「=」：イコールが1つの時は代入

「==」：イコールが2つの時は「等しい」という条件

✕ if (a = 3): エラー

○ if (a == 3): aが3と等しい時にインデントした命令が実行される

コードの例

```
s = input()          入力でryoと入力したら

if (s == "ryo"):
    print("Hello!", s, "san!")

else:
    print(".....")
```

挨拶が入力される

コードの例

```
s = input()

if (s == "ryo"):
    print("Hello!", s, "san!")

else:
    print(".....")
```

それ以外の時には黙る

プログラムの実行順序について

プログラムは、上から順に実行される。

if文の場合、どれかの条件に当てはまったら、elseの命令より後ろに移動する。

```
1 s = input("和訳したい英語を入れて下さい: ")
2
3 print(s + "は日本語で")
4 if (s == "rice"):
5     print("ごはん")
6 elif (s == "bread"):
7     print("パン")
8 elif (s == "noodle"):
9     print("うどん")
10 else:
11     print("まだその単語を知りません.")
12
13
```

if, elif, else 文

```
1 s = input("和訳したい英語を入れて下さい: ")
2
3 print(s + "は日本語で")
4 if (s == "rice"):
5     print("ごはん")
6 elif (s == "bread"):
7     print("パン")
8 elif (s == "noodle"):
9     print("うどん")
10 else:
11     print("まだその単語を知りません.")
12
13
```

判断したい条件が2つ以上ある場合,
elifという命令を使う

s == "rice"ではないが,
s == "bread"だった場合


if, elif, else 文の実行順序

```
1 s = input("和訳したい英語を入れて下さい: ")
2
3 print(s + "は日本語で")
4 if (s == "rice"):
5     print("ごはん")
6 elif (s == "bread"):
7     print("パン")
8 elif (s == "noodle"):
9     print("うどん")
10 else:
11     print("まだその単語を知りません.")
12
13
```

ユーザの入力を待ち受ける

入力されたsをそのままprintする.

if, elif, else 文の実行順序



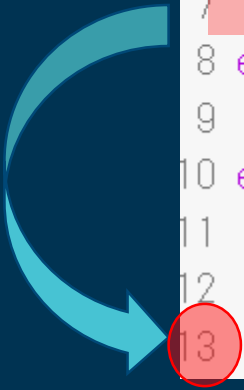
```
1 s = input("和訳したい英語を入れて下さい: ")
2
3 print(s + "は日本語で")
4 if (s == "rice"):
5     print("ごはん")
6 elif (s == "bread"):
7     print("パン")
8 elif (s == "noodle"):
9     print("うどん")
10 else:
11     print("まだその単語を知りません.")
12
13
```

Sの中に含まれる文字列が“rice”と等しいか判断

もし等しければ if の下の命令が実行され、
else より下に移動

if, elif, else 文の実行順序

```
1 s = input("和訳したい英語を入れて下さい: ")
2
3 print(s + "は日本語で")
4 if (s == "rice"):
5     print("ごはん")
6 elif (s == "bread"):
7     print("パン")
8 elif (s == "noodle"):
9     print("うどん")
10 else:
11     print("まだその単語を知りません.")
12
13
```

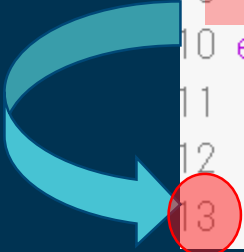


Sの中に含まれる文字列が“rice”と等しくない場合
s == “bread” かどうかを判断する.

もし等しければ elifの下で命令が実行され、
elseより下に移動

if, elif, else 文の実行順序

```
1 s = input("和訳したい英語を入れて下さい: ")
2
3 print(s + "は日本語で")
4 if (s == "rice"):
5     print("ごはん")
6 elif (s == "bread"):
7     print("パン")
8 elif (s == "noodle"):
9     print("うどん")
10 else:
11     print("まだその単語を知りません. ")
12
13
```




Sの中に含まれる文字列が
“rice”とも“bread”と等しくない場合

s == “noodle” かどうかを判断する.

もし等しければ elifの下に命令が実行され,
elseより下に移動

if, elif, else 文の実行順序

```
1 s = input("和訳したい英語を入れて下さい: ")
2
3 print(s + "は日本語で")
4 if (s == "rice"):
5     print("ごはん")
6 elif (s == "bread"):
7     print("パン")
8 elif (s == "noodle"):
9     print("うどん")
10 else:
11     print("まだその単語を知りません.")
12
13
```



上のどの条件とも当てはまらなければ,
else 下の条件が実行されて,
if else のかたまりより下に移動

自由課題1 ずるいじゃんけんゲーム

input()関数でじゃんけんの手を読み込み，コンピュータが絶対に負けないようにprint文で手を出力してください。

if, elif, elseを全部使います。

ヒント

```
print("初めまして，私はじゃんけんに負けないゲームです。私とじゃんけんしませんか？")  
  
print("じゃんけん(グー，チョキ，パーのどれかを入力してください。)")  
s = input()  
  
if (s == ?):  
    print("ポン！ 私の出した手は...")  
elif (s == ?):
```

答え

```
1 print("初めまして、私はじゃんけんに負けないゲームです。私とじゃんけんしませんか?")
2 print("じゃんけん(グー, チョキ, パーのどれかを入力してください。)")
3
4
5 s = input()
6
7 if (s == "パー"):
8     print("ポン! 私の出した手はチョキでした")
9 elif (s == "チョキ"):
10    print("ポン! 私の出した手はグーでした")
11 elif(s == "グー"):
12    print("ポン! 私の出した手はパーでした")
13 else:
14    print("そんな手は聞いたことがありません")
15
```

if else文のインデント

- 条件を満たした後に表示したい, 命令したいことが2つ以上ある場合

```
if (条件):  
    print("Hey!")  
    print("Nice to meet you!")  
    ...  
else:  
    print("Hello!")  
    x = 10  
    y = 20  
    ...
```

全部揃える

全部揃える

それぞれの色をつけた
かたまりをブロックという

【やや難】ネストされたif文

if 文のネスト: if 文の命令の中に, さらにif文を書くこと



```
1 x = 10
2 y = 20
3
4 if (x == 10):
5     print("x is equal to 10")
6     if (y == 20):
7         print("and y is equal to 20")
```



```
x is equal to 10
and y is equal to 20
```



x == 10を満たすときに実行される内容

【やや難】ネストされたif文

if 文のネスト: if 文の命令の中に, さらにif文を書くこと



```
1 x = 10
2 y = 20
3
4 if (x == 10):
5     print("x is equal to 10")
6     if (y == 20):
7         print("and y is equal to 20")
```



```
x is equal to 10
and y is equal to 20
```



x == 10を満たすときに実行される内容



x == 10を満たし, かつ
y == 20 を満たすときに実行される内容

【やや難】ネストされたif文

ネストされたif文を実行したときの様子

```
▶ 1 x = 10
   2 y = 20
   3
   4 if (x == 10):
   5     print("x is equal to 10")
   6     if (y == 20):
   7         print("and y is equal to 20")
```

```
↳ x is equal to 10
   and y is equal to 20
```

```
▶ 1 x = 6
   2 y = 20
   3
   4 if (x == 10):
   5     print("x is equal to 10")
   6     if (y == 20):
   7         print("and y is equal to 20")
```

y == 20を満たすが、 x == 10を満たさないので
[and y is equal to 20]は表示されない

エラーが出てきたら

- ググると90%は答えが出てくる
 - 出てこない場合 [iiijima\[at\]aoni.waseda.jp](mailto:iiijima[at]aoni.waseda.jp) で相談
- エラーの文章をそのままコピー => 検索にかける or “原因”, “エラー”, “対策”などと付け足して検索
- 英語が苦手であれば日本語のページを見る

エラーの良くある原因

- tabキーでインデントしていますか？

```
if (条件):  
    print("Hey!")  
    print("Nice to meet you!")
```

上と比べてずれている
(tabで揃える)

- 文字を半角のダブルクォーテーションで囲っていますか？

- 文字列は必ず半角のクォーテーション, ダブルクォーテーションで囲う

```
print("Hello!")
```

全角のダブルクォーテーションでエラー

random モジュール

- random() 機能

0~1までの間でランダムな値が自動で選ばれる

```
import random  
print(random.random())
```

random random機能
モジュールの

を書いて何度か実行してみてください。

自由課題2・3 コイントス

[2] コイントスをした結果を表示するプログラムを作成してください。

- `x = random.random()` を条件にうまく使ってください
 - 0.5 未満なら 表 (Head)
 - 0.5 以上なら 裏 (Tail) など

コードを実行すると、`x`という変数に0~1までのランダムな数字が入ることを使う

[3] それもできたら、`input()`で入力として H, T を受け取り、勝負を仕掛けてくるプログラムを作ってください。

プログラムは、ユーザが選ばなかったほうを選びます。

自由課題4 ずるくないじゃんけんゲーム

- さっき作ったコードを改良して、ランダムにじゃんけんの手を出すじゃんけんゲームを作成してください。
- `x = random.random()` を条件にうまく使ってください

もっと色々試したい場合

- ノーマル、レア、激レアはどう設定すればいいか
(ガチャはどのような仕組みで動いていると想像できるか?)

大吉、小吉、吉、凶
- 入力されたパスワードがあっている時だけようこそ！と表示する
- 朝起きたら自分の予定をIF文で書いて管理しよう (IF THEN プランニング)

簡潔な条件を書くために

- 適切な条件を決めてif 文をかけるようにする.
 - 適切な条件の選び方: 論理学の考え方を利用 (slackで資料を配布)
- 問題を解いて練習する.
 - ABC 088 A Infinite Coins
 - https://atcoder.jp/contests/abc088/tasks/abc088_a
 - ABC 086 A Products
 - https://atcoder.jp/contests/abc086/tasks/abc086_a