データ科学基礎演習B（2）

データ科学科目部会

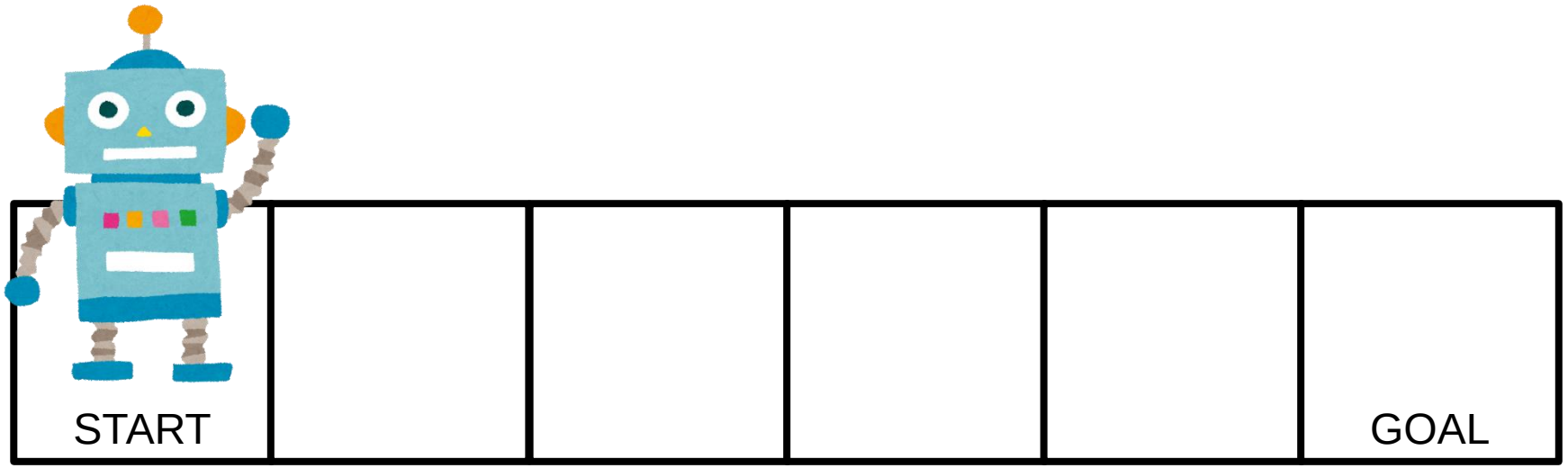
制御構造（その１）

繰り返し処理（for, while）をマスターしよう

どうすればロボットは目的地に着けるでしょう？

- 利用できる命令セット

  ① １歩前に進む



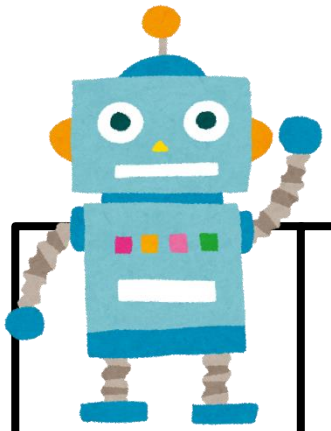
どうすればロボットは目的地に着けるでしょう？

- 利用できる命令セット
  ① １歩前に進む

無駄が多い

プログラム1

１歩前に進む
１歩前に進む
１歩前に進む
１歩前に進む
１歩前に進む



START

GOAL

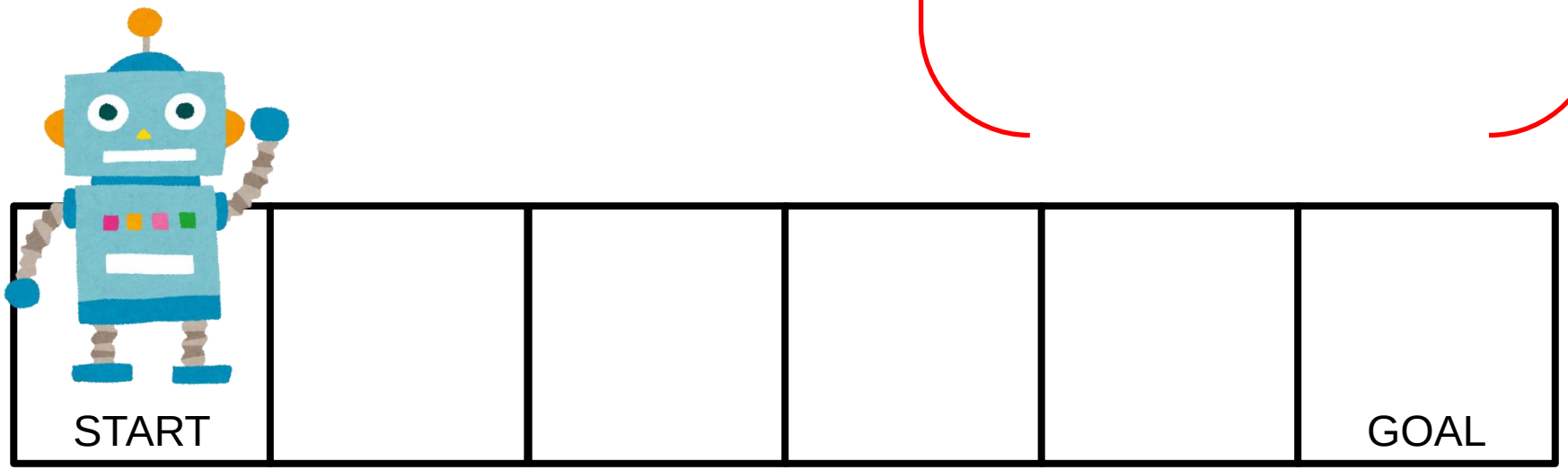
どうすればロボットは目的地に着けるでしょう？

- 利用できる命令セット
  - ① １歩前に進む
  - ② X回繰り返す

プログラム2

５回繰り返す

１歩前に進む

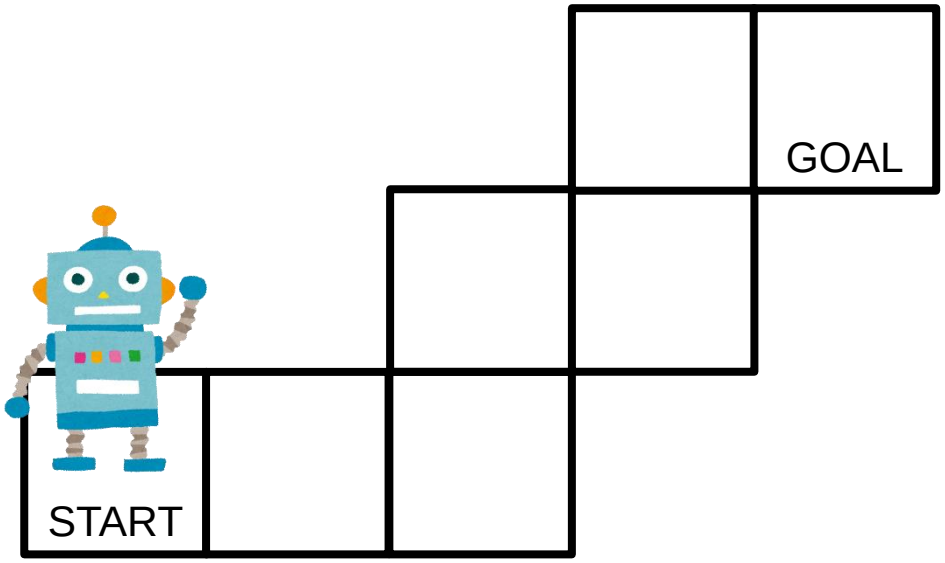


どうすればロボットは目的地に着けるでしょう？

- 利用できる命令セット
  - ① １歩前に進む
  - ② １歩上に進む
  - ③ X回繰り返す

プログラム3

次のスライドを見る前に
考えてみてください



どうすればロボットは目的地に着けるでしょう？

- 利用できる命令セット
  - ① １歩前に進む
  - ② １歩上に進む
  - ③ X回繰り返す

プログラム3

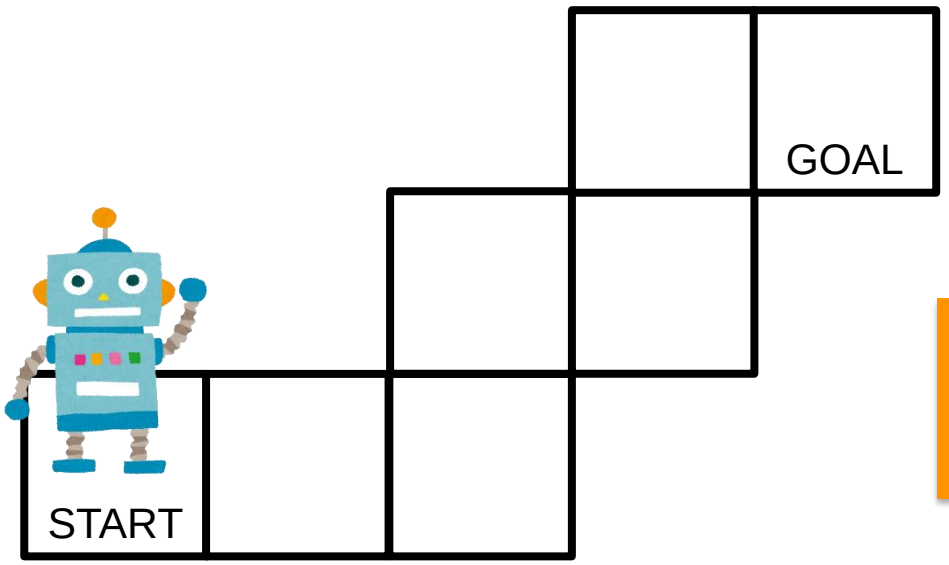
１歩前に進む

２回繰り返す

　１歩前に進む

　１歩上に進む

１歩前に進む



繰り返しをうまく使って
処理を効率化

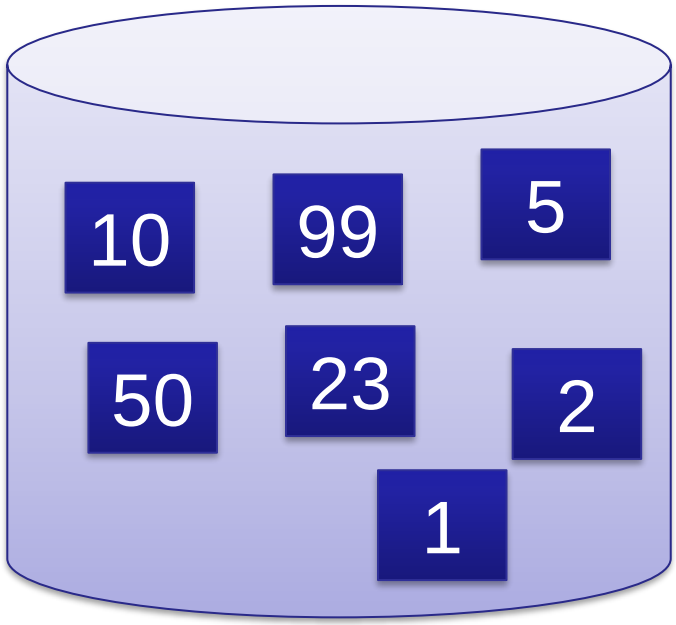
繰り返し処理とは？

- 同じ処理を（データを変えながら）繰り返し実行するしくみ

- Pythonで用意されている繰り返し処理は2種類
  - for
    - 与えられたデータを一つづつ処理するしくみ
  - while
    - ある条件を満たすまで処理を繰り返すしくみ

for

- 与えられたデータ集合（配列，辞書，etc.）から一つずつ要素を取り出して処理を行う



データ集合
（配列，辞書，etc.）

何らかの処理

for の構文

- データ集合から要素を1つずつ取り出して変数に代入し，forの処理ブロックを実行

？

```
for 変数 in データ集合:
    …処理1…
    …処理2…
```

Pythonと処理ブロック

- 処理ブロック ＝ 処理のまとまり
- Pythonはインデントを使って処理ブロックを判別
  - インデントは半角スペース4個（PEP8準拠の場合）

```
for 変数 in データ集合:
    …処理1…
    …処理2…
    for 変数 in データ集合:
        …処理3…
        …処理4…
```

半角スペース4個＝インデント1

半角スペース8個＝インデント2

※ PEP8 = Python Enhancement Proposals

for の構文

- データ集合から要素を1つずつ取り出して変数に代入し，forの処理ブロックを実行
- データ集合からすべての要素を取り出したら繰り返し処理は終了

末尾はコロン(:)

```
for 変数 in データ集合:
    …処理1…
    …処理2…
```

データ集合はどうやって用意すればいい？

- データ集合には配列，辞書，等が利用可能
- 1, 2, …, N のような値が入った配列の作り方
  - range 関数を使って連番配列を作成

① range(終了値)
② range(開始値，終了値)
③ range(開始値，終了値，ステップ幅)

range(終了値)

- [0,1,2,…,終了値-1] の配列（実際はrange型のオブジェクト）を返す
  - 配列の最初の値は 0
  - 配列の最後の値は 終了値-1
  - 配列の要素数 ＝ 終了値
- 配列に変換する際は list(range(N))

range(開始値，終了値)

- [開始値,開始値+1,…,終了値-1] を返す
  - 配列の最初の値は 開始値
  - 配列の最後の値は 終了値-1

```
[1]  1  a = range(1, 5)
     2  list(a)

     [1, 2, 3, 4]

     1
```

range(開始値，終了値，ステップ幅)

- [開始値,開始値+ステップ幅,…,終了値-1] を返す
  - 配列の最初の値は 開始値
  - 配列の最後の値は 終了値-1
  - 配列の要素の間隔は ステップ幅

```
[1]  1  a = range(1, 10, 2)
     2  list(a)
```

```
[1, 3, 5, 7, 9]
```

```
[2]  1  a = range(1, 10, 3)
     2  list(a)
```

```
[1, 4, 7]
```

```
1
```

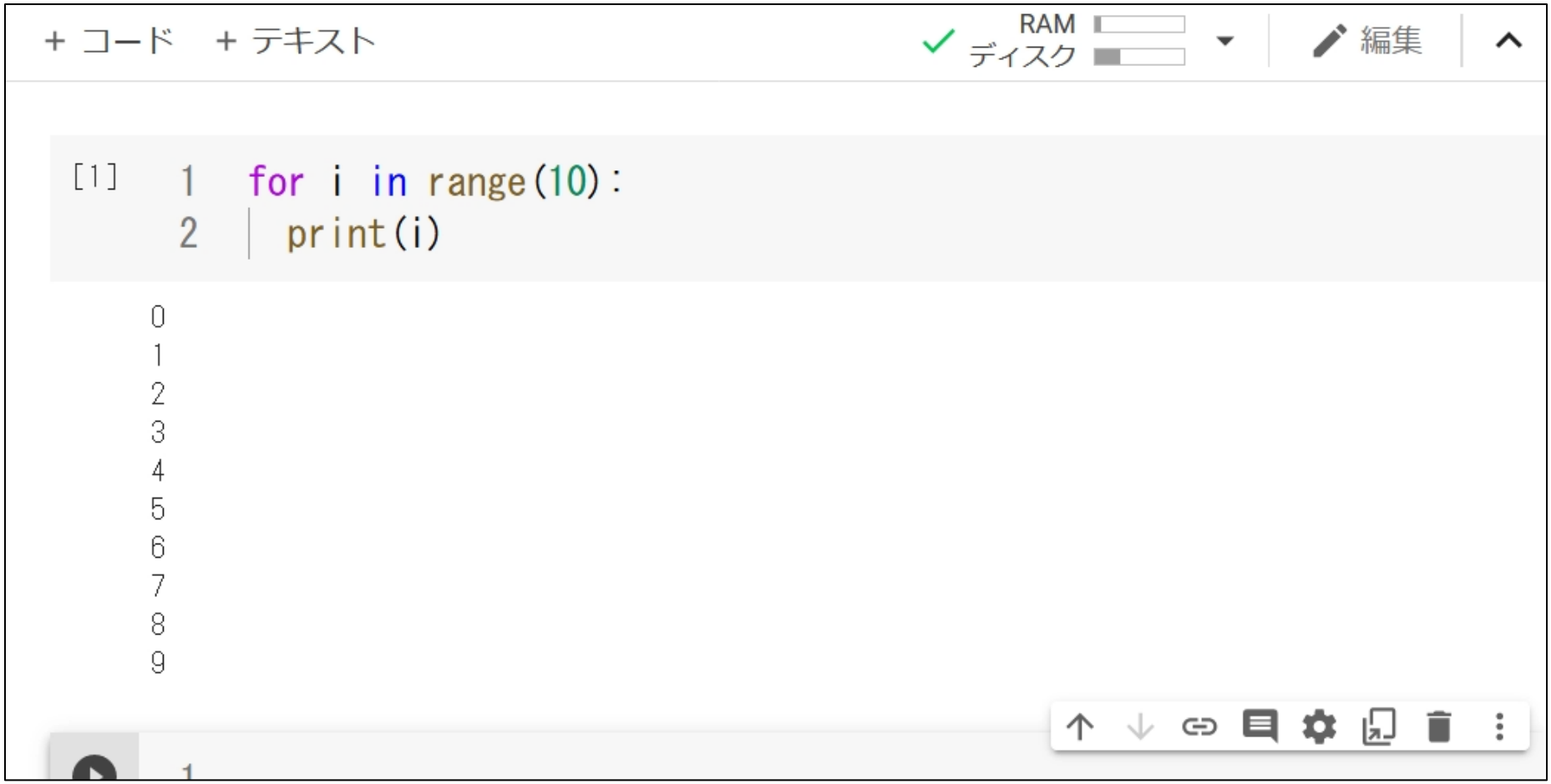
for の構文（再掲）

- データ集合から要素を1つずつ取り出して変数に代入し，forの処理ブロックを実行
- データ集合からすべての要素を取り出したら繰り返し処理は終了

range関数を使って任意の繰り返しに対応するリストを設定

```
for 変数 in データ集合:
    print(変数)
```

0～9までの数字を順に表示してみよう



数字のリストの表示にチャレンジしよう

- 次の順で値を表示するプログラムを書こう

① 3, 4, 5, …, 10

② 2, 5, 8, 11

③ 10, 9, 8, 7, …, 1

while の構文

- 条件式が真（True）の間は処理ブロックを繰り返す

？

```
while 条件式:
    …処理1…
    …処理2…
```

条件式

- 式の計算結果が真(True)もしくは偽(False)となるもの
  - 値の比較(大小関係，等号，不等号，...)
  - True, False そのものも条件式になる
- 条件式は(and，or，not)で組み合わせ可能

- 条件式の例

① 変数 i が 10 より小さい	`i < 10`
② 変数 i が 10 より大きい	`i > 10`
③ 変数 i が 10 以上	`i >= 10`
④ 変数 i が 10 以下	`i <= 10`
⑤ 変数 i が 10 と等しい	`i == 10`
⑥ 変数 i が 10 と等しくない	`i != 10`

条件式の例

```
[4]  1  i = 11
```

```
[5]  1  i < 10
```

```
False
```

```
[6]  1  i > 10
```

```
True
```

```
[7]  1  i >= 11
```

```
True
```

```
[8]  1  i == 11
```

```
True
```

```
[9]  1  i != 11
```

```
False
```

```
1
```

whileを使って1～10を表示してみよう

```
i=1
while i<=10:
    print(i)
    i=i+1
```

変数iを初期化

変数iの値が10以下の間繰り返す

変数iの値を表示

変数iの値を+1

数字のリストの表示にチャレンジしよう

- whileを使って次の順で値を表示するプログラムを書こう

① 3, 4, 5, …, 10

② 2, 5, 8, 11

③ 10, 9, 8, 7, …, 1

print関数の文字出力をカスタマイズしよう（1）

- format関数で文字列中に変数の値を埋め込む
  - 文字列中の {0} や {1} をformat関数の引数で置換
    - {0} や {1} は順番入替や複数回の使用も可能
    - format関数の1番目の引数が {0} になる点に注意！

```
'文字列'.format(変数0,変数1,…)
```

{0}は変数0, {1}は変数1 に置き換えられる

print関数の文字出力をカスタマイズしよう(2)

```
[3]  1  x = 1
     2  y = 2
```

```
[4]  1  print('x, y = {0}, {1}'.format(x, y))
```

```
x, y = 1, 2
```

```
[5]  1  print('y, x = {1}, {0}'.format(x, y))
```

```
y, x = 2, 1
```

```
[6]  1  print('x, y = {0}, {1}, {0}'.format(x, y))
```

```
x, y = 1, 2, 1
```

```
     1
```

print関数の文字出力をカスタマイズしよう(3)

- 文字出力に埋め込む値を書式化する
  - 小数点以下3桁で表示したい

{0} → {0:.3f}

変数の番号

小数点表示指定

```
[1]  1  x = 1/3

[2]  1  print('{0}'.format(x))

     0.3333333333333333

[3]  1  print('{0:.3f}'.format(x))

     0.333
```

print関数の文字出力をカスタマイズしよう(4)

- その他の書式設定
  - 先頭を 0 埋めして表示したい
    - {0:0=5}　　※ 100 → 00100
  - 先頭に必ず符号を付与したい
    - {0:+}　　※ 100 → +100
  - 3桁毎にカンマ(,)を挿入したい
    - {0:,}　　※ 1000000 → 1,000,000

- 文字列中で波括弧({ と })を使う場合
  - 波括弧を2個重ねる(¥は使えない)
    - {{ や }} とする

課題

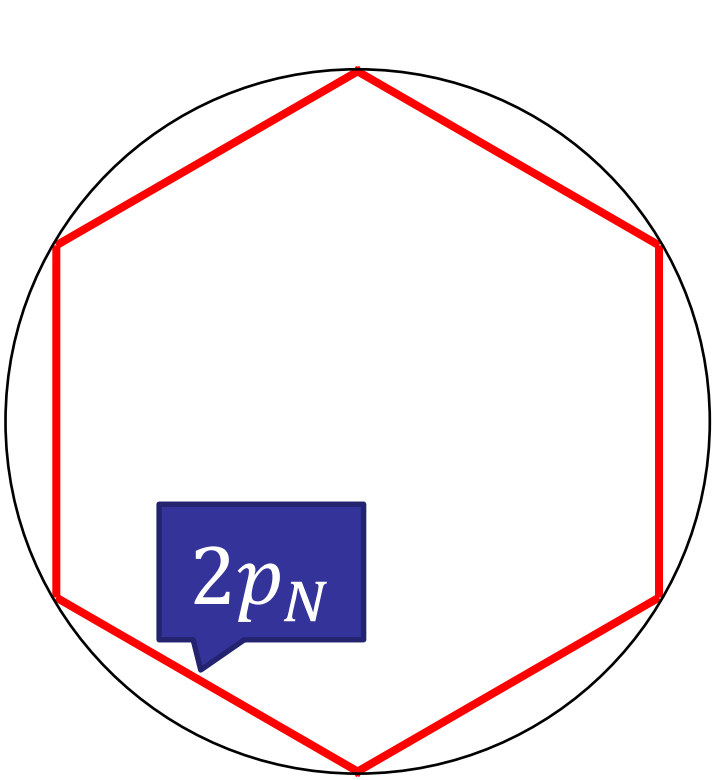
- 繰り返し（for, while）を使って円周率を求めよう

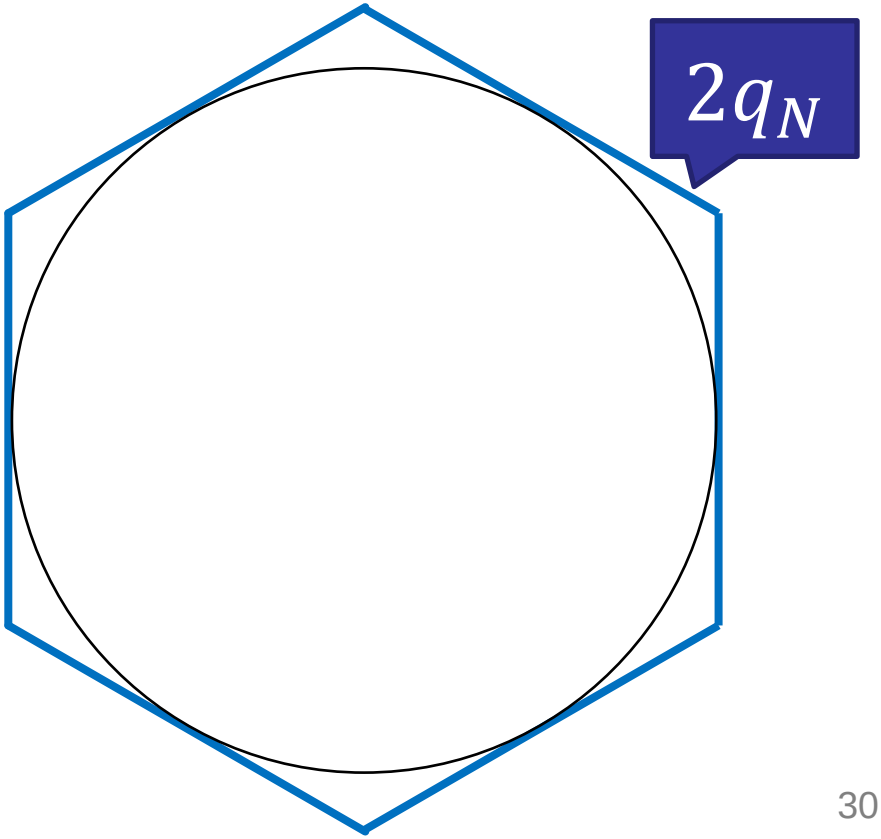
- 下記URLから演習課題のノートブックを自身のGoogle Driveにコピーし，Google Collaboratoryを起動して各設問に回答すること
  - https://bit.ly/3cNyBNZ

- 演習課題を提出する際は，ノートブックのURLで共有し，そのURLを提出すること
  - 課題のヒントは次ページで紹介

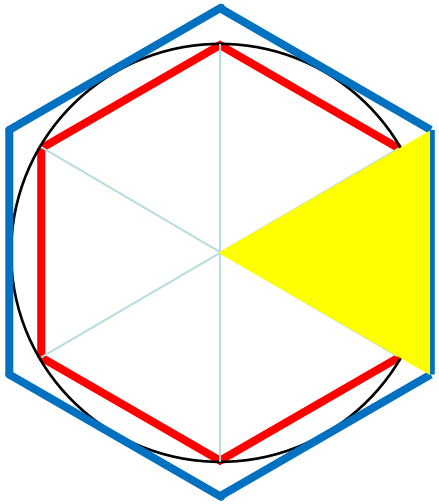
課題のヒント（アルキメデスの定理）

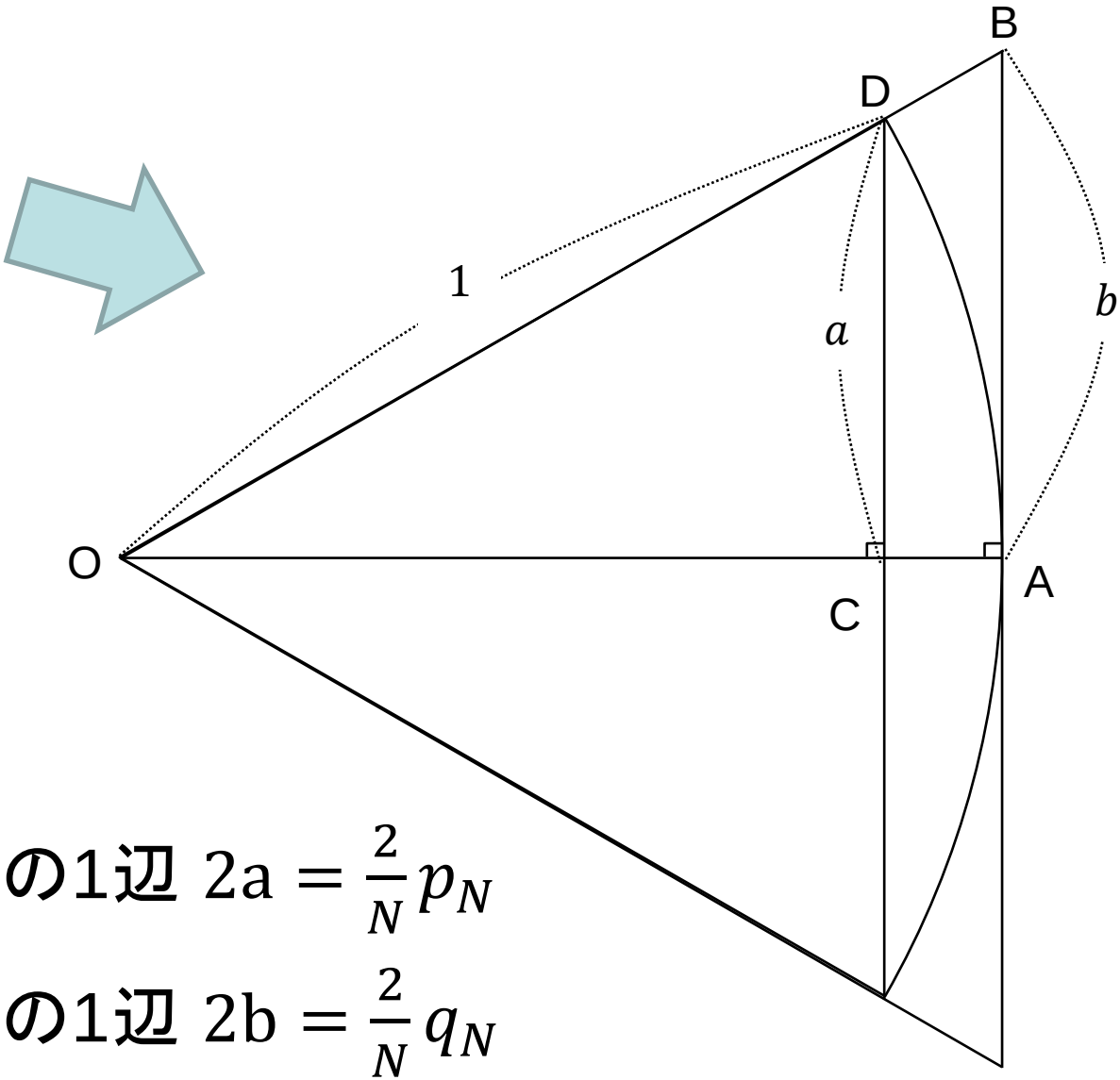
- 半径1の円に内接する正N角形の周長 $2p_N$
- 半径1の円に外接する正N角形の周長 $2q_N$
- 円周率πは $2p_N < 2\pi < 2q_N$ の範囲にある





課題のヒント（アルキメデスの定理）

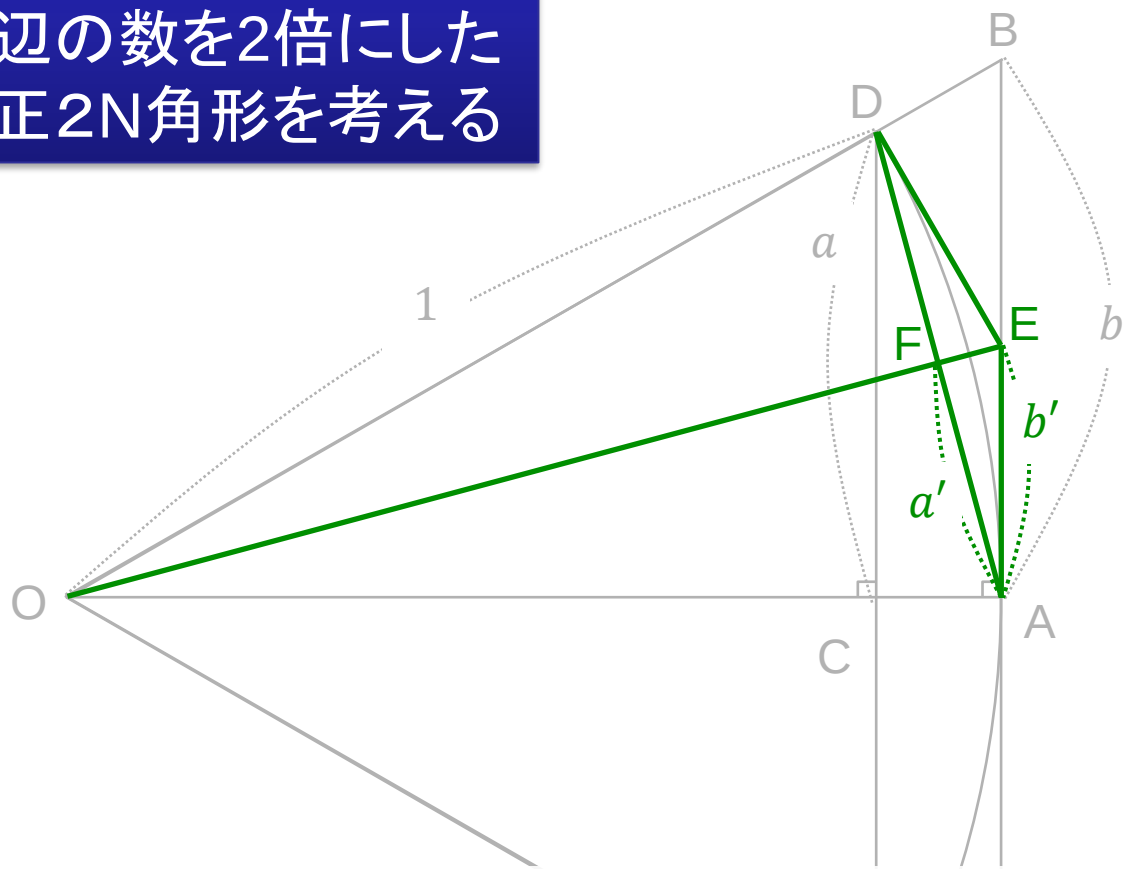




- 内接正N角形の1辺 $2\mathrm{a} = \frac{2}{N} p_N$
- 外接正N角形の1辺 $2\mathrm{b} = \frac{2}{N} q_N$

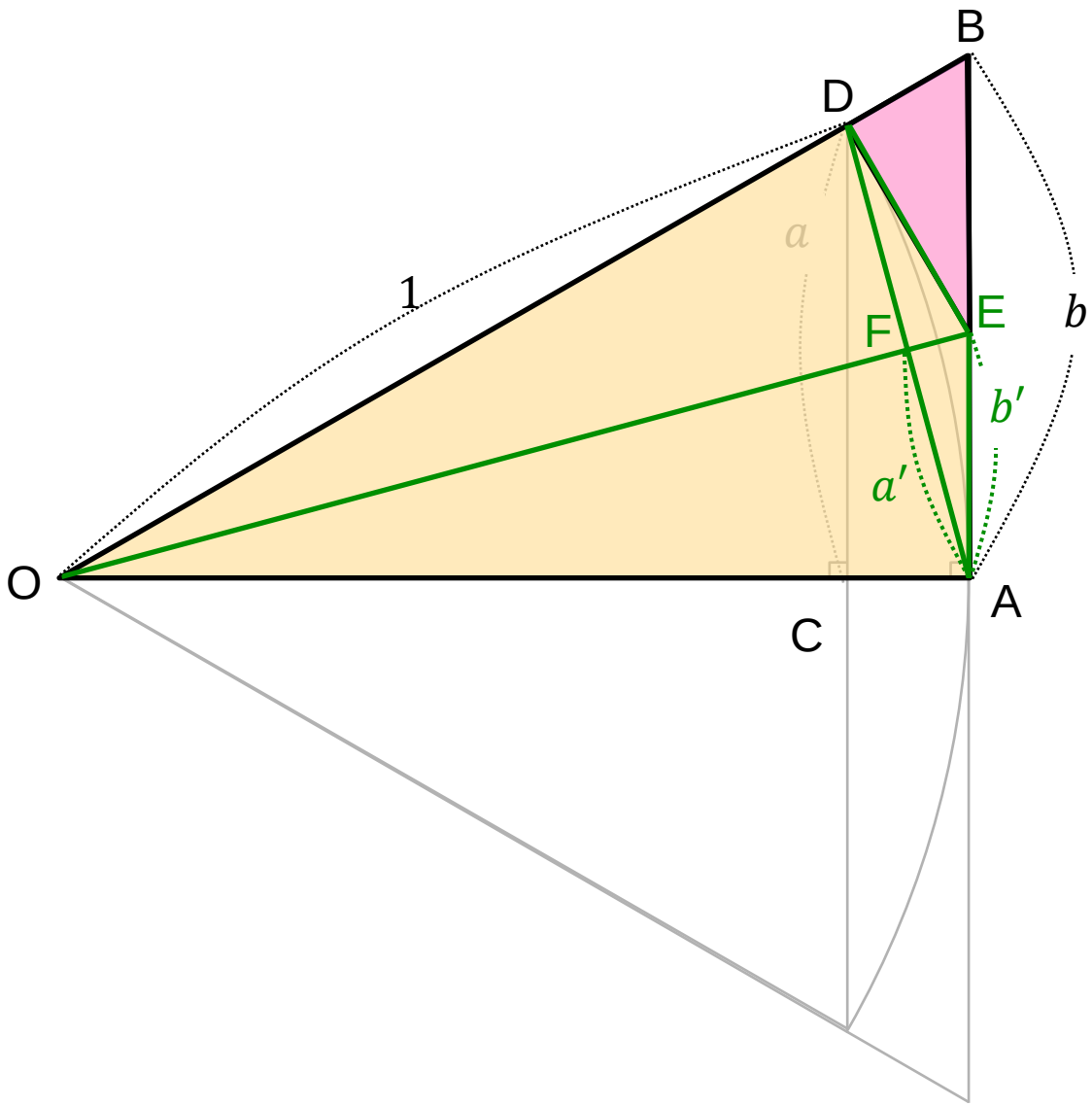
課題のヒント（アルキメデスの定理）

辺の数を2倍にした
正2N角形を考える



- 内接正2N角形の1辺 $2a' = \frac{2}{2N} p_{2N} = \frac{1}{N} p_{2N}$
- 外接正2N角形の1辺 $2b' = \frac{2}{2N} q_{2N} = \frac{1}{N} q_{2N}$

課題のヒント（アルキメデスの定理）



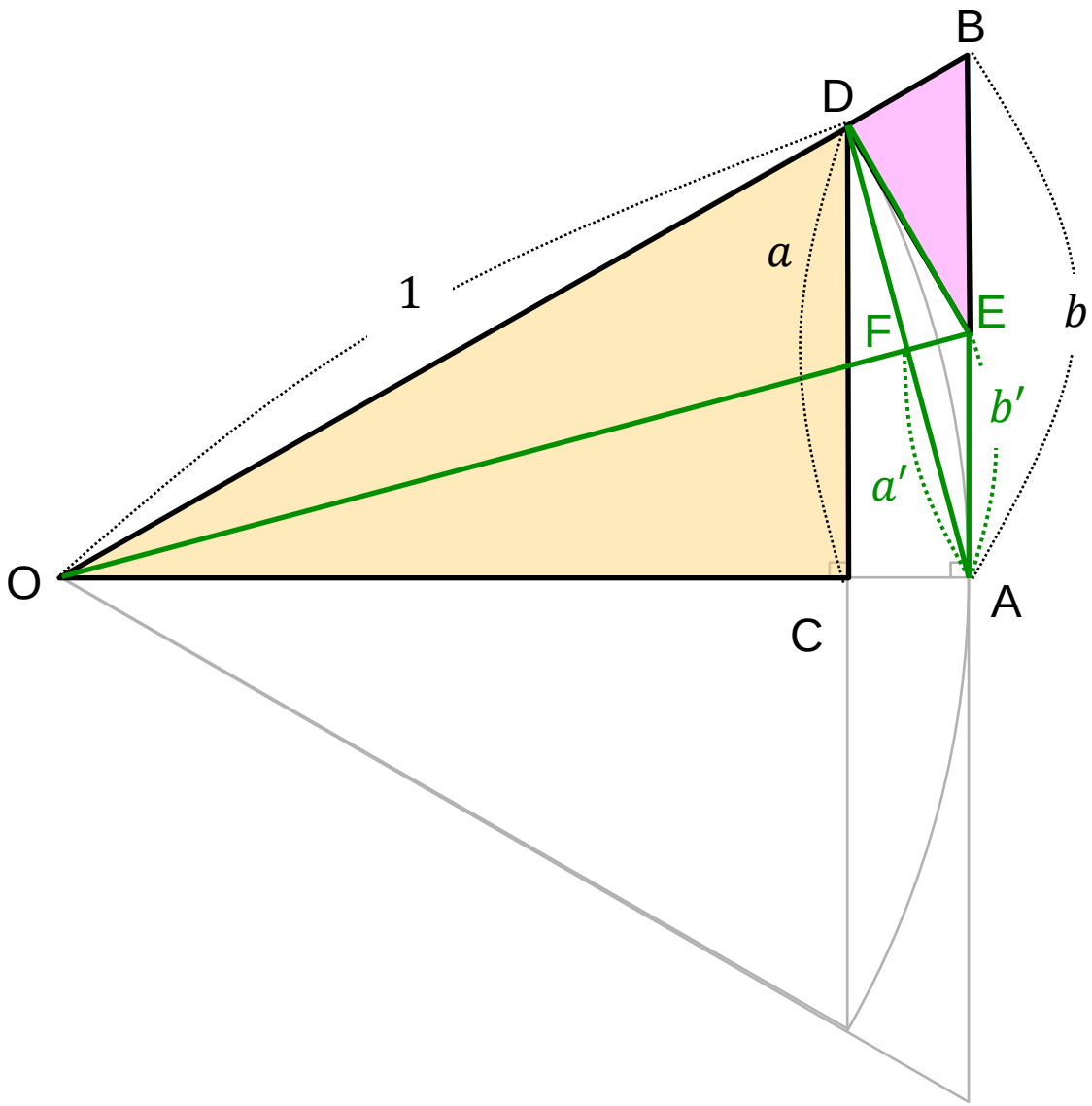
① ∠OBA=∠EBDより

- $\mathrm{OA} : \mathrm{ED} = \mathrm{AB} : \mathrm{DB}$

②ED = EA = b' より

- $1 : b' = b : \mathrm{DB}$
- $\mathrm{DB} = bb'$

課題のヒント（アルキメデスの定理）



① ∠OBA=∠EBDより

- $\mathrm{OA} : \mathrm{ED} = \mathrm{AB} : \mathrm{DB}$

②ED = EA = b' より

- $1 : b' = b : \mathrm{DB}$
- $\mathrm{DB} = bb'$

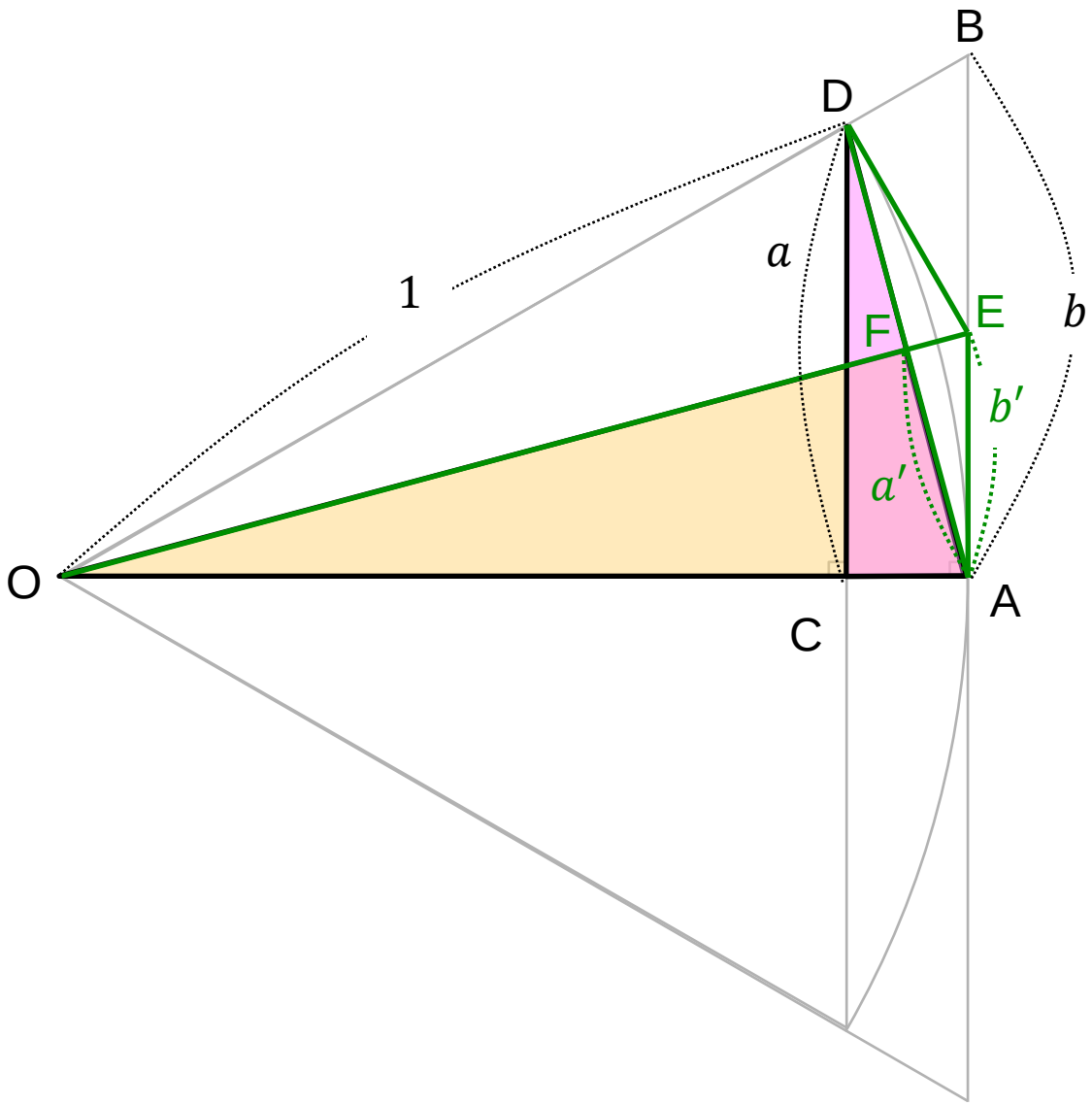
③ ∠ODC =∠EBDより

- $\mathrm{OD} : \mathrm{EB} = \mathrm{CD} : \mathrm{BD}$

④$\mathrm{EB} = b - b'$ より

- $1 : b - b' = a : bb'$
- $bb' = a(b - b')$
- $b' = \frac{ab}{a+b}$

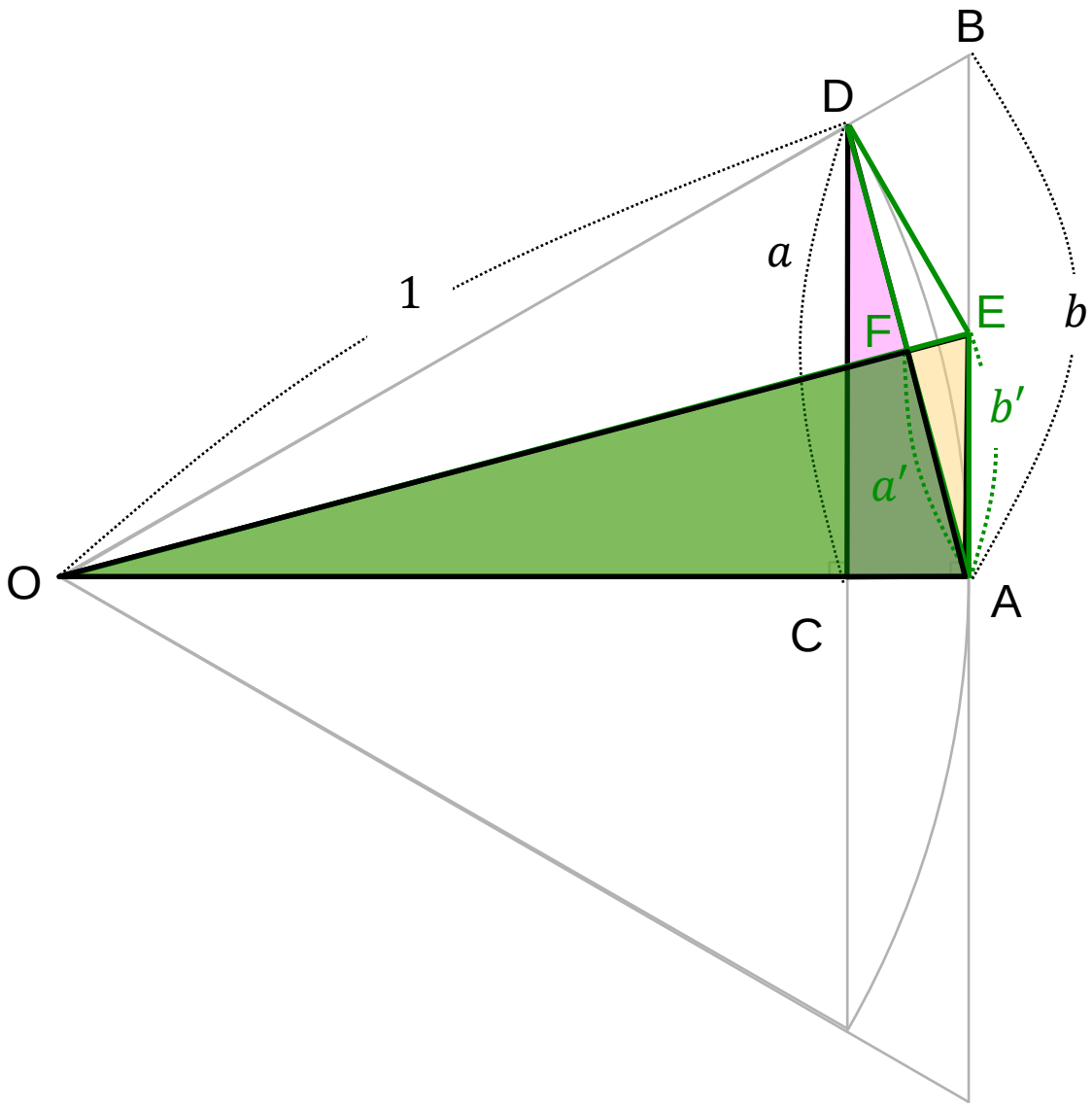
課題のヒント（アルキメデスの定理）



① ∠OAF=∠DACより

- $\mathrm{OA} : \mathrm{DA} = \mathrm{AF} : \mathrm{AC}$
- $1 : 2a' = a' : AC$
- $\mathrm{AC} = 2a'^2$

課題のヒント（アルキメデスの定理）



① ∠OAF=∠DACより

- $\mathrm{OA} : \mathrm{DA} = \mathrm{AF} : \mathrm{AC}$
- $1 : 2a' = a' : AC$
- $\mathrm{AC} = 2a'^2$

② ∠EOA=∠AOFより

- $\mathrm{OA} : \mathrm{DC} = \mathrm{AE} : \mathrm{CA}$
- $1 : a = b' : \mathrm{CA}$
- $\mathrm{CA} = ab'$

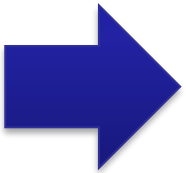
③ $2a'^2 = ab'$ より

- $a' = \sqrt{\frac{ab'}{2}}$

課題のヒント（アルキメデスの定理）

- もともとの設定
  - 内接正N角形の1辺 $2\mathrm{a} = \frac{2}{N} p_N$
  - 外接正N角形の1辺 $2\mathrm{b} = \frac{2}{N} q_N$
  - 内接正2N角形の1辺 $2\mathrm{a}' = \frac{1}{N} p_{2N}$
  - 外接正2N角形の1辺 $2\mathrm{b}' = \frac{1}{N} q_{2N}$

- 図形の相似から求めた式
  - $b' = \frac{ab}{a+b}$
  - $a' = \sqrt{\frac{ab'}{2}}$

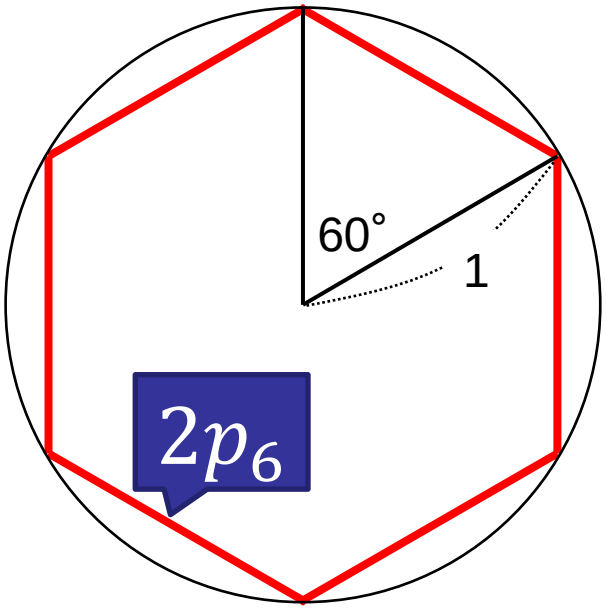


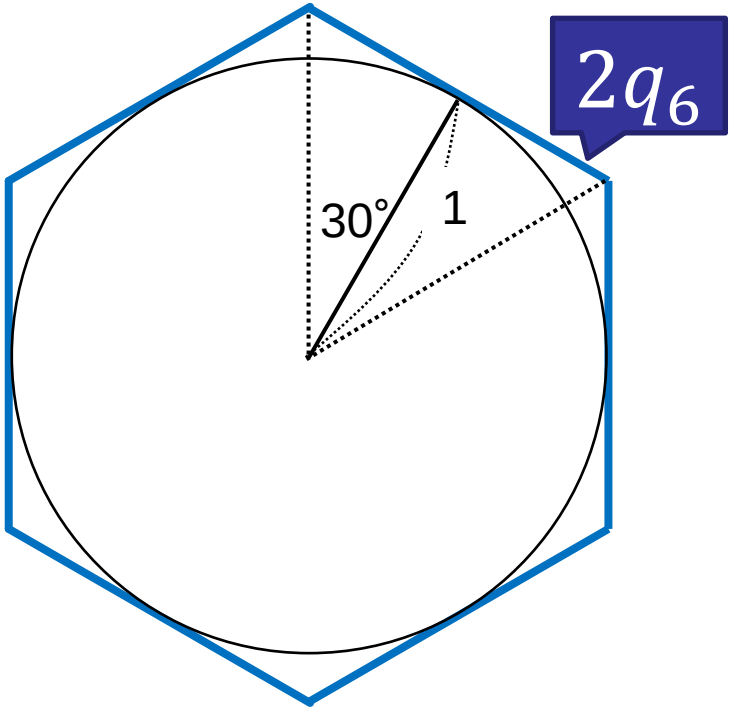
$$q_{2N} = \frac{2p_N q_N}{p_N + q_N}$$

$$p_{2N} = \sqrt{p_N q_{2N}}$$

課題のヒント（アルキメデスの定理）

- 正6角形（$N = 6$）の場合
  - $p_6 = 3,\ \ q_6 = 2\sqrt{3} \ \rightarrow p_6 = 3 < \pi < q_6 = 3.464\cdots$
- $N = 12, 24, \cdots$ は p_6 と q_6 を用いて順に計算





課題のヒント（アルキメデスの定理）

- 正12角形（$N=12$）の場合
  - $q_{12}=\frac{2p_6q_6}{p_6+q_6}=\frac{2\times3\times2\sqrt{3}}{3+2\sqrt{3}}=12\left(2-\sqrt{3}\right)$
  - $p_{12}=\sqrt{p_6q_{12}}=\sqrt{3\times12\left(2-\sqrt{3}\right)}=6\sqrt{2-\sqrt{3}}$

  $\rightarrow p_{12}=3.1058\cdots<\pi<q_{12}=3.2153\cdots$

- $N=24,48,\cdots$ は p_{12} と q_{12} を用いて順に計算

課題提出時の注意点

- 課題提出の際には提出前に必ず以下2点を確認すること
  - Google Collaboratoryの共有設定の際には『リンクを知っている全員』にチェックを入れる
  - 課題提出時に貼り付けたURLの末尾が「?usp=sharing」となっている（共有設定を開き、「リンクのコピー」ボタンを押して共有用のURLをコピー＆ペーストする）