

## 「エラーで学ぶ Python」を活用した情報 I の授業カリキュラム案

「情報 I」はすべての高校生が学ぶ科目となりました。この科目は 4 つの分野に分かれており、「コンピュータとプログラミング」はそのうちの一つです。この分野では高校生が「プログラミング」を学びます。また、大学入学共通テストに「情報 I」が出題されることが決定し、「プログラミング」の理解は受験生にとって必要不可欠となりました。そのため、高校現場では「プログラミング」の実習を伴う授業がますます求められています。

本書は、エラーから Python を学ぶ方式を採用しており、高校での授業に取り入れやすい構成になっています。また、本書は高校生が授業で学ぶためだけでなく、受験生が受験対策として使用するのにも最適です。

本書は平均的な「情報 I」の授業よりも高い難易度をカバーしています。そのため、本書を使って授業を行おうと考える情報科の先生方が、どの程度までどのように授業を進めるべきか悩むことがあるかもしれません。そうした先生方の悩みを解決するため、プログラミング分野の授業カリキュラム案をご提案します。これはあくまで「案」であり、学校や生徒の状況に応じて修正していただくことをお勧めします。以下のカリキュラム案は、中堅校以上で進学志向の生徒が多い学校の「情報 I」授業を想定して作成しました。

ここでは、全 12 回の授業を想定しています。もしこれを短縮したい場合は、7 回目、9 回目、11 回目（さらに短縮するならば、3 回目、5 回目）の授業を実施せずに、章末問題を宿題とすることも可能です。以下の①の授業ではその項目の説明を中心とし、②の授業では①で得た知識および技能を基に、章末問題の演習と解説を行う形です。これにより思考力を育み、さらに最後の 12 回目の授業において、生徒は自身の考えでプログラミングをし、他者に披露するための準備をし、それにより表現力を身につける、というカリキュラムとなっています。

### プログラミング 全 12 回

- |                 |             |
|-----------------|-------------|
| 1. オリエンテーションと準備 | 7. 繰り返し②    |
| 2. 変数と演算①       | 8. 条件分岐①    |
| 3. 変数と演算②       | 9. 条件分岐②    |
| 4. 配列と関数①       | 10. 時間・乱数①  |
| 5. 配列と関数②       | 11. 時間・乱数②  |
| 6. 繰り返し①        | 12. まとめ（発表） |

## ■ 1 時間目      オリエンテーションと準備

### (1) オリエンテーション

この 1 時間目のオリエンテーションは、これから始まるプログラミング学習の基礎となるため、特に重要です。そのため、他の授業よりも詳細に説明します。

生徒たちはこれからプログラミングを学びますが、**まずは彼らにプログラミング学習の必要性について考えさせる時間を設けてください**。多くの生徒がプログラミングを深く学んだ経験がないため、これまでこのようなことを考える機会はほとんどなかったと思われます。私自身、プログラミング以外の単元でも、「なぜその単元を学ぶのか」という点を生徒に考えさせるようにしています。これにより、学ぶ理由が明確になり、生徒のモチベーションを高めることができます。ただプログラミングの画面をいきなり見せられても、多くの生徒は興味を持たず、途方に暮れるかもしれません。「急に難しいことが始まった」「数学みたい」と感じる生徒もいるでしょう。だからこそ、学ぶ理由を明確にすることが重要です。また、先生方が考える学ぶ理由を伝えることも大切です。

例えば、以下のように説明できます。「私たちが使っているパソコンやスマートフォン、その他の家電製品はすべてプログラミングによって動いています。しかし、これらは他人が設定した動作しか行いません。もし自分の好きなようにこれらの機器を設定できたらどうでしょうか？また、私たちが使用するアプリも全てプログラミングによって動いています。自分の思い通りのアプリがなかったり、あるアプリに特定の機能があれば便利だと思うことはありませんか？これらを実現するにはプログラミングスキルが必要です。この授業では、自由にアプリを作れるようになるまではいかなくとも、その基礎を学ぶことができます。ぜひ、この 12 回の授業を通じてプログラミングの基本を身につけましょう。」といった内容で、生徒がプログラミングを「自分ごと」として捉えられるような工夫をすることが推奨されます。

### (2) 環境の準備

次に、Google Colaboratory の使用方法を学びます。生徒たちは、9～10 ページに記載されている手順に従って、Google Colaboratory の設定と基本的な使い方を学びます。これで 1 時間目の授業は終了です。

## ■ 2 時間目 変数と演算①

変数と演算について学ぶことを伝え、変数とは、「本文：プログラムを動かすために重要な仕組みが『変数』です。変数はいろいろな値を入れておくための入れ物と考えればよいでしょう。」と説明します。

### (1) 変数を使う

```
01: apple = 200
02: print(apple)
```

このプログラムを示し、生徒にはこのプログラム結果はどうなるのかをたずねてください。発表等で生徒の意見が出た後に、「本文：apple = 200 は、変数 apple に 200 という整数の値を代入しています。「=」は代入演算子と呼ばれ、apple = 200 は右辺の 200 を左辺の apple に代入するという意味になります。apple ← 200 と考えるとわかりやすいでしょう。print(apple)は、apple という文字列が表示されるのではなく、変数 apple の値である 200 が表示されるのです。プログラムは、上から下へ順に実行されます。これを「順次構造」と呼びます。」と説明をします。

### (2) データ型について

#### データ型

多くのデータ型がありますが、まずは次の5つを押さえておきましょう。

データ型	説明	記述例
文字列 (str)	シングルクォーテーション (' ') とダブルクォーテーション (" ") のどちらかで囲む	apple = 'リンゴ'
整数 (int)	小数点を含まない数値	num = 6 num = -3
浮動小数点(float)	小数点を含む数値	num = 3.14
真偽 (bool)	真 (True)、偽 (False) で定義する	check = True check = False
配列 (list)	複数の要素 (文字列、整数など) を含む	fruits = ['リンゴ', 'バナナ', 'みかん', 'ぶどう']

上記を示しつつ、「本文：プログラミング言語によっては、『この変数は整数が入る入れ物です』とデータ型を最初に宣言します。最初に型宣言することで、本来は整数の入る箱なのに文字列を入れてしまったというような間違いを防ぐことができます。Python では、この型宣言がありません。同じ変数に整数を入れたり、文字列を入れたりできます。自由度が高

い分、気がついたらプログラムの途中でデータ型が変わっていたということに注意しなくてはなりません。」と説明します。

### (3) 変数の命令規則について

#### 変数の命名規則

Pythonの変数名には、次のような規則があります。

- アルファベット、0~9の数字、アンダースコア ( `_` ) を使うことができる
- 最初の文字は、アルファベットまたはアンダースコアにする
- 予約語は使えない
- アルファベットの小文字と大文字は区別される

予約語とは、Pythonでプログラムを記述するときに特別な意味を持つ重要な単語のことです。次の単語が予約語となっています。

```
False None True and as assert async await break class continue def del elif  
else except finally for from global if import in is lambda nonlocal not or  
pass raise return try while with yield
```

### (4) 四則演算について

これは演習を伴う説明をしてください。

```
01: apple = 200  
02: banana = 100  
03: total = apple + banana  
04: print(total)
```

上記を示して、出力されるものは何になるのかを答えさせてください。

その後、四則演算について説明をしてください。

四則演算は以下の4種類です。かけ算、わり算の記号が数学の場合と異なるので注意が必要です。

たし算 (+) …和    ひき算 (-) …差    かけ算 (\*) …積    わり算 (/) …商

※ 時間があれば15~19ページも説明します。時間がない場合には、宿題として読んでくれるように伝えます。

## ■ 3 時間目 変数と演算②

3 時間目は演習から入ります。

### Q 問題

200円のリンゴ1個と100円のバナナ1本を買って、500円を払いました。  
おつりはいくらでしょうか。

```
01: apple = 200
02: banana = 100
03: change = 500 - (apple + banana)
04: Print(change)
```

結果が表示されません。

このように問題を示し、生徒に考えさせた後に、解答を見せていくというやり方で実施していきます。

「おつりの計算2 (25 ページ)」「リンゴとバナナの値段 (27 ページ)」「小数の足し算 (29 ページ)」「数字の入れ替え (31 ページ)」を順に生徒に見せて、考えさせるといいでしょう。一定の時間を区切ってグループ単位で発表させるのも良い方法だと思います。

仮に全部の問題について実施することが難しくても、「数字の入れ替え (31 ページ)」は特に大切な考え方なので、最後に取り扱うように時間調整をしてください。

## ■ 4 時間目 配列（リスト）と関数①

配列について説明をします。「本文：変数は数や文字列などいろいろな値を1つだけ入れておく箱のような入れ物です。『配列（以下、リスト）』は複数の値が入る引き出しのようなものです。」と説明します。

### (1) 配列（リスト）を使う

**例** 20以下の素数のリスト (prime\_number) の値を表示します。

```
01: prime_number = [2,5,7,11,13,17,19]
02: print(prime_number[3])
```

このプログラムを示し、生徒にはこのプログラム結果はどうかをたずねてください。発表等で生徒の意見が出た後に、

変数名	prime_number						
インデックス	0	1	2	3	4	5	6
要素	2	5	7	11	13	17	19

こちらを示し、「本文：リストは、角カッコ（ [] ）を使って定義し、それぞれの要素の間はカンマ（ , ）で区切ります。要素が文字列の場合には、シングルクォーテーション（ ' ）かダブルクォーテーション（ " ）のどちらかで囲みます。それぞれの要素を取り出すときには、リスト名[要素番号] のように指定します。要素番号をインデックスといいます。インデックスは「0 ゼロ」から始まるので注意が必要です。」

と説明をしてください。

### (2) 関数を使う

**例** 入力したリンゴの値段を表示します。

```
01: apple = input('リンゴはいくらですか?')
02: print('リンゴの値段は、' + apple + '円です')

リンゴはいくらですか?200
リンゴの値段は、200円です
```

このプログラムを示し、生徒にはこのプログラム結果を示してあげてください。

その後、「本文：input 関数は、ユーザーからのキーボード入力を受け付けるための関数です。a=input() とすると、入力されたデータが変数 a に代入されます。a=input(文字列) とすれば文字列を表示した状態で、入力を受け付けることができます。ただし、input 関数で受け付けたデータは数字であっても文字列として扱われるので注意が必要です。print(3) や input('いくら?') のように、関数の ( ) の中に書かれているものを「引数（ひ

きすう)」と呼びます。関数はプログラムから引数を受け取り、その値をもとに様々な処理を行います。」と説明をします。



時間があれば、Q1～4の練習問題を解かせて、説明をしてください。

宿題として、40～43ページを確認してくるよう伝えます。

## ■ 5 時間目 配列（リスト）と関数②

5 時間目は演習から入ります。

### Q 問題

二次方程式  $ax^2 + bx + c = 0$  の解を求めます。

$$\text{解の公式 } x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

```
01: a = 2
02: b = -7
03: c = 3
04: x1 = -b+sqrt(b*b-4*a*c)/2*a
05: x2 = -b-sqrt(b*b-4*a*c)/2*a
06: print(x1)
07: print(x2)
```

結果が出力されません。

このように問題を示し、生徒に考えさせた後に、解答を見せていくというやり方で実施していきます。

「3つの数の最大値 (53 ページ)」「数字の削除 (55 ページ)」「リストの並び替え 1 (57 ページ)」「リストの並び替え 2 (57 ページ)」を順に生徒に見せて、考えさせるといいでしょう。一定の時間を区切ってグループ単位で発表させるのも良い方法だと思います。

ちなみに内容が難しくなってきているので、学校の実情に合わせて問題を選ぶのも良いと思います。

## 6 時間目 繰り返し①

本日は繰り返しについて学ぶことを伝え、「本文：プログラミングでは、決まった回数や条件を満たしている間は同じ処理を繰り返したりすることがよくあります。Python では for 文と while 文を使うことができます。これを『反復構造』といいます。」と説明します。

### (1) for 文を使う

「本文：for 文は決められた回数を繰り返す処理を行う命令です。基本的な for 文の書き方は次のようになります。」と説明した後にプログラムを示します。

**例** 0~4までの整数を表示します。

```
01: num = [0, 1, 2, 3, 4]
02: for i in num:
03:     print(i)
```

このプログラムを示し、生徒にはこのプログラム結果はどうなるのかをたずねてください。発表等で生徒の意見が出た後に、

「本文：リスト型の変数 num に 0~4 までの整数が要素として入っています。for i in num: とすることで、変数 num のインデックスの小さい方から順番に要素が i に代入され、最後の要素まで繰り返されます。結果的にリストの要素数だけ処理が繰り返されることとなります。また、リストは num = [0, 1, -1, 9] のように、数値が順に並んでいなくても OK です。」と説明します。

※ 時間があれば 63~65 ページも説明します。

### (2) while 文を使う

「本文：while 文は条件式が真(True)の間、処理を繰り返し行う命令です。基本的な while 文の書き方は次のようになります。」と説明した後にプログラムを示します。

**例** 0~9までの整数を表示します。

```
01: i = 0
02: while i < 10:
03:     print(i,end=',')
04:     i += 1
```

このプログラムを示し、生徒にはこのプログラム結果はどうなるのかをたずねてください。発表等で生徒の意見が出た後に、

「本文：変数 i の初期値は 0 です。2 行目の条件式 i < 10 が真 (True) なので、3 行目で 0 が表示され、4 行目で i が +1 されて、i = 1 となります。4 行目が実行されると、2 行目に戻り、再度、条件式 i < 10 と判定され、3・4 行目が実行されます。このような処理を繰り返す。」

返し、 $i = 10$  のとき、条件式  $i < 10$  (False) と判定され、while 文の繰り返しが終わります。」と説明をします。

>を比較演算子といいます。Pythonでは、以下の比較演算子を使うことができます。

小なり	<
大なり	>
以下	<=
以上	>=
等しい	==
等しくない	!=

## ■ 7 時間目 繰り返し②

7 時間目は演習から入ります。

### Q 問題

1~19までの奇数 (1,3,5,7,9,11,13,15,17,19) の和を求めます。

```
01: total = 0
02: for num in range(10):
03:     odd = 2 * num + 1
04: total = total + odd
05: print(total)
```

計算結果が正しくありません。

19

このように問題を示し、生徒に考えさせた後に、解答を見せていくというやり方で実施していきます。

「1~10000 までの積 (75 ページ)」「文字列の逆転 (77 ページ)」「九九を表示 (79 ページ)」「リストの要素を逆順に表示 (81 ページ)」「リストの要素を合計 (83 ページ)」「入力した自然数の和 (85 ページ)」「フィボナッチ数列 (88 ページ)」を順に生徒に見せて、考えさせるといいでしょう。一定の時間を区切ってグループ単位で発表させるのも良い方法だと思います。

ちなみに内容が難しくなっているのですが、学校の実情に合わせて問題を選ぶのも良いと思います。

## ■ 8 時間目 条件分岐①

本日は繰り返しについて学ぶことを伝え、「プログラミングでは、ある条件が真 (True) か偽 (False) によって、どのような処理を行うかを定めることができます。これを条件分岐 (あるいは選択構造) といいます。Python では if 文が使えます。」と説明します。

### (1) if 文を使う

「本文：条件式が真 (True) のときに処理を行う、基本的な if 文の書き方です。」と説明した後に if 文の型を示します。

```
01: if 条件式:
02:     条件式が真のとき実行する処理1
03:     条件式が真のとき実行する処理2
04:     ...
```

「本文：条件式が真 (True) のときの処理と偽 (False) のときの処理を行う、基本的な if 文の書き方です。」と説明した後に if 文の型を示します。

```
01: if 条件式:
02:     条件式が真のとき実行する処理
03:     ...
04: else:
05:     条件式が偽のとき実行する処理
06:     ...
```

「if 文を記述するときの注意点が 3 つあります。」と述べて、以下を示します。

- ・ if と else の末尾に : (コロン) が必要になる
- ・ if と else のインデントは揃える
- ・ if と else の以降は、インデントする

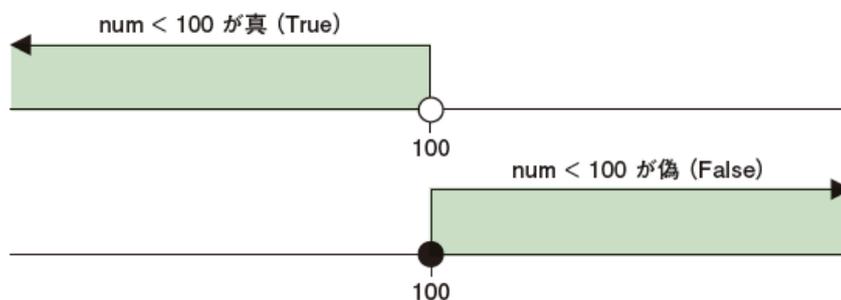
条件式では以下の比較演算子を使うことができます。

小なり	<
大なり	>
以下	<=
以上	>=
等しい	==
等しくない	!=

以下のプログラムを示し、生徒にはこのプログラム結果はどうなるのかをたずねてください。

```
01: num = 120
02: if num < 100:
03:     print('100より小さい数です')
04: else:
05:     print('100以上の数です')
```

発表等で生徒の意見が出た後に、正解を示し、さらに「本文:条件式  $num < 100$  が真(True)の場合と偽(False)の場合の数の範囲は以下ようになります。」と説明します。



※ 時間があれば 93~100 ページも説明します。

## ■ 9 時間目 条件分岐②

9 時間目は演習から入ります。

### Q 問題

入力された数が100と一致するか否かを判定します。

```
01: num = input('整数を入力してください')
02: if int(num) = 100:
03:     print('入力した数は100です')
04: else:
05:     print('入力した数は100ではありません')
```

実行するとエラーが発生します。

このように問題を示し、生徒に考えさせた後に、解答を見せていくというやり方で実施していきます。

「反比例の値 (103 ページ)」「入場料の判定 (105 ページ)」「複数条件の判定 (79 ページ)」「絶対値を求める (110 ページ)」「3 つの数の最小値を求める (112 ページ)」「素因数分解 (115 ページ)」「二次方程式の解を求める (118 ページ)」を順に生徒に見せて、考えさせるといいでしょう。一定の時間を区切ってグループ単位で発表させるのも良い方法だと思います。

ちなみに内容が難しくなってきたので、学校の実情に合わせて問題を選ぶのも良いと思います。

## ■ 10 時間目 時間・乱数①

時間と乱数について学ぶことを伝え、「本文：プログラミングでは、時間を管理できます。処理にどれくらいの時間がかかったかを計測したり、処理に時間制限を設けたりできます。ゲームなどで次第に表示が速くなることにも利用されています。」「本文：乱数を利用すると、プログラムの利用範囲が広がります。ゲームなどで敵の出現の仕方や回数を変えることにも利用されています。」と説明します。

### (1) time モジュールを使う

「本文：time モジュールを使うと、現在時刻を確認したり、処理を一時的に停止できる関数が利用できます。」と説明した後にプログラムを示します。

**例** 現在時刻を表示します。

```
01: import time
02:
03: now = time.strftime('%Y年%m月%d日 %H時%M分%S秒')
04: print(now)
```

```
2023年11月06日 02時52分38秒
```

time モジュールの strftime 関数を使うと、年月日などを好みの形式で表示できます。

**例** 1 2 3 ダー！ を1行ずつ表示します。各表示で1秒の間が空きます。

```
01: import time
02:
03: print(1)
04: time.sleep(1)
05: print(2)
06: time.sleep(1)
07: print(3)
08: time.sleep(1)
09: print('ダー！')
```

```
1
2
3
ダー！
```

sleep 関数を使うと、指定した秒数で処理を停止させることができます。

time.sleep(1) で1秒間停止します。

※ 時間があれば 123 ページも説明します。

## (2) random モジュールを使う

「本文：random モジュールを使うと、決められた範囲の数を生成させたり、順番をバラバラに混ぜたりする関数があります。」と説明した後にプログラムを示します。

**例** 範囲を指定して整数を生成します。

```
01: import random
02:
03: for i in range(10):
04:     num = random.randint(1,9)
05:     print(num,end=', ')
3,8,2,1,8,5,3,6,3,9,
```

random モジュールの randint 関数を使います。

random.randint(start,end) とすると、 $start \leq n \leq end$  の範囲の整数  $n$  を生成できます。

この場合は、1 から 9 までの数字を生成します。

random モジュールには、以下のような乱数を生成する関数があります。

- random.random() 0.0 以上 1.0 未満の浮動小数点数
- random.uniform(a,b)  $a \leq n \leq b$  (または  $b \leq n \leq a$ ) の範囲の浮動小数点数

※ 時間があれば 124~134 ページも説明します。

## ■ 11 時間目 時間・乱数②

12 時間目は演習から入ります。

### Q 問題

「気合いだー！」を10回点滅させて表示します。

1回しか表示されないように見えます。

```
01: import time
02:
03: for i in range(10):
04:     print('%r気合いだー!',end='')
05:     time.sleep(1)
06:     print('%r',end='')
```

このように問題を示し、生徒に考えさせた後に、解答を見せていくというやり方で実施していきます。

「ルパン三世風タイトル表示 (137 ページ)」「10 秒で止めろ! (139 ページ)」「くじびき 1 (141 ページ)」「くじびき 2 (143 ページ)」「スピード暗算 (145 ページ)」を順に生徒に見せて、考えさせるといいでしょう。一定の時間を区切ってグループ単位で発表させるのも良い方法だと思います。

ちなみに内容が難しくなってきたので、学校の実情に合わせて問題を選ぶのも良いと思います。

## ■ 12 時間目 まとめ（発表）

これまでの学習を活かし、生徒たちに何かプログラムを作成することを奨励してください。単純なプログラムでも構いません。これまでに取り組んだ章末問題のプログラムを少しアレンジするだけでも十分です。作成したプログラムをクラスの仲間に発表する時間も設けることが望ましいです。自らが作り上げたプログラムを仲間に披露することによる**成功体験は、将来より複雑なプログラムやアプリの開発への意欲を育てます**。この授業の最後に、生徒たちがプログラミングに対する興味や意欲を持って終わることが重要です。

生徒たちに、これまで学んだ内容は基礎であり、この知識と技能を活かして、自分自身で望むプログラムを作成していくことを伝えてください。これまでのプログラムのエラーから学んだ経験を持つ生徒なら、さらに少しの知識を加えることで簡単なアプリを作成できるレベルに達するでしょう。**プログラミングは自分の理想を実現するための学びであり、生徒たちがそれを「自分ごと」として感じられるようになることが望ましいです**。

以上