オンラインプログラミング環境Google Colaboratoryにおける Pythonの導入実践と生徒の実態

山本　周[1,a)]　清水 克彦[2,b)]

概要：「情報 I」の導入により，プログラミングが必修となった情報の授業では，文部科学省が公開した教員研修用教材で Python が使用されており，採択される可能性が高い．そこで Python によるプログラミング指導の効果の検討を目的とした．“Google Colaboratory”を用いた Python の導入授業を行った．プログラミング経験者が 1 割程度である高校 3 年生に対して，計 5 時間の実施をした．“Google Colaboratory”を用いることで面倒な環境構築がなくなり，関連するライブラリのインストールも容易であった．また，実施したアンケートでは授業前に「プログラミングに興味があるか」に対して，「興味がある」が 20.0 %，「興味がない」が 27.6 %であった．計 5 時間授業を行った後に「プログラミングに興味を持ったか」に対して，「興味を持った」が 51.0 %，「興味を持たなかった」が 16.6 %などの結果を得た．“Google Colaboratory”による Python の指導は，導入なども含めて有用であることが分かった．アンケートの結果からは，生徒の興味を引くことができる等の結果を得た．

キーワード：プログラミング，Python，Google Colaboratory，情報 I

1. はじめに

高等学校の共通教科情報科は，高度情報化社会に対応した人材を育成するために，情報の収集・分析から発信までを総合的に学ぶために，2003 年に新設された教科である．当初は，情報教育の 3 本柱である「情報活用の実践力」を重視する科目「情報 A」，「情報の科学的な理解」を重視する科目「情報 B」，「情報社会に参画する態度」を重視する科目「情報 C」の 3 科目の中から，1 科目を選択必履修する形で始まった．その後，2009 年の学習指導要領改訂 [1] では内容が整理され，選択必履修科目も「社会と情報」と「情報の科学」の 2 科目となった．そして，2018 年に告示された新学習指導要領解説 [1] では，「問題の発見・解決に向けて，事象を情報とその結び付きの視点から捉え，情報技術を適切かつ効果的に活用する力を全ての生徒に育む」ことを目標とし，従来の科目選択制から共通必履修科目として「情報 I 」が設定された．さらに発展的な内容の選択科目として「情報 II 」を設けた．「情報 I 」の学習内容の構成 [1，p.21] は以下の通りである．

内容 (1) 情報社会の問題解決
内容 (2) コミュニケーションと情報デザイン
内容 (3) コンピュータとプログラミング
内容 (4) 情報通信ネットワークとデータの利用

今回の改訂における中心的な変化としては，内

[1] 東京理科大学大学院理学研究科科学教育専攻
[2] 東京理科大学大学院
[a)] 1719527@ed.tus.ac.jp
[b)] kats @ ma.kagu.sut.ac.jp

容 (3) に「コンピュータとプログラミング」が作られたため，すべての高校生がプログラミングを学ぶことである．内容としても，「アルゴリズムを表現する手段，プログラミングによってコンピュータや情報通信ネットワークを活用する方法」[1] に関する知識及び技能を身に付け，「目的に応じたアルゴリズムを考え適切な方法で表現し，プログラミングによりコンピュータや情報通信ネットワークを活用するとともに，その過程を評価し改善する」[1] など現行の学習指導要領には記載されていなかった内容も追加された．この改訂の背景には現行学習指導要領の成果と課題 [1] より，「情報の科学的な理解に関する指導が必ずしも十分ではない」[1] ことや「情報やコンピュータに興味・関心を有する生徒の学習意欲に必ずしも応えられていない」[1] といった課題があった．さらに,「生徒の卒業後の進路等を問わず, 情報の科学的な理解に裏打ちされた情報活用能力を育むことがいっそう重要となってきている」ことが改訂の理由としてあげられている．これらのことより,「情報 I」においては，すべての高校生に対して成果と課題で挙げたことを解決するようなプログラミングの授業が求められているとわかる．さらに，小学校では 2020 年度よりプログラミング教育が必修となることで，高等学校情報科に対して学習内容としてもより高度なものが求められることが予想される．また，「中学校技術・家庭科技術分野の内容「D 情報の技術」の学習を踏まえたプログラミングを扱う」とあり，小中高のプログラミング教育の接続も求めらている．そのため今後は今までよりプログラミング経験がある生徒が高等学校に入学してくる可能性が十分に考えられる．しかし一方で，現在のカリキュラムにおけるプログラミングにあたる部分は「情報の科学」であるが，情報教育に関する資料 [3] によると「社会と情報」が 80%,「情報の科学」が 20%程度であることから現状として高等学校におけるプログラミング教育は浸透していないことが分かる．また，高等学校共通教科情報の変遷と課題 [2] によると，教科「情報」担当教員の約 3 割が免許外，他教科との兼任は約 5 割となっていることからも，プログラミング経験が十分でない教員が担当していることが予想されるなど，多くの課題を抱えていることが分かる．そこで筆者は，プログラミング学習環境として位置付けられ，web ブラウザのみで動作可能な Google Colaboratory を使用し，初学者向けの授業に適したプログラミング環境を用いて，Python の授業実践を行った．今回はその授業実践と生徒の意識についての報告をする．Python は，近年プログラミングの入門言語として注目されている．その大きい要因としてまず，ライブラリが豊富であり，最近注目されているデータサイエンスなどで使用されていることが挙げられる．また，オフサイドルールによりコードが読みやすく書きやすいものになっている．情報 I の学習内容の構成 (3) において「プログラミング，モデル化とシミュレーション」があり，情報Ⅱの学習内容の構成 (3) においても「情報とデータサイエンス」があり，さらに「情報システム，ビッグデータ」などを扱うことが学習指導要領に明記されている．また，文部科学省が公開している高等学校情報科「情報 I」教員研修用教材 [4] に掲載されているサンプルプログラムには Python が使われているため，学校現場で指導されることが予想される．しかし，一般的な高校におけるコンピュータ教室において，Python の処理系や関連するライブラリを教師自身がインストールすることは困難であると思われるが，Google アカウントがあればインストールをせずに Python 環境を整えることができる Google Colaboratory(以下,「Colaboratory」と記す) を使用することで，環境構築の経験がない教員も授業で扱うことができる．本研究では授業前後に実施した生徒へのアンケート結果と共に授業実践に関する効果を報告する．

2. 高等学校におけるプログラミング教育の現状

今年参加した情報科教育学会 [8] と日本デジタル教科書学会 [9] の学会誌とキミのミラノ発見 [7] を参考にまとめた．

多くの学校で「社会と情報」が選択されていることや情報の専任教員が少ないため，プログラミ

表 1　使用したプログラミング言語

使用言語	実践数
Java Script	6
ドリトル	4
Python，Scratch	3
CSS，HTML，C 言語，VBA，micro:bit	2
DNCL，Java，Excel	1

1 授業において複数言語使用の際はそれぞれ 1 つとカウント

ング言語を使用した実践例は 20 ほどしかなかったと考えられる．また，Web サイトを作成する授業は HTML、CSS、Java Script を使い、ドリトルまたは micro:bit、Scratch と Java Script または C 言語などの組み合わせによる実践例が多くあった．

2.1　Python を用いた他の実践例

表 2　実践例の比較

	使用環境	内容
[10]	Bit Arrow	数値計算 (numpy) と グラフ (matplotlib) の実装
[11]	PowerShell	アンケートの統計分析
[12]	Colaboratory	Python の基本的文法， バブルソートの体験

[11] に関しては，課題に「授業外の負担」とありこれは Python を使用するにあたり環境構築が必要であったためであると考えられる．また，[10] に関しては，“Bit Arrow”[13]，[12] においては今回筆者の実践にも用いた Colaboratory の環境で報告されている．共通することとしては，どちらもオンライン上で Python を実行できることと環境構築が不必要であることである．一方，内容を見てみると教員研修用教材を元にした実践，数学の統計分野との接続を意識した実践，プログラミングの体験であった．さらに，上記の実践例はどれも 2018 年以降で Python の実践例はこれから増えていくと予想できる．

3.　Colaboratory の概要

Colaboratory は，完全にクラウドで実行されるインタラクティブなノートブック環境であり，それは Jupyter ノートブックと呼ばれている [5].

3.1　Colaboratory のメリット

メリットとしては以下のものが挙げられる．

- 設定不要 (環境構築等) である
- Google アカウントがあれば， 無料で使用できる
- チーム内での共有が簡単である

これらのメリットを説明すると，Python の環境構築が不要であり，Numpy など機械学習で必要なほぼ全ての環境がすでに構築されており，さらに GPU(Tesla K80 GPU) も無料で使えること，次に必要なものは Google アカウントで，ブラウザのみですぐに機械学習を始めることが可能であるため情報Ⅱのデータサイエンスなどの題材においても活用できること，Colaboratory で書いたコードは，Google Drive で保存され，グループ内でのノートブックの共有などが非常に簡単で，かつ権限管理など Google Drive 上で行えることである．

4.　実践報告

4.1　対象学年

高校 3 年生 (週 2 コマ (1 コマ : 50 分)), 全 5 コマ
文系： 3 クラス (各 37，34，36 名)
理系： 2 クラス (各 25，27 名)

4.2　生徒状況

プログラミング経験はビジュアル，テキスト言語共に 1 割程度であった．(詳しくは 4.4.1 にて後述)

4.3　実施期間

2019 年 9 月 10 日から 10 月 21 日

4.4　実践環境

- デスクトップ型のパソコンを 1 人 1 台使用できる．
- 入学時に学校から個人の Google アカウントが与えられている．

G Suite for Education が導入されており，情報の授業では Google classroom にて授業のプリントや課題の配布等の管理を行っている．他の科目においても活用されている．

4.5　授業内容

表 3　実践授業内容

授業数	内容
1	Python とは，四則演算，比較演算子の学習
2	変数，リストの学習
3	分岐 (if,elif,else) の学習
4	反復 (for,range) の学習
5	最終課題

主な授業の流れは，最終課題を数学の整数問題に設定し，その問題を解決するためにプログラミングの基本的な考え方である逐次･分岐･反復処理のツールを 4 回の授業に分けて学んでもらい，5 回目の授業で最終課題に取り組むという形で行った．各授業の流れとしては内容ごとの概念を説明し，簡単な問題を一緒に実行した後，生徒一人ひとりにその内容に関する練習問題を取り組ませた．また，プログラミング初学者が課題に対して論理的に思考を適用した課題解決が行えるようにするための教材とした．

4.5.1　1 回目

社会における Python の活用例の紹介，プログラミング言語における Python の特徴等の説明を行った．四則演算に関しては，まずはじめに「1+1」のコードを入力させながら，Colaboratory の基本的な使い方の説明も行った．四則演算の演算子を提示し，以下の問題を練習問題とした．

練習問題

問題 1：10−2
問題 2：10 × 2
問題 3：10 ÷ 2
問題 4：11 ÷ 2 の余りの表示
問題 5：2^{10}

解答コード例

```
[1] 10-2 #問題1
    8
[2] 10*2 #問題2
    20
[3] 10/2 #問題3
    5.0
[4] 11%2 #問題4
    1
[5] 2**10 #問題5
    1024
```

図 1　四則演算　解答

授業中や授業後に提出された生徒のコードを確認すると，問題 3 と 4 において使用する演算子の違いがわからない生徒が見られた．また，問題 5 に関しては「*」を 2 つではなく，2 の後ろに「*」を 10 個つけている生徒もおり，さらに，キーボードで「*」,「%」の位置がわからない生徒も一部存在した．

比較演算子に関しては以下のような練習問題とした．

練習問題

問題 1：2==2
問題 2：2!=2
問題 3：2>2
問題 4：2>=2
問題 5：i =100
　　i%2==0
問題 6：i =101
　　i%2==1

解答コード例

比較演算子に関してはコードを写す形にし，返ってきた答えに対してどうしてそのような結果が返ってくるのかを考えさせ，隣同士で確認するようにした．問題 5，6 において，はじめはなぜ“true”や“false”が返ってくるのかわからない生徒もいたが隣同士で説明し合うことで理解していた．

四則演算と比較演算子を実行するにあたり共通することとしては“SyntaxError: invalid character in identifier”である，数字や「+」,「=」などの記号が半角ではなく，全角入力されていることで起

```
[1]  2==2 #問題1
     True
[2]  2!=2 #問題2
     False
[3]  2>2 #問題3
     False
[4]  2>=2 #問題4
     True
[5]  i = 100     #問題5
     i % 2 ==0
     True
[6]  i = 101     #問題6
     i % 2 ==1
     True
```

図 2　演算子　解答

こるエラーが多数見られた．初めに「1+1」を実行するところとそれを全角で入力した際には同様の“SyntaxError”エラーが起こるということを説明していたが，助けを求める生徒もいた．

4.5.2　2 回目

変数に関する練習問題は以下のようにした．

練習問題

問題 1：name1=”自分の姓”

問題 2：name2=”自分の名”

問題 3：type(name1)

問題 4：type(total)

解答コード例

問題 4 に関しては，変数の導入の際に total には，数字を入力させる練習を行った．変数において type まで意識させた点は，次にリストの学習をさせる際に「string 型」と「integer 型」は，「integer 型」を「string 型」に直して実行しないとエラーが起こってしまうためであるのでここで学習させることとした．ここでのエラーとしては，一度「type=name1」を実行してから間違いに気づき，「type(name1)」を実行し，“’str’ object is not callable”とエラーがでてしまい，どうしてエラーが起こったのかわからない生徒が存在した．

リストに関する練習問題は以下のようにした．

```
[1]  total=100+200+300
[2]  name1="山本"
     name1
     '山本'
[3]  name2="周"
     name2
     '周'
[4]  type(name1)
     str
[5]  type(total)
     int
```

図 3　変数　解答

練習問題

- 名前：○○○○
- 生年月日：○月○日
- 性別：男性
- 趣味：○○○○○

解答コード例

```
[1]  list=["山本","周",4,25,"男性","ソフトテニス"]
[2]  print("名前:"+list[0]+list[1])
     print("生年月日:"+str(list[2])+"月"+str(list[3])+"日")
     print("性別:"+list[4])
     print("趣味:"+list[5])
     名前:山本周
     生年月日:4月25日
     性別:男性
     趣味:ソフトテニス
```

図 4　リスト　解答

空のリスト，リストの先頭は 0 から始まることの説明を行った後，リストの中から自分が取り出したいものを選んで，表示させる方法の説明後，自分の好きな情報 (部活や趣味) をリストに追加し表示させるという練習問題を各自で行った．ここで生年月日を表示させる際に変数で行った“string”と“integer”の内容が使われるようになっており，配布したプリントを見て解決する生徒もいた．さらに初めは気が付かずに実行したが，エラーを見て解決した生徒も多くみられた．理由としては，エラーが“must be str, not int”と比較的わかりや

すいエラーメッセージであったことが挙げられる．

4.5.3　3 回目

if に関する練習問題は以下のようにした．

練習問題

80 点以上・・・「優」
60 点以上〜80 点未満・・・「良」
60 点未満・・・「不可」
を表示するようなプログラムの作成

解答コード例

```
[1]  x=80
     if x>=60:
       print("合格")
     else:
       print("不合格")

     合格

[2]  x = int(input("x="))
     if x>=80:
       print("優")
     elif x>=60:
       print("良")
     else:
       print("不可")

     x=90
     優
```

図 5　if 文　解答

if 文の概念を説明し，教員研修用教材 [4] の分岐のコードを入力し，コード内容の説明をした．ここからは，穴埋め式の見本コードも Google Classroom 上に用意した．また，練習問題としては “elif” を使い，3 つ以上の分岐するような練習問題とした．さらに “input” も教えることでコードの実行後，いろいろな値に変えた値を入力し実行結果より自分のコードが正しく書けているかを確認させた．ここでのエラーとしては，インデントによるエラーが多かった．Python には可読性を高めるためにインデントがあるが，今回はそれによりコードはあっているにもかかわらずエラーが返ってきてしまい，困惑してしまう生徒がいた．

4.5.4　4 回目

for 文に関する練習問題は以下のようにした．

練習問題

問題 1：0,1,2,3,4,5,6,7,8,9,10 を表示
問題 2：5,6,7,8,9,10 を表示
問題 3：1,3,5,7,9 を表示
問題 4：問題 3 と別の書き方
問題 5：1 から 20 までの中で，3 の倍数を表示
問題 6：問題 5 に加えて，3 の倍数の中でさらに 2 の倍数の時は，数字を出力させた後，”6 の倍数！” と表示

穴埋めコード

```
for i in range(?):
  print(i)
```

図 6　for 文　穴埋めコード

解答コード例

```
[1]  for i in range(11):
       print(i)

     0
     1
     2
     3
     4
     5
     6
     7
     8
     9
     10
```

図 7　for 文　解答

for 文の概念を説明し，最終問題を解く際に使用する “range” の仕組みを説明し，練習問題を行わせた．また，if 文の時と同様に穴埋め式の見本コードを用意した．「range」を使い最後の数字を表示する際には一つ先の数字をコードに書く必要があるが，リストの理解が不十分な生徒はなぜ一つ先の数字を入れる必要があるのかわからない生徒が多数見られた．しかし，穴埋め式のコードを用いて，トライアンドエラーを繰り返しながら行うことで多くの生徒が全ての課題に取り組むことがで

きていた．

4.5.5　5 回目

最終問題は以下のように設定した

最終問題

3 以上 9999 以下の奇数 i で，$i^2 - i$ が 10000 で割り切れるものを全て求めよ．

解答コード例

```
[1]  for i in range(3, 10000):
       if i % 2 == 1:
         if (i**2-i) % 10000 == 0:
           print(i)

     625

[2]  for i in range(3, 10000, 2):
         if (i**2-i) % 10000 == 0:
           print(i)

     625
```

図 8　最終課題　解答

最終課題は東京大学の入試問題 [6] を取り扱ったが，初めは入試問題であることは伏せて行った．授業の最後に入試問題であることを伝えると特に文系クラスにおいて反応がよかった．最終課題の流れとしては，予めフローチャートを用意し，前回の復習とフローチャートの練習を兼ねて，for 文の練習問題をはじめに行った．始めに 15 分間の時間を取り，見本の穴埋めフローチャートがなくてもフローチャートが書ける人は，白紙の部分にコードの流れを書き，そうでない人は穴埋め式のフローチャートを使い，アルゴリズムを考えさせた．その後，コードを書くように授業設計を行った．コードの解答例としては，if 文を 2 回使うものと「range」を用いる 2 パターンであった．多くの生徒が前者である if 文を 2 回使うコードであった．理由としては，前の授業で行なった「問題 6」が最後にあったためであると考えられる．さらに，机間指導していく中で最終課題は幾つの条件から出来ているかと声かけをしたことも理由として挙げられる．また，「(i**2-i)」の「()」がない生徒も多く見られた．さらに，「()」がない場合においてもエラーが返ってこないため，間違いに気がつく生徒は少数であった．また，この問題は数学の問題として解くと難解で解答が長いが，Python で行うと短いコードであれば 4 行で書くことができることから，プログラミングの有用性や面白さを実感している生徒が多く見られた．

5.　授業アンケートから見る生徒の実態

授業アンケートはプログラミングの授業の開始する 1 番初めと前半授業 5 回終了後に行った．

5.1　事前アンケートの項目

Q1:プログラミングに興味があるか (4 択)

Q2:ビジュアルプログラミングをしたことがあるか (スクラッチ, ビスケット, グーグルブロッキーなど)(2 択)

Q2':Q2 にてビジュアルプログラミング経験したことがあると回答した人に対して，順次構造・分岐構造・反復構造の経験があるか (3 択)

Q3:テキストプログラミングをしたことがあるか (C++、Java、python など)(2 択)

Q3':テキストプログラミング経験したことがあると回答した人に対して，順次構造・分岐構造・反復構造の経験があるか (3 択)

5.2　事後アンケートの項目

Q4:授業を受けてプログラミングに興味を持ったか (4 択)

Q5:プログラミングは楽しかったか (6 択)

Q6:四則演算・リスト・分岐構造・反復構造・分岐と反復の組み合わせ問題それぞれに対する難易度 (5 択)

Q7:プログラミングを行い、達成感を得られたか (2 択)

Q7':Q7 において達成感が得られたと回答した人に対して，具体的には何があるか (自由記述)

Q8:プログラミングの授業が終わってもプログラ

ミングをしてみたいか (5 択)

Q9:プログラミングをしていて大変だな、または難しいなと思うところはあったか (2 択)，大変, 難しいと感じた人は具体的には何があるか (自由記述)

Q9':Q9 において大変, 難しいと回答した人に対して，具体的には何があるか (自由記述)

Q10:情報の授業以外で自分でプログラミングの勉強をしたか (情報の授業を行ってから)

Q11:今後プログラミングでしたいことはあるか (あれば具体的に)(自由記述)

5.3 集計結果

全てのアンケートは欠損値を除いた n=145 で集計を行なった.

5.3.1 対象生徒のプログラミング学習経験 (Q2,2',3,3')

実践前に生徒に行ったアンケート調査によると，「ビジュアルプログラミングをしたことがあるか (スクラッチ, ビスケット, グーグルブロッキーなど)」(n=145) において「ある」と答えた生徒が 13.1%(19 人),「ない」と答えた生徒が 86.9%(126 人) であった．また,「テキストプログラミングをしたことがあるか (C++、Java、python など)」(n=145) において「ある」と答えた生徒が 10.3%(15 人),「ない」と答えた生徒が 89.7%(130 人) であった．さらに，文系理系に関して T 検定を行うと，Q2 の P 値が 0.589(>0.05)，Q3 の P 値が 0.240(>0.05) となり，有意差が認められなかったため文系理系は同じ傾向にあることがわかった．また，それぞれの質問において「ある」と答えた生徒に対して,「順次処理」,「分岐構造」,「反復構造」の経験があるかと聞いたところほとんどの生徒がわからないと解答していた.

5.3.2 授業前後におけるプログラミングへの興味 (Q1,4)

授業の前後におけるプログラミングに対する興味の変化を見ると,「興味がある」は 31 ポイント上昇し,「興味がない」は 9 ポイント減少した．さらに,「興味がある」と「どちらかと言えば興味がある」を合わせると，50.3%から 80.7%と 33.4 ポイント上昇している．この結果からプログラミング初学者への興味を育むことができた授業内容であったと言える.

	興味がある	どちらかと言えば興味がある	どちらかと言えば興味がない	興味がない	合計
前(Q1)	20.0%(29)	30.3%(44)	22.1%(32)	27.6%(40)	100.0%(145)
後(Q4)	51.0%(74)	29.7%(43)	2.8%(4)	16.6%(24)	100.0%(145)

図 9　授業前後におけるプログラミングへの興味

5.3.3 プログラミングをしていて大変だな、または難しいなと思うところはあったか (Q9,9')

プログラミングをしていて大変だな、または難しいなと思うところはあったか	あった	特になかった	合計
	81.4%(118)	18.6%(27)	100.0%(145)

図 10　プログラミングをしていて大変だな、または難しいなと思うところはあったか

数式などの記号を覚えること，エラーの修正 (半角)，エラーの内容がそもそもわからない，タイピング，インデント，コードの理解，問題の解答としてはおかしいがコードによるエラーはなかったためエラーが返ってこなかった ((a**2-a)%10000 の () が必要であったこと) など様々挙げられた．Python のコードのルールは今回の授業で経験してもらうことも目的であったのでよかったが，数学的な要素やコンピュータの操作によるものなどに関してはなるべく差が生まれない状態にしたい.

5.3.4 プログラミングは楽しかったか (Q5)

プログラミングは楽しかったか	
とても楽しかった	24.8%(36)
楽しかった	29.0%(42)
どちらかと言えば楽しかった	26.2%(38)
どちらかと言えば楽しくなかった	4.8%(7)
楽しくなかった	5.5%(8)
全く楽しくなかった	9.7%(14)
合計	100.0%(145)

図 11　プログラミングは楽しかったか

「とても楽しかった」,「楽しかった」が 53.8%,「楽しくなかった」,「全く楽しくなかった」が 15.2% であったことから，半数以上の生徒が今回の授業においてプログラミングを楽しむことができたと回答した．また，楽しむことが出来なかったと回答した生徒の多くは Q9 においてエラーが何度も起こってしまうことや課題の中に苦手な数学があったことが挙げられていた．

5.3.5　プログラミングの授業が終わってもプログラミングをしてみたいか (Q8)

プログラミングの授業が終わってもプログラミングをしてみたいか	
ぜひしたい	15.9%(23)
機会があればしたい	33.8%(49)
どちらかと言えばしたい	19.3%(28)
どちらかと言えばしたくない	14.5%(21)
全くしたくない	16.6%(24)
合計	100.0%(145)

図 12　プログラミングの授業が終わってもプログラミングをしてみたいか

「ぜひしたい」,「機会があればしたい」が 49.7% とあり，肯定的な回答が半数近くあることから初学者におけるプログラミング例としてはよかったと考える．

5.3.6　プログラミングを行い、達成感を得られたか (Q7,7')

プログラミングを行い、達成感を得られたか	得られた	得られなかった	合計
	75.9%(110)	24.1%(35)	100.0%(145)

図 13　達成感を得られたか

「練習問題，最終課題が解けた時」,「コードエラーを直せた時」,「トライアンドエラーを繰り返している時」,「エラーなく正しいコードを打てた時」,「コードの意味がわかった時」などが多数挙げられた．問題が解けた際に「できた！」など喜んでいる姿を多数見ることができたことからも多くの生徒が達成感を得ることができたと考える．

下記の内容の難易度	[四則演算](n=145)	[リスト](n=145)	[if文](n=145)	[for文](n=145)	[if文とfor文の組み合わせ](n=145)
簡単	33.1%(48)	24.8%(36)	17.2%(25)	13.8%(20)	7.6%(11)
やや簡単	17.9%(26)	17.9%(26)	12.4%(18)	11.0%(16)	6.2%(9)
普通	22.8%(33)	28.3%(41)	29.0%(42)	25.5%(37)	15.2%(22)
やや難しい	5.5%(8)	7.6%(11)	15.9%(23)	20.7%(30)	22.8%(33)
難しい	20.7%(30)	21.4%(31)	25.5%(37)	29.0%(42)	48.3%(70)

図 14　各項目の難易度

5.3.7　各項目の難易度に関する反応 (Q6)

図 14 によると問題の難易度としては授業回数の増加に応じて，難しいと思う割合も増加し，一方で簡単だと感じる割合が減少している．この結果から授業の構成として適切であったと言える．また，if 文と for 文の組み合わせにおいて難しいと感じている生徒の割合がとても多く，授業中においても悩んでいる生徒が多くいた．よって，問題練習を行う時間を増やしたり，課題をスモールステップにする必要があると考える．

5.3.8　情報の授業以外で自分でプログラミングの勉強をしたか (情報の授業を行ってから)(Q10)

授業以外でプログラミングをした生徒は 9 人であり，「GAS を使ってみた」,「配布した資料を見た」,「インターネットにてプログラミングに関することを調べた」など具体的な回答の生徒は事前アンケートにて過去にプログラミング経験があると答えていた．高 3 ということもあり，授業で疑問に残ったことや気になったことを調べたり，配布資料をもとに復習することはほとんどなかったと考える．

5.3.9　今後プログラミングでしたいことはあるか (Q11)

web ページの作成，ゲームの作成，アプリケーションの作成，データの集計，ハッキング，レゴを使ったプログラミングなどが挙げられた．また，何かしたいが何ができるかわからないという回答も多数見られた．

5.4　まとめ

web ブラウザのみで動作可能な Google Colabo-

ratory を使用し，初学者向けの導入授業を行なった．生徒の実態から Python を用いて論理的に思考を適用した課題解決が行うことができたと考える．また教師の立場から考えると，Python を行うための環境設定は一切なかったことから負担が少なかった．また，G Suite が導入されており，Google classroom にて課題や見本コードの配布，生徒の作成物の管理が大変楽であった．生徒においては，「難しかったが，楽しかった」や「達成感があった」など肯定的な声が多数あり，導入の授業としては適していることがわかった．一方で今回学んだことが何につながるのかわからないという声もあり，今後は学んだことが何かにつながることを意識できるような課題や Python の豊富なライブラリを活用した機械学習を含むような内容にする必要がある．さらに，Google Colaboratory の共同作業ができる機能を使った授業例を検討していきたい．

参考文献

[1] 文部科学省．(2018). 新学習指導要領解説　. http://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2019/03/28/1407073_11_1_1.pdf(2019 年 11 月 1 日確認)

[2] 中野 由章. 高等学校共通教科情報科の変遷と課題. 2018. https://www.ipsj.or.jp/magazine/9faeag0000005al5-att/5910peta.pdf(2019 年 11 月 1 日確認)

[3] 文部科学省．情報教育に関する資料．2015. http://www.mext.go.jp/b_menu/shingi/chukyo/chukyo3/059/siryo/__icsFiles/afieldfile/2015/11/11/1363276_08_1.pdf(2019 年 11 月 1 日確認)

[4] 高等学校情報科「情報Ⅰ」教員研修用教材. 2013.http://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2019/10/09/1416758_005.pdf

[5] **Colaboratory** へようこそ. 2013.https://colab.research.google.com/notebooks/welcome.ipynb?hl=ja(2019 年 11 月 1 日確認)

[6] 東京大学 (理系) 前期入試問題.2005.https://www.densu.jp/tokyo/05tokyospass.pdf(2019 年 11 月 1 日確認)

[7] キミのミライ発見 教科「情報」 授業例 .https://www.wakuwaku-catch.net/%E6%8E%88%E6%A5%AD%E4%BA%8B%E4%BE%8B-%E3%83%90%E3%83%83%E3%82%AF%E3%83%8A%E3%83%B3%E3%83%90%E3%83%BC/(2019 年 11 月 27 日確認)

[8] 情報科教育学会．第 12 回全国大会講演論文集．2019.

[9] 日本デジタル教科書学会．第 8 回年次大会発表予稿集．2019.

[10] 本多佑希．オンラインプログラミング環境「**Bit Arrow**」の **Python** 対応. キミのミライ発見. https://www.wakuwaku-catch.net/jirei19133/(2019 年 11 月 27 日確認)

[11] 阿部百合．問題解決への利用を目的とした統計の学習 **(Python** を利用して**)**. 2018. http://www.johobukai.net/20181227/181227_03_01.pdf(2019 年 11 月 27 日確認)

[12] 遠藤優一．**2020** 年度必修化に向けた高等学校プログラミング授業の実践報告．デジタル教科書学会．pp. 43-44. 2019https://www.wakuwaku-catch.net/%E6%8E%88%E6%A5%AD%E4%BA%8B%E4%BE%8B-%E3%83%90%E3%83%83%E3%82%AF%E3%83%8A%E3%83%B3%E3%83%90%E3%83%BC/(2019 年 11 月 27 日確認)

[13] オンラインプログラミング学習環境 **Bit Arrow**. https://bitarrow.eplang.jp/(2019 年 11 月 27 日確認)