オンラインプログラミング環境Bit Arrowにおける Python処理系

長 慎也[1,a)]　長島 和平[2]　間辺 広樹[3]　兼宗 進[4]　並木 美太郎[2]

概要：
筆者らは現在，Web ブラウザを用いてプログラミング学習が可能な環境「BitArrow」を，初学者向けのプログラミング学習活動に用いている．BitArrow は，複数のプログラミング言語の学習ができるように設計されており，これまで，JavaScript，C 言語，ドリトルなどでのプログラミング学習に対応してきた．本発表では，BitArrow に Python を動作させる仕組みの提案と，動作可能なプログラムの事例を紹介する．
プログラミング学習において，Python 言語が注目されている理由の一つに，統計処理などのライブラリが充実していることが挙げられる．BitArrow でこれまでに実装した言語処理系では，処理のほとんどを Web ブラウザ側で実行するようにしてきたが，前述のようなライブラリを容易に利用できることを考慮してサーバーサイドでの実行も行えるようにした．

Learing C programming using “Bit Arrow”, an online programming environment.

CHO Shinya[1,a)]　NAGASHIMA Kazuhei[2]　MANABE Hiroki[3]　KANEMUNE Susumu[4]
NAMIKI Mitarou[2]

1. はじめに

筆者らは，プログラミング学習環境として，Web ブラウザのみで動作可能な BitArrow[1] を開発し，初学者向けの授業におけるプログラミング環境に用いる実践を行っている．BitArrow は現在，JavaScript，C 言語，ドリトル，DNCL，Python で書かれたプログラムを実行することが可能であり，今回の発表では，Python の処理系の実装状況を報告する．

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．その大きい要因の 1 つに，ライブラリが豊富であることが挙げられる．特に，統計処理のための scipy[2] や numpy[3] などを用いることで，最近注目されているデータサイエンスなどの題材を手軽に実践できる．

高校の学習指導要領 [4] においても，「データサイエンス等に関する内容を大幅に充実」させ，教科「情報」の科目である「情報 I」では「プログラミング，モデル化とシミュレーション」を扱い，「情報 II」では「情報システム，ビッグデータ」などを扱うことが明記されており [5]，これらの学習を行う環境として Python は最適であると考えられる．

しかし，一般的な高校における教室環境において，Python の処理系や，関連するライブラリ群をインストールすることは困難であるため，Web ブラウザだけで動作させることができる BitArrow において Python が動作することで，高校においても広く Python を使った学習ができるようになると考えられる．

今回開発した Python の処理系を用いることで，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 [6] に掲載されている Python プログラムの大部分

[1] 明星大学
Meisei University, Japan
[2] 東京農工大学
Tokyo University of Agriculture and Technology, Japan
[3] 神奈川県立柏陽高等学校
Hakuyo High School, Japan
[4] 大阪電気通信大学
Osaka Electro-Communication University, Japan
[a)] cho@eplang.jp

を BitArrow 上で動作させることができるようになった．

2. BA-Python の設計

BitArrow では，これまで実装してきた JavaScript，C 言語，ドリトル，DNCL の処理系ではいずれも，学習者の書いたプログラムを JavaScript に変換してブラウザ上で動作するようにしている．その理由として，学習者にとって興味ある題材であるアニメーションやゲームなどのインタラクティブ性の高い作品にも対応できるようにすることや，サーバ側の設置の手間や負荷を減らすことなどが挙げられる [7]．

一方で，JavaScript に変換してブラウザ上で動作するプログラムから各種ライブラリを使う場合，ライブラリ自身も JavaScript で実装する必要がある．しかし，先ほど述べたように，Python の魅力は既存のライブラリにあるといってもよく，ライブラリの機能を十分に活かし，かつそれらをブラウザ側で実行するためには，それらのライブラリを JavaScript で実装しなおす必要が出てくる．これには膨大な労力がかかると予想される．

また，統計処理においては，入力データを与えて，それに対応する出力結果を得る，というインタラクションはあるものの，その頻度は 1 回の実行につき数回程度（多くの場合 1 回だけ）であることが多い．ゲームなどで，1 秒間に数十回のインタラクションをすることと比較すれば，リアルタイムなインタラクションは必ずしも必要でないことが多い．したがって，ブラウザ側で動かす必要性が比較的乏しい題材ということもできる．

そこで，BitArrow で動作する Python（以下 BA-Python と呼ぶ）では，サーバ側での実行もサポートできるよう，次のような要件を満たすように設計した．なお，以降「Python 処理系」は BA-Python ではない従来の Python 処理系 (公式サイト [8] で配布されているもの) のことを指すものとする．

- **危険なプログラムを動作させないようにすること**
  Python 処理系は，ファイルなどの各種リソースに直接アクセスする API をもっているため，悪意のあるなしにかかわらず，学習者がプログラムを通じてサーバのリソースを破壊・漏洩させる可能性を考慮しなくてはならない．したがって，API の使用を制限したり，実際の API の動作に変更を加えたりして，不用意なアクセスから守る必要がある．
- **サーバの負荷をかけすぎないようにすること**
  学習者が一斉にサーバ上でプログラムを動作させると，同時に多数のプロセスが起動してサーバに負荷をかける可能性がある．特に統計処理は処理がかかるものが多い．また学習者が誤って無限ループのプログラムを実行した場合も負荷が上がる．このため，サーバに負荷がかかりそうな場合は適切に実行させるプロセスの実行時間や個数を制限する必要がある．
- **プログラムのエラーをブラウザ側でチェックできること**
  文法や意味が誤っていてエラーになるプログラムをサーバに送信すると，Python の処理系を起動してしまうことになりサーバへの負荷となりうる．特に，すでに多数の Python 処理系のプロセスがサーバ上で動いているときは，誤ったプログラムを送信してもすぐにエラーであることが返ってこない可能性がある．このためデバッグにかかる時間が増えてしまう可能性がある．このため，ブラウザ側でエラーのチェックを行うことによって，サーバへの負荷を低減させることが望ましい．
- **学習者独自のデータを扱えるようにすること**
  統計処理には，入力となるデータ（表形式や画像など）が必要になることが多い．これらのデータを学習者が独自に用意し，プログラムを通じて解析することで，学習者の興味にあった題材で学習ができると考えられる．また教員が用意したデータをクラス全体で共有する仕組みも必要と考えられる．
- **グラフ機能を利用できること**
  統計処理においては，解析した結果を数値で表示するだけでは全体を把握することは難しい．結果をグラフに表示することでより結果を直観的に理解できるようになるとともに，学習者への興味を喚起することにもつながる．したがって Python のグラフライブラリである matplotlib（pyplot）[9] などが利用できることが望ましい．
- **従来の BitArrow 同様，ブラウザ側での実行もサポートすること**
  先述の通り，サーバでの実行はサーバへの負荷がかかるため，Python 処理系が用意しているライブラリを使う場合を除いては，ブラウザ側で実行させたほうが軽快に動作する．例えば，ライブラリを使う前提として Python 言語の基礎を習ったり，Python でゲームアプリを作成したり [10] するような使い方も考えられる．したがって，BitArrow でサポートする他の言語同様，Python で書かれたプログラムを JavaScript に変換してブラウザ側で動作させる，という選択肢も残す必要がある．

3. BA-Python の実装

これまで，BitArrow では学習対象となる各種言語の処理系はすべて JavaScript（JS）で記述され，学習者が書いたプログラムは処理系により JavaScript に変換（トランスパイル）され，すべて Web ブラウザで実行されていた．

BA-Python でもこれまでと同様，Python のトランスパイラを実装した．これにより**図 1** のように Python を

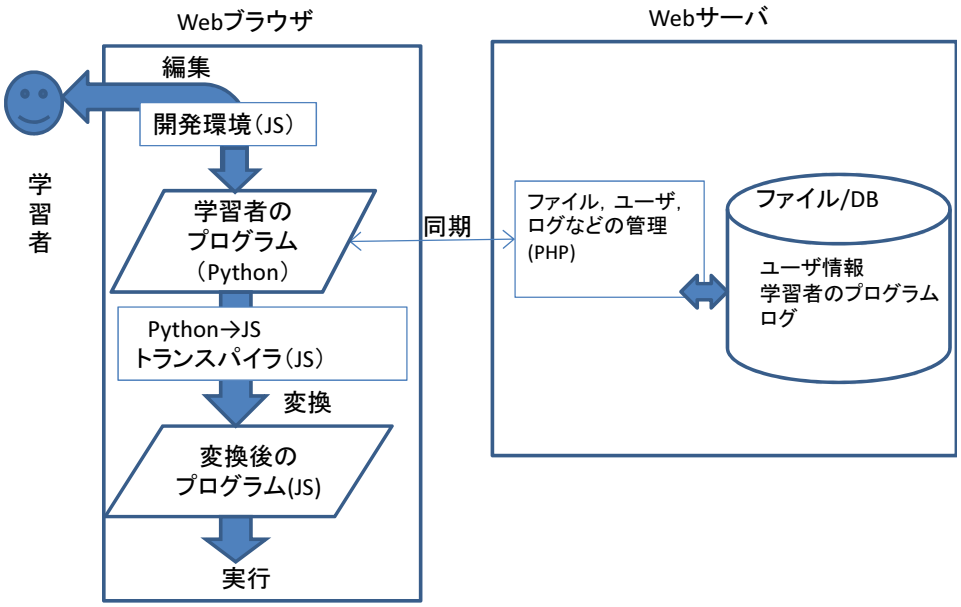

図 1　BA-Python のブラウザ側での実行

JavaScript に変換してブラウザ上で実行することも可能であるが，加えてサーバ側での実行もサポートするようにした．動作の流れを図 3 に示し，次にサーバ側の動作の詳細を示す．

なお，プログラムをブラウザ側とサーバ側のどちらで実行するかは，ユーザ自身にも意識させる必要があると考え，図 2 のようにユーザに明示的に選択させている．

図 2　Python の実行メニュー

3.1　エラーチェッカ

先述した通り，エラーのあるプログラムはサーバに送信する前にチェックして，学習者に即座にフィードバックすることが望ましい．エラーチェッカは，学習者の書いたプログラムについて，構文のエラーや，未定義の変数や関数の使用などの意味的なエラーをチェックして，エラーがあれば学習者のブラウザの画面に提示する機能をもつ．

さらに，BA-Python では，危険なプログラムを動作させるのを防ぐため，使用できる組み込みの関数やライブラリに制限を持たせており．BA-Python で許可していない関数やライブラリを呼び出した場合もコンパイルエラーになるようにしている．

エラーチェッカはブラウザ側で実行されるため，エラーのあるプログラムを実行させようとしても，サーバ側に負荷をかけることがなく，学習者へのエラーの通知も即座に行うことができる．

3.2　Python 実行器

エラーチェッカによってエラーがないと判断されたプログラムは，サーバに送信されてサーバ側に設置された Python 実行器を使って実行される．

Python 実行器には，ブラウザ側と同一のエラーチェッカをもっており，受け取った Python プログラムをもう一度チェックする．これは，ブラウザ側のエラーチェッカを経ずにサーバへのリクエストを直接生成する手法で不正なコードが送信される可能性もあるためである．エラーチェッカによってエラーがないと判断された後，Python 処理系（Python3）を用いて実行を行う．

さらに，実行のリクエストが集中してサーバの負荷が上がるのを防ぐため，一定数の Python 処理系のプロセスが起動していたら実行を中止させる．また，すでに起動している Python 処理系のプロセスが一定時間経過しても終了しない場合は，強制終了させる仕組みも用意されている．

エラーチェッカはすべて JavaScript で実装されており，ブラウザ側ではブラウザの JavaScript 処理系を用い，サーバ側では node.js を用いて動かしており，コードの内容はほとんど同一である．

3.3　ラッパーライブラリ

BA-Python ではサーバ側で学習者の Python プログラムを実行するときに，呼び出すことができるライブラリに

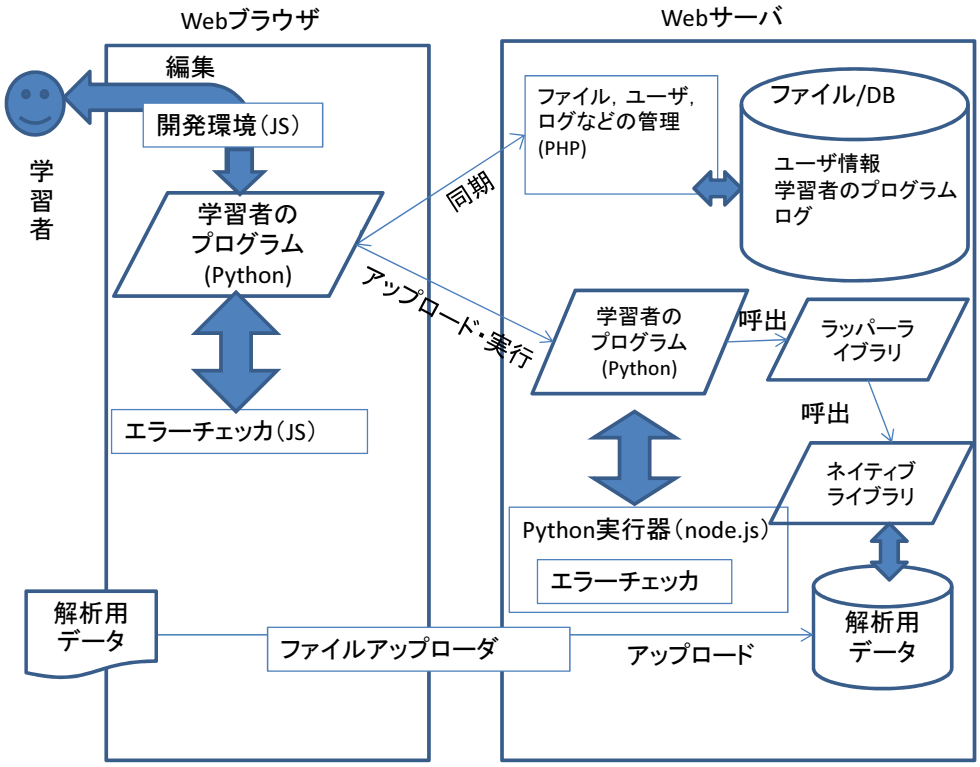

図 3　BA-Python のサーバ側での実行

制限を持たせている．これは，Python 処理系に用意されているライブラリ (ネイティブライブラリ) をラッパーライブラリを通じて間接的に呼び出すことで実現する．ラッパーライブラリは，ネイティブライブラリの中で，呼び出しても安全と判断される特定のメソッドのみを呼び出せるようにしている．

現在，Python 処理系と共通して BA-Python で使える関数は次のものになる．

- 組み込み関数は `range`, `input`, `str`, `int`, `float`, `len`, `type`, `quit`, `exit`, `sorted` が使用できる．
  - `print` については，Python2 と Python3 で記法が異なるが、いずれの記法でも動作するようになっている（図 4 参照）．
  - `input` については，ブラウザ側で実行した場合は `input` を呼び出すたびにブラウザ標準のダイアログで入力をさせる．サーバ側で実行する場合は入力される予定の値を実行前にすべて入力しておく必要がある（したがって，`input` が何回呼ばれるか事前にわからないプログラムは正しく実行できない）．
- ライブラリは，`datetime`, `random` ,`math`, `re` が全機能を利用できる．`os`，`matplotlib`(`pyplot`), `numpy` は一部のメソッドが利用できる．詳しくは BitArrow のサイト [11] を参照されたい．

また，いくつかのメソッドにおいては，ネイティブライブラリの入出力を加工するものがある．

```
print("hello") #python 2,3 で共通
print "hello" #python2 のみ
print "hello", #python2 のみ．改行なし
print("hello",end="") #python3　改行なし
```

図 4　print の記述方法

- ファイル操作系のメソッドは，次に述べるファイルアップローダでアップロードできる領域のみにアクセスできるようパスの内容を加工してからネイティブライブラリに渡している．
- matplotlib におけるグラフ表示命令は，通常はサーバ側のグラフィックス画面 (X-window サーバなど）へ表示を行うが，BitArrow においては結果をブラウザに表示させる必要があり，出力を一旦画像ファイルに変換してブラウザに送信している．

3.4　ファイルアップローダ

ファイルアップローダ（図 5）は，解析に用いるデータ（解析用データ）を学習者や教員が自由にアップロードできる機能である．

アップロード先は「user」と「class」がある．user は各学習者が自分のプロジェクト内からアクセス可能なファイル，class は教員が管理するクラスの受講者が共通してア

図 5 ファイルアップローダ

クセス可能なファイルである．これにより，学習者が自分のデータで独自の分析を行ったり，教員が分析用のデータを学習者に配布したりすることが可能になる[*1]．

4. 関連研究

統計処理やプログラミングを学習する枠組として，プログラミング言語 R による授業の提案が行われている [12]．この提案の中では R のメリットとして「打ち込めばすぐに結果が得られる」「身の回りのデータを簡単に解析できる」「汎用的なプログラミング言語であり，プログラミング教育にも使える」「ダウンロードすれば管理者権限がなくても起動する」という点を挙げている．R と Python（のライブラリである numpy，matplotlib など）は提供している API が似ており，Python でも R と同様の簡便さで統計処理を行える点は変わらず，またプログラミング言語教育にも十分利用できるものである．さらに，BA-Python は Web アプリケーションであるためダウンロードすら不要で利用できる点で，R より手軽に利用できると考えられる．Web アプリケーションはダウンロードが不要である一方で，ネットワーク環境は必須であるというデメリットも存在するが，Web アプリケーションをオフラインで動作するアプリケーションに変換する仕組み [13] も存在するため，環境に合わせて柔軟に対応することが可能である．

Python や R を用いた統計処理を学習させるシステムも提案され [14]，Web サイトに公開されている [15]． このシステムでは Python や R の統計処理プログラムを教員が作問し，学習者に解答することができ，作問と解答の作業を Web ブラウザ上で行えるようになっている．正誤判定は Python や R のプログラムをサーバ上で実際に実行させて行っている．ただ，サーバ上の実行環境はあくまで正誤判定のためのものであり，実際の学習は学習者の端末に Python や R の処理系をインストールさせて使用させることを推奨しているため，すべての学習を Web ブラウザ上で行うことは想定していないと考えらえる[*2]．

[*1] 今回の発表とは直接関係ないが，ファイルアップローダを利用することで，BitArrow 上の JavaScript(HTML) プログラムにおいても学習者が独自の画像を使用できるようになった．

[*2] 2019 年 5 月現在，サイトを試用したところサーバ上でプログラムを実行させようとすると必ずエラーになるので，詳細は不明である

他にも，Web ブラウザ上で Python を実行できる環境も多数存在している [16][17][18][19][20] が，筆者がこれらの環境を試したところ，どの環境も 2 章で述べたような要件を満たしていなかった．その理由として，まず教員や学習者が独自に用意したデータにアクセスできない点が挙げられる．ファイルへのアクセスが最初から禁止されている環境や，一時的な保存はできてもしばらく経つとすぐに消えてしまう環境などがあった．さらに，numpy や matplotlib などのライブラリに不備がある点も挙げられる．そもそも当該ライブラリがインストールされていなかったり，グラフを表示しようとするとエラーが発生 (X-window サーバへのアクセスに失敗するのが原因？）したりする環境などが目立った．

5. サンプルプログラム

本章では，現段階の BA-Python の実装で動作する，文部科学省の教員研修用教材 [6] が提示している「コンピュータの仕組み」「アルゴリズムとプログラミング」「モデル化とシミュレーション」などのサンプルプログラムをいくつか紹介する．

- 「コンピュータの仕組み」では，コンピュータ内部での数値の表現について解説する中で，Python を用いて数値のオーバーフローを体験させるプログラムが載っている．これは，従来の Python 処理系で実行する想定で書かれており，サーバ側で実行する必要がある（ブラウザ側で JavaScript に変換した場合，JavaScript の数値表現に影響されて結果が異なる可能性がある）．
- 「アルゴリズムとプログラミング」では，探索や並び替えなどのアルゴリズムを Python で実装させ，データの個数に応じて並べ替え回数などの性能がどう変わるかを実験させている．BA-Python ではこれらのプログラムはいずれもブラウザ側で実行可能である．また，同教材では，データ数と性能のグラフが掲載されているが，グラフは表計算ソフトなどで別途作成させることを想定している．BA-Python では，**図 6**，**図 7** のように numpy や matplotlib を用いて，このグラフ自身を Python で描画するプログラムも作成可能である（この場合はサーバ側での実行が必要になる）．
  さらに，応用的課題として，WebAPI を利用したプログラム (**図 14**) や，組み込み機器である MicroBit を動作させるプログラムも掲載されている．WebAPI の呼び出しはサーバ側で実行したときに動作可能であるが，組み込み機器の制御は，BitArrow から当該機器への接続手法が確立していないため動作しない．
- 「モデル化とシミュレーション」では，乱数によるシミュレーションとして，**図 8**，**図 9** のようなさいころの出目の頻度を集計，**図 10**，**図 11** のようなモンテカルロ法による円周率の計算などを行い，グラフによっ

表 1　文科省教材 [6] の動作可否

項目	学習内容	動作可否
ア-1	コンピュータの仕組み	-
ア-2	計算誤差	○
イ-1	外部機器との接続	×
イ-2	基本的プログラム	◎
イ-3	応用的プログラム	○
イ-4	アルゴリズムの比較	◎
ウ-1	モデル化とシミュレーション	○
ウ-2	確定モデルと確率モデル	○
ウ-3	自然現象のモデル化とシミュレーション	○

-:プログラム掲載なし，◎:サーバ側とブラウザ側両方で実行可能，
○:サーバ側でのみ実行可能，× :実行不可能

て結果を確認させる演習を行っている．

本教材には掲載されていないが，BA-Python では，**図 12** のように教員や学習者が用意したデータをファイルから入力させて，**図 13** のような散布図を描画し，相関係数を求めたり，k-means 法で分類を行ったりすることも可能であり，学習者がテーマを決めて調査したデータを使って分類をさせる演習も行わせることができる．

教材に掲載されているプログラムと BA-Python での動作可否を**表 1** に示す．

6. 今後の予定

BA-Python は現在，ラッパーライブラリを整備している段階であり，先述した numpy や matplotlib などのすべての機能を利用できる状態にはなっていない．高校での学習を想定して，教材 [6] に書かれている範囲で使用されている機能を優先的に実装してきたが，今後教材から派生する発展的な課題も行えるようなサンプルプログラムを制作し，必要となる機能を追加していく．

また，現時点では，サーバ側での実行が必要なプログラムがが多く，一斉授業においてはサーバへの負荷の集中により授業が滞る懸念もあるため，ライブラリを JavaScript に移植して可能な限りブラウザ側で動作させるようにしていく．

さらに，使用できるライブラリを増やすことも検討している．例えば近年注目されている機械学習などのプログラムが作成可能な chainer[21] なども使用できるようにしていく予定である．

7. まとめ

本発表では，Web ブラウザ上で動作可能な学習環境 BitArrow における Python の処理系である BA-Python の機能を紹介した．

BitArrow の設計方針は，学習の対象となる言語で書かれたプログラムを JavaScript に変換することにより，Web ブラウザ上でプログラムを実行させるものであったが，

```
import numpy as np
import numpy.random as rd
import matplotlib.pyplot as plt

swc=0
def selectionsort(a):
    global swc
    for i in range(0,len(a),1):
        for j in range(i+1,len(a),1):
            if a[j]<a[i]:
                swc+=1#並べ替え回数
                temp = a[i]
                a[i] = a[j]
                a[j] = temp

#配列数を増やしながら
#並べ替え回数をグラフ化
for n in range(5,15):
    a = rd.randint(1, 100, n)
    print(" ソート前 ",a)
    selectionsort(a)
    print(" ソート後 ",a)
    print("並び替え ",swc)
    plt.scatter(n,swc,color="blue")

plt.show()
```

図 6　並べ替えアルゴリズムの性能グラフ化

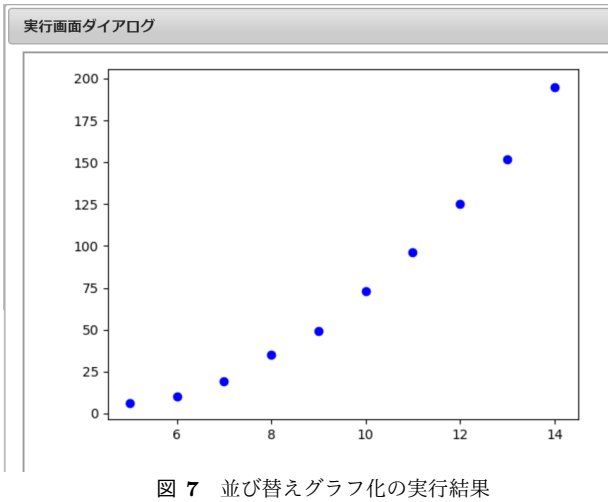

図 7　並び替えグラフ化の実行結果

BA-Python ではその方針を踏襲しつつもサーバ側での実行も可能なように機能拡張した．

これにより，従来の Python 処理系が用意している豊富なライブラリをそのまま利用することが可能になり，基本的なアルゴリズムの演習や，数値計算ライブラリを利用したモデル化とシミュレーションなどの演習を可能としている．

```
import numpy as np
import numpy.random as rd
import matplotlib.pyplot as plt

#サイコロを 100 回振る
saikoro = rd.randint(1, 6+1, 100)

print(saikoro) #乱数の値を表示
deme = [] #出目の数を数える配列
for i in range(6):
    deme.append(np.count_nonzero(saikoro==i+1))
print("出現数:",deme) #乱数の値を表示

left = [1, 2, 3, 4, 5, 6]
#グラフのタイトル，軸ラベル
plt.title("SAIKORO SIMULATION")
plt.xlabel("ME")
plt.ylabel("KAISUU")
#グラフをプロットして表示
plt.bar(left, deme, align="center")
plt.show()
```

図 8　さいころシミュレーション

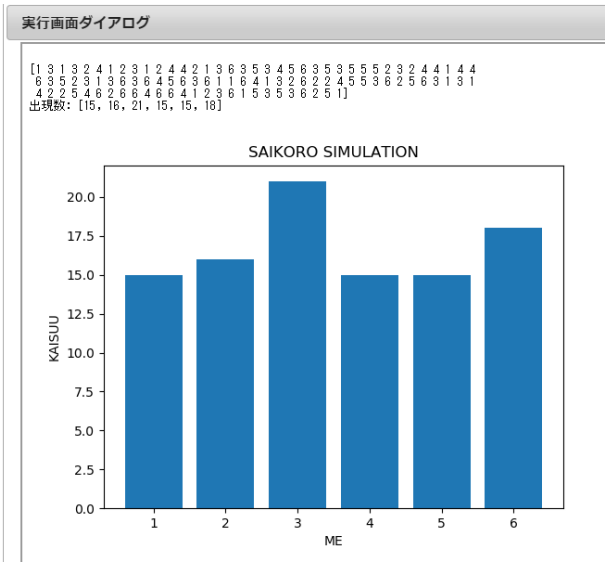

図 9　さいころシミュレーション実行結果

BA-Python は，文部科学省が高校向けに提示している教材で用いられている Python プログラムの大部分をそのまま動かすことができ，Python 処理系の導入が難しい学校の演習室などでも Web ブラウザだけで Python プログラミングの演習ができる環境となっていることを確認した．

謝辞　本研究は JSPS 科研費 19K03153 の助成を受けたものです。

```
import numpy.random as rd
import matplotlib.pyplot as plt

totalcount = 1000 #ランダムに打つ点の総数
incount = 0 #円に入った点の数
for i in range(totalcount):
    x = rd.rand() #0-1 の範囲の値
    y = rd.rand() #0-1 の範囲の値
    if x**2 + y**2 < 1.0:
        plt.scatter(x, y, c="red")
        incount+=1
    else:
        plt.scatter(x, y, c="blue")
#求まった円周率
print("円周率:", incount * 4.0 / totalcount)
plt.title("Monte Carlo method")
plt.show()
```

図 10　モンテカルロ法による円周率計算

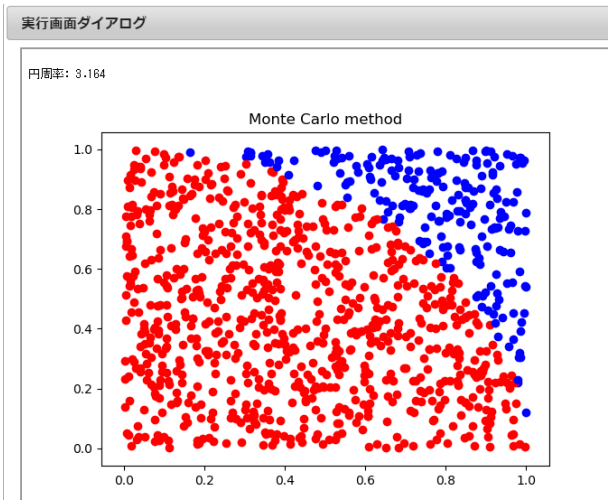

図 11　円周率計算結果

参考文献

[1] 長島和平，本多佑希，長　慎也，間辺広樹，兼宗　進，並木美太郎：オンラインで複数言語を扱うことができるプログラミング授業支援環境，情報教育シンポジウム 2016 論文集，Vol. 2016, pp. 1–9 (2016).

[2] SciPy developers: Scipy, https://www.scipy.org/. 2019 年 5 月 16 日アクセス.

[3] NumPy developers: Numpy, https://www.numpy.org/. 2019 年 5 月 16 日アクセス.

[4] 文部科学省：平成 29・30 年改訂学習指導要領，解説等，http://www.mext.go.jp/a_menu/shotou/new-cs/1384661.htm. 2019 年 5 月 16 日アクセス.

[5] 文部科学省：（情報編）高等学校学習指導要領（平成 30 年告示）解説，http://www.mext.go.jp/a_menu/shotou/new-cs/1407074.htm. 2019 年 5 月 16 日アクセス.

[6] 文部科学省：高等学校情報科「情報 I」教員研修用教材第 3 章，http://www.mext.go.jp/a_menu/shotou/zyouhou/detail/1416756.htm. 2019 年 5 月 20 日アクセス.

[7] 長　慎也，長島和平，堀越将之，兼宗　進，並木美太郎：

```
import matplotlib.pyplot as plt

f=open("user/scatter.csv","r")
xvalues=[]
yvalues=[]
for line in f:
    (x,y)=line.strip().split(",")
    xvalues.append(float(x))
    yvalues.append(float(y))
f.close

#グラフのタイトル，軸ラベル
plt.title("Scatter graph")
plt.xlabel("X")
plt.ylabel("Y")
#グラフをプロットして表示
plt.scatter(xvalues,yvalues)
plt.show()
```

図 12 ファイルからのデータを散布図で表示

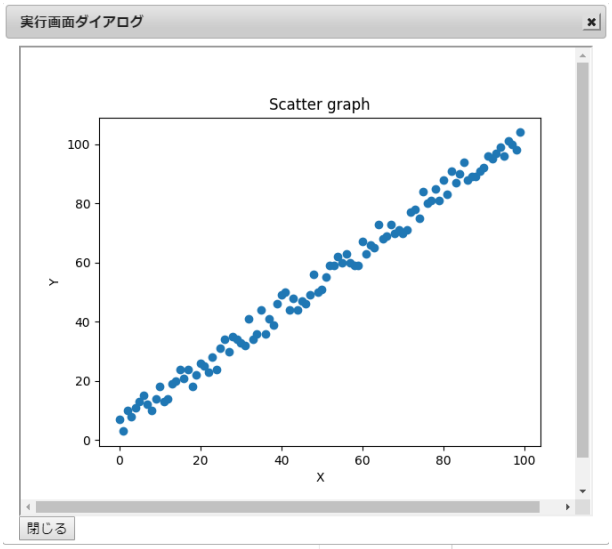

図 13 散布図実行結果

```
import requests
import json
# 使用する WebAPI の URL
url = "http://zipcloud.ibsnet.co.jp/api/search"
param = {"zipcode": "100-0013"} #WebAPI の引数
#WebAPI の戻り値が res へ
res = requests.get(url,params=param)
print (res.text)
response = json.loads(res.text)
address = response["results"][0]
print(address["address1"] +
address["address2"] +
address["address3"])
```

図 14 WebAPI による住所検索

オンラインプログラミング環境 Bit Arrow を用いた C 言語プログラミングの授業実践，情報処理学会情報教育シンポジウム論文集，Vol. 2017, No. 17, pp. 121–128 (2017).

[8] Python Software Foundation : Python, https://www.python.org/. 2019 年 5 月 21 日アクセス.

[9] Hunter, J., Dale, D., Firing, E. and Droettboom, M.: matplotlib, https://matplotlib.org/. 2019 年 5 月 16 日アクセス.

[10] Warren, J., Rixner, S., Greiner, J. and Wong, S.: Facilitating Human Interaction in an Online Programming Course, *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, SIGCSE '14, New York, NY, USA, ACM, pp. 665–670 (online), DOI: 10.1145/2538862.2538893 (2014).

[11] 大阪電気通信大学，東京農工大学，明星大学: Bit Arrow, https://bitarrow.eplang.jp/. 2019 年 5 月 21 日アクセス.

[12] 奥村晴彦: R を使った情報教育，情報処理学会情報教育シンポジウム 2010 論文集，Vol. 2010, No. 6, pp. 77–80 (2010).

[13] NW.js community: NW.js, https://nwjs.io/. 2019 年 7 月 04 日アクセス.

[14] Anderson, P. E., Nash, T. and McCauley, R.: Facilitating Programming Success in Data Science Courses Through Gamified Scaffolding and Learn2Mine, *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '15, New York, NY, USA, ACM, pp. 99–104 (online), DOI: 10.1145/2729094.2742597 (2015).

[15] Anderson, P., Turner, C., Dierksheide, J., Nash, T. and McCauley, R.: Learn2Mine, http://learn2mine.appspot.com/. 2019 年 5 月 16 日アクセス.

[16] Steven Hazel: codepad, http://codepad.org/. 2019 年 5 月 21 日アクセス.

[17] ギノ株式会社: paiza.io, https://paiza.io/. 2019 年 5 月 21 日アクセス.

[18] Sphere Research Labs: ideone, https://ideone.com/. 2019 年 5 月 21 日アクセス.

[19] The Rextester Team: rextester, https://rextester.com/. 2019 年 5 月 21 日アクセス.

[20] ギノ株式会社: PaizaCloud, https://paiza.cloud/ja/. 2019 年 5 月 21 日アクセス.

[21] Preferred Networks: Chainer, https://chainer.org/. 2019 年 5 月 21 日アクセス.