

2021 年度東京都私学財団 私立学校研究助成
研究報告書

情報 I を見据えたプログラミング教育

～学習者の独自設計を可能にさせたミニチュア配膳ロボットの プログラミング教材の開発と実践と評価～

2022 年 1 月

佼成学園中学校高等学校
岡野 英樹
萩原 知明

目次

1. はじめに.....	3
2. 本校の『情報』の授業.....	4
3. 今回の研究の目的.....	5
3.1. 学習指導要領.....	5
3.2. 学習指導要領解説（情報編）.....	6
3.3. 本校の生徒像.....	7
3.4. まとめ.....	8
4. 授業設計.....	9
4.1. 単元前の設計（情報社会、情報デザイン、デジタル、ネットワーク）.....	9
4.1.1. タイピング.....	9
4.1.2. 教科書内容と HTML&CSS.....	10
4.2. 単元の設計（プログラミング、問題解決の導入）.....	11
4.2.1. プログラミングの学習法.....	11
4.2.1.1. 学習教材.....	12
4.2.1.2. 学習順序.....	12
4.2.1.3. スパイラル学習.....	13
4.2.1.4. スキルの差の吸収.....	14
4.2.2. 本題：配膳ロボのプログラミング.....	14
4.2.2.1. コース.....	14
4.2.2.2. 配膳ロボ基本編（目標に準拠した評価）.....	15
4.2.2.3. 配膳ロボカスタマイズ編（学習者の独自設計、再構成要素）.....	16
4.2.2.4. プレゼンテーション発表（ルーブリック評価）.....	17
4.3. 単元後の設計.....	19
5. プリント集.....	20
6. 分析と考察.....	51
6.1. タイピング速度.....	51
6.2. 事前アンケート.....	51
6.3. Python の基礎課題と進捗状況.....	54
6.4. 配膳ロボ基本編.....	55
6.5. 配膳ロボカスタマイズ編.....	57
6.6. プレゼンテーション.....	58
6.7. 事後アンケート.....	58
7. まとめ.....	65
8. 謝辞.....	66
9. 参考文献.....	66
9.1. 引用文献.....	66
9.2. 参考文献.....	66

1. はじめに

令和 4 年度から新課程「情報 I」が必修で始まる。同一年次で履修を原則とし、単位数は 2 単位となる。また、令和 7 年（2025 年）から共通テストに「情報」が組み込まれることも昨年話題となった。教科「情報」は新しい科目であることや、他教科に比べ内容が流動的なこと、さらには教員数が少ないということなど、色々と変わっている教科である。さらに、平成 30 年告示の学習指導要領の改訂に伴い、文部科学省は、新たな情報の体制を作ろうとしている。しかし、現場レベルで考えると情報の授業を行う上で、まだまだ多くの問題が存在する。

例えば、一つ目に、生徒のタイピングが遅いなどのコンピュタリテラシーの問題である。課題の指示を出そうとするが、操作速度が遅く、課題量が少なくなってしまう、授業進度が遅くなってしまう。また、画面とキーボードを目が行き来するような状態では、健康にも悪く、あまり好ましい状態ではない。

二つ目に、生徒のスキルに差がある問題である。スキルの低い生徒は、課題が全部終わらず、残ってしまうのに対して、スキルの高い生徒は、課題を早く終わらせて時間を持て余してしまう問題である。

三つ目に、一人ひとりに対応していたら、時間がなくなってしまうという問題である。Team-Teachingで行っている学校もあるように、一人のトラブルに対応していたら授業が終わってしまう場合も多い。

四つ目に、情報の授業の時間だけでは、単元を全部消化できるのかという問題である。2 単位という時間の中であれだけの量を実習を交えながらこなそうとすると時間が足りなくなるという懸念がある。COVID-19 の影響で、オンラインやアーカイブ配信の補講をすることで多くの内容を教えることができたという報告をする学校があるが、それでも時間不足は情報教員を日々悩ませている。筆者の学校も上記の内容に苦労はしているが、少しずつアイデアで問題をつぶしていつているが完全ではない。

今回は、これらの問題を色々なアイデアや経験を通して少しは改善した授業研究を報告書にまとめていく。そして今回のメインとなるプログラミング教育では、東京都私学財団の授業研究助成を利用し、近未来に世の中に普及する可能性が高い配膳ロボットのミニチュア版（以下、配膳ロボ）を製作し、取り組みを実践しその内容を分析、考察していく。タイピングを学習し、Python の基礎知識を教えた後、チームで配膳ロボのプログラミングに挑み、一通り配膳ロボが機能したら、最後はチームでその配膳ロボをカスタマイズ（再構成）させ、自分達のオリジナルの配膳ロボを作りプレゼンテーションを行うというものである。

今回、配膳ロボに関しては初めての取り組みであり、うまくいかない部分も多かった。そして、多くの改善点も発見できた。つたない研究ではあるが、本校の取り組みの一部を参考にさせていただけたら幸いであるし、逆により取り組みがあった場合は、ご報告いただけると幸いである。

そして最後に、本研究に携わって頂いた本校情報科教諭の萩原知明先生、そして、配膳ロボを製作して頂いた株式会社スイッチエデュケーションの宗村様と木原様、そしてこれらの研究を支えてくれた妻と子供の家族に感謝申し上げたい。

佼成学園中学校高等学校 岡野英樹

2. 本校の『情報』の授業

本校は、三学期制の学校であり、情報は『情報の科学』を選択し、高校3年生で2単位履修している。また、情報教員は専任教員二人おり、Team-Teachingは行わず、一人で各々のクラスを担当している。高校3年生は1, 2学期までしかなく、時間が限られているため、複数の内容を組み合わせ、複数の内容を一つのプロジェクトにした状態にして、多くの内容をカバーしようと心掛けている。昨年度は、COVID-19の影響であまり内容の濃い授業はできなかったが、まず1学期にタイピングと教科書内容を学んだ。そして、2学期はmicro:bit(図1)でPythonを学習したあと、小型アクチュエータや小型センサーなどが入ったモジュールキットのBoson Starter Kit for micro:bit(図2)を教材として利用し、プログラミングを通して問題解決型の授業を行った。しかし、Boson Starter Kit for micro:bitを利用してもプロトタイプ的な作品しかできず、もう少し学習者に具体的な製品や商品をイメージさせる教材が必要だと考えていた。

そこで、今回は、配膳ロボを題材にしたプログラミング教材を開発していくことにした。また、今回の研究は高校3年生ではなく、1年B組の生徒で行う。研究対象となる1年B組は本校で今年から新設された「グローバルコース」のクラスであり、通常のコースとは少し違う教育課程を展開しており、情報も高校1年生で週2日の2単位扱うことにしている。教科横断型の授業やプロジェクト型、探究ベースの授業を行い始めており、本年度の1学期は国語×世界史や国語×生物で教科横断型の授業を行った。情報科も3学期に数学科とデータの活用で教科横断型の授業を行うつもりである。

また、PC教室の環境であるが、PC端末はWindows10を導入しており、授業支援ソフトのSKYMENUで管理・監視が可能な状態にしている。授業支援ソフトを使用すると、教員一人でも生徒の画面を自分の席にしながら把握・操作することができるので、大変便利なソフトウェアである。生徒から操作系で質問があると、確率的に8割ぐらいSKYMENUを通して解決できるため、教員の移動する時間や手間を削減することができる。また、教員は自分の席にしながら、生徒のソフトウェアの開いている状況も把握できるため、操作をしていない生徒がいれば、その生徒の名前を呼んであげることで、一体感や統一感のある授業を行うことができる。

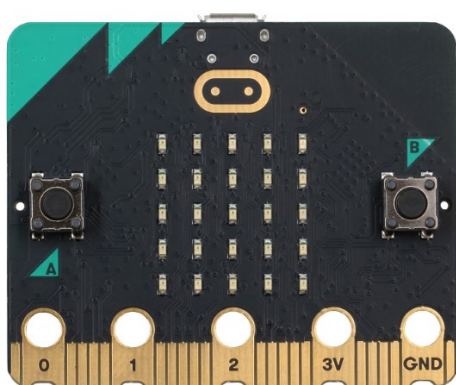


図1 micro:bit v2



図2 Boson Starter Kit for micro:bit

3. 今回の研究の目的

近年の教育法を踏襲しつつ、情報Ⅰに対応するものを作ることが目的だが、学習指導要領や本校の生徒像を抑えることも重要である。まずは、その点を整理していく。

3.1. 学習指導要領

平成 30 年に改定された学習指導要領では、情報において、「情報に関する科学的な見方・考え方を働かせ、情報技術を活用して問題の発見・解決を行う学習活動を通して、問題の発見・解決に向けて情報と情報技術を適切かつ効果的に活用し、情報社会に主体的に参画するための資質・能力」を養うことをその目標としている。

この目標の中で特に「問題の発見・解決」という要素が注目できる。節中にも多く登場する用語であり、本要領における「情報」教育の精神となる要素だと考えられる。つまりは「情報」教育は知識の習得に終始するものではなく、その知識を生かす実践を伴う必要があるということであろう。これと同時に第 1 款の(2)にあるように、我々が生きる上で出会う様々な事象について「情報」の知識からアプローチすることで「問題の発見・解決」に豊かな手段を与えるというのも重要である。よって普段の授業ではただ知識を身につけるための講義を行うだけでなく、実践を伴う必要がある。これは第 3 款 1-(1), 2-(3),(4)でも明確に言及されている。「情報」は配当されている単位数が少なく、実際に開講されるのも週 1~2 回程度にとどまるものと予想される。このなかで目標に照らした知識の獲得や問題の発見・解決活動を行うには、講義中心の考え方から脱却した、アクティブラーニングあるいはワークショップの要素を中心とした授業を設計していく必要性を感じる。今回研究する題材は「コンピュータとプログラミング」の分野となるが、レストランにおける配膳ロボを取り上げ、技術を用いて社会の問題を解決することについて実例を挙げながら実践させるような授業を実現していこうとするものである。

次に、内容について考察していく。ここでは、「情報Ⅰ」(第 10 節第 2 款第 1)について触れることとする。2 の内容に関しては(1) 情報社会の問題解決を根本とし、それを支えるものとして(2) コミュニケーションと情報デザイン、(3) コンピュータとプログラミング、(4) 情報通信ネットワークとデータの活用という 4 つのアプローチが横たわっている構造と考える。

今回の題材に照らせば、(1) においてはすでに言及している部分とも重なる。特にアの知識及び技能においては(ア)、イの思考力、判断力、表現力においては(ア)の要素に照らすことができる。ロボットというメディアを用いて、レストランの持つ人手不足や魅力向上といった課題を学習者が見出し、解決のためにロボットを構成するプログラムやモータ、LED などの技術を組み合わせて活用することでその実現につなげようとするものである。これを支えるアプローチとして一番重要なものが(3)である。配膳ロボとレストランを、ライントレースとモータによる移動をはじめとした拡張機能を持つマイクロコンピュータとこれに適したコースにモデル化し、問題の解決の方法を考える。このことは、ア-(ウ)、イ-(ウ)に沿っている。その際、マイクロコンピュータの扱う言語、処理能力、拡張機能の性能を把握したうえで、問題の解決を行うために必要なプログラミングコードや機能を選択して適切なアルゴリズムを組み立てていく必要がある。このうち前者はア-(ア)とイ-(ア)、後者はア-(ウ)とイ-(ウ)に沿っていこう。

これらを根幹として(2)や(4)では、(2)のコミュニケーションと情報デザインでは、マイクロコンピュータの拡張機能のカスタマイズにおける LED 表示や LED ライトを通したレストラン顧客への適切な

案内の提供やアミューズメントによる問題解決を試みることへの期待がある。さらに、この問題発見・解決の過程のプレゼンテーションを課すことにより、(2)だけではなく問題意識を支持するための適切なデータの活用として(4)の要素が求められている。

今回研究する題材の授業はこの指導要領に示されるすべての内容を網羅することは当然ながらできない。しかし、「問題の発見・解決」という精神を失うことなく「コンピュータとプログラミング」の目標に照らした知識の獲得をおこなうことを目指す、新たな情報教育の一つの具体化したモデルとなるだろう。

3.2. 学習指導要領解説（情報編）

続いて同学習指導要領の解説（情報編）に対する考察に移る。第1部3節-1によれば情報教育を「情報活用能力を育む教育」と位置づけ、その目標の観点を「情報活用の実践力」「情報の科学的な理解」「情報社会に参画する態度」の3つに整理している。

これらを今回の題材に照らして整理してみる。まず、配膳ロボをモデル化したマイクロコンピュータとその拡張機能における、プログラミング及びカスタマイズにおいては第1部3節2-(1)で示される「情報活用の実践力」の「課題や目的に応じて情報手段を適切に活用する」活動や、第1部3節2-(2)で示される「情報の科学的な理解」の「情報活用の基礎となる情報手段の特性の理解」、「情報を適切に扱ったり、自らの情報活用を評価・改善するための基礎的な理論や方法の理解」がこれにあたるといえる。生徒はマイクロコンピュータという一種の「メディア」を通じてレストランのスタッフや顧客に対する快適性や利便性を表現し、問題解決の一過程とする。この表現のためにはマイクロコンピュータとその拡張機能が持つ性能を把握し、この制約の中で可能な範囲の表現を行い、評価・改善していく必要がある。その際に必要となる知識や基礎理論の習得はマイクロコンピュータに対するものにとどまらず、顧客の安全性や利便性を考慮するための情報デザインの分野など、幅広く広がっていくことが想定できる。

また、カスタマイズしたマイクロコンピュータを架空の会社の商品としてレストランの経営者に紹介することを想定したプレゼンテーションでは、「情報社会に参画する態度」における「社会生活の中で情報や情報技術が果たしている役割」とその影響の理解を把握した上で臨む必要がある。加えて、「情報活用の実践力」における「受け手の状況などを踏まえて発信・伝達できる能力」を養い、経営者にとって納得できる、効果のあるプレゼンテーションとしていくことができるだろう。また、プレゼンテーション中に用いられるデータの説得性には「情報活用の実践力」における「必要な情報を主体的に収集・判断・表現」する力、「情報の科学的な理解」における「自らの情報活用を評価・改善」していくことが関わってくる。

1部3節-3で示される「資質・能力の三つの柱」においては、授業における活動において、プログラミングについて学習することにより「情報と情報技術を活用した問題発見・解決等の方法」を学んだ上で、マイクロコンピュータのカスタマイズとプレゼンテーションを通して「情報や情報技術を適切かつ効果的に活用し」、「様々な事象を情報とその結び付きの視点から捉え、複数の情報を結び付けて新たな意味を見いだす」ことによって、問題の発見・解決につなげていくことを実現していくことができるといえるだろう。

最後にこの研究におけるテーマであるプログラミング教育について考察しておく。1部3節-4には、平成28年12月の中央教育審議会答申では『時代を超えて普遍的に求められる「プログラミング的思考」などを育む』ことや『小・中・高等学校を見通した学びの過程の中で、「主体的・対話的で深い学び」の実現に資する』ことが示されている。このプログラミング教育は今回の指導要領改定で注目されるキーワードであるが、あくまで「情報の科学的な理解」を行うための「あらゆる学問」、そして「情報と情報技術を活用し

た問題の発見・解決等の方法」の中の一つに当たるということを根本に据え、技術トレンドを意識しながらも、これに左右されないこれらの技能資質、その表現力を養うカリキュラムづくりを心がけておきたいものである。今回の題材においても、マイクロコンピュータを現在の技術トレンドに沿った Python 言語にてプログラミングするが、ただ機能をプログラミングするのみにとどまらず、それをカスタマイズし、適切な拡張機能を追加することによる技能資質の習得、問題発見・解決のための技術の表現、新たな情報あるいはその技術への再構成が目指されており、答申の示すものにできる限り近づこうとする取り組みである。

3.3. 本校の生徒像

最後は本校の生徒像に関しても考察しておく。研究対象となるクラスは、今年度から新設された高校1年生のグローバルコース 36 名である。本校は中高一貫校であるが、高校からの入学者もあり、帰国生も若干名存在する。グローバルコースは混在コースとなっており、割合は以下の通りである(図3)。

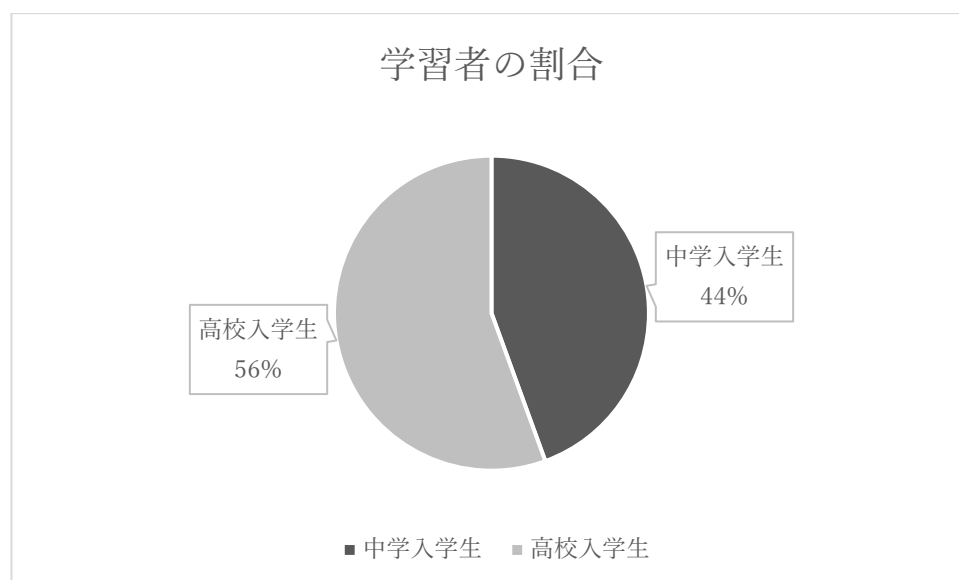


図3

このクラスは、他教科でも探究型の授業、プロジェクト型の授業を行っており、目標に準拠した評価にする授業を行うと、本当に意欲的に授業を受ける様子が見受けられる。また、グループ学習も得意であり、プロジェクト型の内容になると日々議論しながら取り組んでいる。ただ単に、「課題をやりなさい」や「作品を作りなさい」というようなものではなく、作品を作る際も、ルーブリック評価など、評価指標を提示すると、学習者も意欲的になってくる。

また、プログラミング経験に関してもアンケートをとった。アンケート内容は「あなたのプログラミング経験を以下の中から答えてください。」という問いに対し、「1,まったくない」「2,中学校の技術の時間程度」「3,プログラミングを学習中」「4,一通りプログラミング学習が終わって、色々プログラムを改変して遊んでいるレベル」「5,作品作っているレベル」から選んでもらった(図4)。

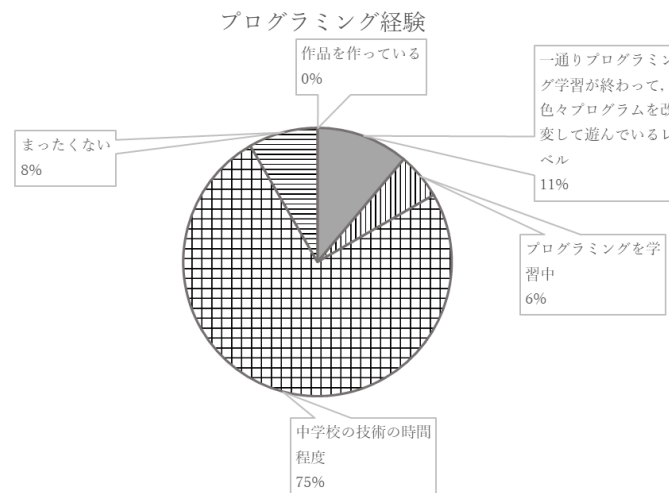


図 4

アンケートの結果，作品を作っている生徒は 0%，一通りプログラミング学習が終わって，色々プログラムを改変して遊んでいるレベルが 11%，プログラミングを学習中が 6%，中学校での技術の時間程度が 75%，まったくないが 8%であった。大半が中学校の技術の時間程度であった。これ以外にも，プログラミングに対する考え方のアンケートを行ったが，これは分析と考察の章の事前アンケートで紹介する。

3.4. まとめ

色々とまとめてみると，学習指導要領，生徒像など様々なことに留意しなければならないが，一番留意すべき部分は，時間ではないだろうか。時間が足りなくなってしまうような問題が見え隠れする。時間が限られているため，複数の単元を結合させたプロジェクト型授業を行っていくことは必要そうである。そして，学習者のやる気を出させるために，アクティブラーニング的要素，目標に準拠した評価で，再構成要素も入れて問題解決力を養っていく。

4. 授業設計

4.1. 単元前の設計（情報社会，情報デザイン，デジタル，ネットワーク）

まずは、スキルの醸成と知識の理解である。本校では、本年度は COVID-19 の影響で以下のようにした。1 学期は中間考査までは教科書内容とタイピング，期末考査までは教科書内容と HTML&CSS を学習した（表 1）。

表 1 2021 年度 1 学期の内容

	中間考査まで	期末考査まで
座学	教科書の内容	教科書の内容
実習	タイピング	HTML&CSS

4.1.1. タイピング

1 学期中間考査ぐらいまでは教科書の内容とタイピングである。これまでタイピングを学習せずに授業を行ったこともあったが、授業の進度が遅くなってしまう問題があったり、課題に時間を取られてしまうデメリットがあった。1 学期の最初にタイピングをする理由については、この後の様々な授業で課題量が多くても十分に生徒がついてくるスピードをつけるためである。

タイピングは、マイタイピング (<https://typing.twil.me/>) という Web サイトを使って学習している。アカウント登録すれば、自分でタイピングさせたい文章や単語の問題を作ることができ、アカウント登録なくとも、誰でも学習できるタイピングサイトである。以前は様々なタイピングソフトやタイピングサイトを使っていたが、キーの押す指が、自己流で正確な指ではなかったり、キーボードを見てしまう生徒もいるため、様々な教材研究の末にこちらにたどり着いた。

主に、タイピングは目標に準拠した評価をしている。まずは、ホームポジションキーを教え、キーボードの中段 (asdfghjkl;) 「中段マスター」 (<https://typing.twil.me/game/29641>) をまずは学習する。中段で秒間 2.0 タイプ以上（スコアに直すと 2000 点以上）を目標にしている。2000 点以上スコアを取ると、教員に知らせ、教員は手を隠す板を生徒の手の上の置き、そこで再度テストし、2000 点以上とると中段は合格というルールである。次に、中段と上段 (asdfghjkl; と qwertyuiop) で組み合わせたゲームを行い、同様に 2000 点以上取ると、再び教員に知らせ、テストをする。これも合格すると、中段と上段と下段 (asdfghjkl; と qwertyuiop と zxcvbnm,.) になり、これも合格すると、中段と上段と下段と最上段 (asdfghjkl; と qwertyuiop と zxcvbnm,. と 1234567890-) になる。

また、指導上の留意点として、「タイピングはイライラしてくるので、イライラしたら休憩してください。」と話すようにしている。休憩後に、新たな気持ちでタイピングすることで内田クレペリン検査のように休憩後パフォーマンスがあがったりすることを指導しながら体感している。またここ最近に、新しいスキルの習得には、頻繁な休憩がスキルを上達させるという論文が発表された。新しいスキルの習得は練習中に上がっているのではなく、休憩中にあがっているという論文[3]である。来年度はこれを参考にしながら効果的な休憩方法についてもさらに授業研究していく予定である。

このようなルールにすることで、生徒の能力に応じて個々で学習できるという点と、全員がこれを同時に同じ空間で行うことによって、お互いに競争心を持ち、意欲的に学習させることができる。他の生徒が合格していくと、自分もやらざるを得ない状況になってくる。このような状態であると、「中段」合格者は 97.2%, 「中段と上段」合格者 83.3%, 「中段と上段と下段」合格者 61.1%, 「中段と上段と下段と最上段」合格者は 27.8%であった。また、タイピングを学習し終わってから、他教科でもメモを取るときなどにタイピングをする生徒が増えたと担任から報告があった。タイピングの授業後でもタイピングを自ら行っている様子が伺えた。



図5 タイピングサイト
(マイタイピング)

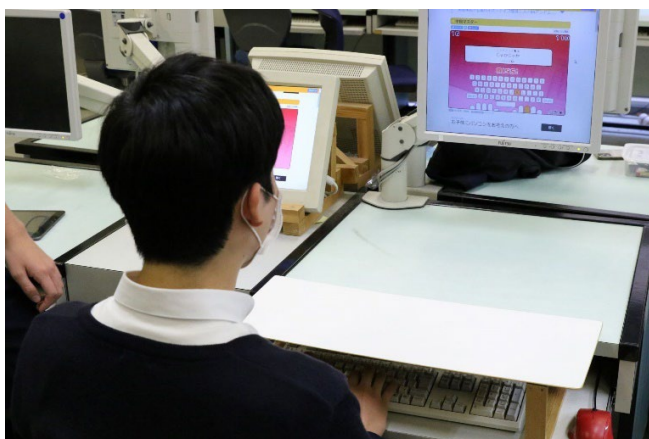


図6 タイピングのテストの様子
(キーボードを木の板で隠してテスト)

4.1.2. 教科書内容と HTML & CSS

1 学期期末考査までは教科書内容と HTML & CSS である。中間考査までに培ったタイピングの能力を落とさないために、キーボードを使った HTML と CSS の実習と教科書の内容を行っている。HTML & CSS は Sublime Text 3 を使って、タグや css のプロパティの入力補完が可能なエディタを利用して学習している。題材は、教科書の学習内容を HTML & CSS に書き起こすような構成にしている。例えば、No1 はデジタル化の色を扱い、rgb の値によって Web ページの背景が変わるような HTML 文書を学習する。No2 は産業財産権を扱い、テーブルタグを学習する。No3 は著作権を扱い、リストタグを学習する。No4 は、デジタル化 (2 進法) を扱い、img タグを学習する。No5 は、デジタル化 (16 進法) を学習し、16 進数によるカラー表現を学習する。No6 は、デジタル化 (ビットとバイト) を扱い、ここでは、今までの課題ができなかったものはその課題を、できている者はこちらから、デバッグの学習をするため、わざと間違えた HTML ファイルを配り、それを修正する授業をしてバランスをとっている。No7 に関してはインターネットの仕組みを扱い、CSS の初歩を学習する。No8 については、プロトコルを扱いし、トップページのレイアウトづくりの CSS を学習する。No9 については、セキュリティを学習し、前回作成したトップページのメニュー部分に今までの HTML 文書をリンクさせ、ハイパーリンクを学習する。また、No1 ~ No9 までのプリントを通して、情報モラルのことを科学的に説明しながら授業を行っている。

4.2. 単元的设计 (プログラミング, 問題解決の導入)

ここからは2学期の内容である。単元としては、プログラミングと問題解決を扱う。予定では、micro:bitでPythonの基礎学習をし、その後、チームで配膳ロボの基本の動作をさせるためのプログラミングを学習し、さらに配膳ロボのカスタマイズを自分たちでプログラミングする問題解決を行おうとしていた。しかし、9月初めは、7月12日から始まった4回目の緊急事態宣言のために、学校全体がオンライン授業(Zoom)に切り替わってしまい、PC教室を使った授業ができなかった。そのため、アルゴリズム2 (2コマ) (図7)と、ドリトル (1コマ) (図8)を使ってプログラミングの考え方に慣れさせる授業をオンラインで行った。

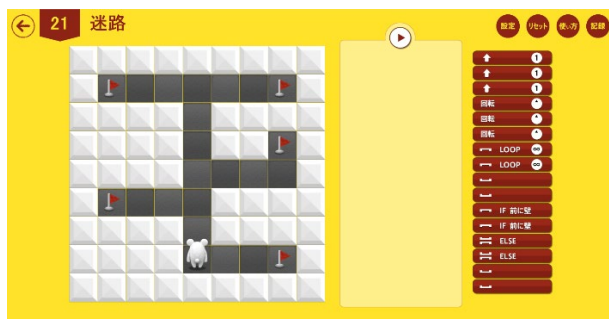


図7 アルゴリズム2

<https://algo.jeita.or.jp/prm/2/index.html>



図8 ドリトル

<https://dolittle.eplang.jp/>

アルゴリズム2は1時間目に使い方の説明をし、どこまでできるかどうかを競わせた。2時間目には迷路抜けの代表的アルゴリズムの「左手法」を教え、アルゴリズムの考え方に迫っていった。また、ドリトルは様々な図形をこちらで考え用意し、図形描画のプログラミングを行った。Zoomのブレイクアウトルームでグループ分けをし、グループで協力しながらお題の図形の作成を分担させプログラミングして与えられた図形を作らせた。そして、その図形のスクリーンショットをとって、ロイロノートで提出というアクティブラーニング型の授業を行った。ドリトルはオブジェクト指向の言語となっており、プログラミング学習でオブジェクト指向の説明が必要になったときに、ドリトルの内容に再び触れる予定である。

4.2.1. プログラミングの学習法

プログラミング初心者にどのようにプログラムを教えていけばよいかを考えるのはやはり頭を悩ませる。様々なプログラミング学習法の記事や文章を読んでいると、大抵、「自分の好きなゲームを作る」や「ゲームプログラムを改変して、自分なりに遊ぶ」などであった。確かに、筆者も自分の作りたいものがあって、プログラミングを勉強していったが、これを学校教育で行うには限界があると感じていた。理由は、作りたいゲームがすべての学習者に対して一緒ではなかったり、つまづきが起きた際に一人で学習していくとそのつまづきをなかなか解決できなかったりするからである。また、学ぶ言語に関しても多少悩まされる。筆者個人の意見として学ぶ言語はなんでもよいと考えている。ただ、大学入学共通テストに登場するプログラミング言語はDNCLと呼ばれ、大変Pythonに似ている形で書かれているので、学校教育ではPythonを選択するという方法が考えられる。なので、Python (正確に言えばmicro:bitはMicroPython) を使って、自分のプログラミング経験や指導の経験を活かし、以下の四つの要素を心掛けた教材作りをした。

4.2.1.1. 学習教材

学習教材はハードウェアを通して学習できる micro:bit v2 を使うことにした。v2 は初期型の micro:bit の改良版で、マイクがついたり、音を micro:bit 本体から出せたり、タッチ検出機能がついて使いやすくなった。また、エディタは、「Python Editor for micro:bit」(<https://python.microbit.org/v/2>) (図 9)を利用した。プログラミング学習というと、CUI ベースで黒い画面にプログラミングコードを入力しながら行うことが多い。しかし、CUI ベースであると最初は興味をもつが、だんだんと興味がなくなってくるのを筆者の大学時代の周りの状況などから感じていた。確かに CUI によるプログラミングの方が、エラーメッセージがすぐわかり、プログラミングに慣れた人にとっては最適な環境ではあるが、初学者ではあまり変化がなく興味が出にくい。一方、ハードウェアを通したプログラミングは、エラーメッセージが見にくく、ハードウェアへの転送時間があるため時間がかかるが、動いた時の感動は生徒たちの心を惹きつけるらしく、動く感動の言葉が教室のあちらこちらから聞こえてくることを体感している。

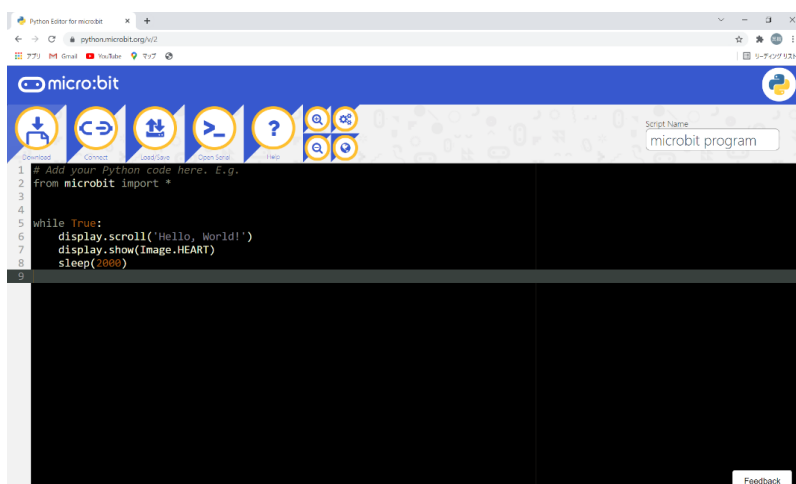


図 9 Python Editor for micro:bit

4.2.1.2. 学習順序

学習順序に関しては特にこだわった。まずは、多くのプログラミング教材を扱う場合、「とりあえず実行」というような内容を初回に行うことは一般的だと考えるが、その後の内容として、エラー対応に強くなるために、トラブル対処方法やコメントアウトなど状況把握や操作方法に時間を割くことにした。具体的には 1 回目はとりあえず実行というような”Hello,world”的なプログラムの実行, 2 回目はエラー内容の見方, 3 回目はコメントアウトとキー操作, 4 回目は変数, 5 回目は疑似乱数と条件分岐①, 6 回目は条件分岐②, 7 回目は初期化と反復処理, 8 回目は, micro:bit のセンサー, 9 回目と 10 回目はアクチュエータなどの実行方法である。多くのプログラミングの学習の専門書を見ると, Hello,world のあと, 変数を扱った四則演算を学習していることが多いが, 今回, 1 回目に Hello,world に似た内容を取り扱った後, 2 回目にエラーの理解の仕方に 1 コマを費やした。意図的にエラーを出すプログラムを配り, エラーメッセージを読み, どこが悪いかを自分で探し, 修正するという内容である。これを 2 回目に持ってきた理由として, 学習者の心理的なエラーへの嫌悪感をいち早く取り除きたいという狙いがあった。3 回目はコメントアウトとキー操作である。プログラミング学習初心者, は, 間違えたプログラムを書いた際, 直すときに今まで書いたコードを消そうと

する。しかし、これでは消したコードが再度必要になった時に、再び記述しなければならないので効率が悪い。コメントアウトを使えば、一時的にその部分を無効化できる。ここで学習することで、今後エラーが出たときに自分で部分的にプログラムを走らせたり、走らせなかったりして検証させるテクニックを身につけさせる狙いがあった。以前、筆者もプログラムは書けるほうではなかった。しかし、プログラミングを得意とする友人の操作画面を横で見えていたら、変数に値が導入されたらそのあとにすぐ変数の値を表示させるような `print` 文などを書いて、変数の中身がどういう型でどのように入っているか確認していた。その変数の値がわかると、コメントアウトをして、プログラムの実行上は消すようにし、次のコードを書きにいていた。しかし、何かエラーが起これと、さらに、そのコメントアウトを再びアンコメントし変数の値を確認したりしていた。この例はプリント No3 や No5 などに登場するので参考にいただけると幸いである。

以上のような、学習順序にこだわり、教材にした。

4.2.1.3. スパイラル学習

学習内容は、前回の内容を必ず復習してから、新しい内容に入っていくような構成を心掛けた。プログラミング学習など新しく何かを学ぶ場合、反復練習を行い少しずつ内容を難しくしていくということは学習者の理解を向上させやすい。また、このようにすることで前回の授業に欠席した者や、理解できなかった者に対して、復習要素を入れることで、すぐキャッチアップさせ新しい内容を理解しやすくする狙いもある。以下の表 2 に扱う内容を灰色で示している。一度で終わるのではなく、次の授業や 2～3 回後の授業で再び登場させ、繰り返し学習を心掛けている。

表 2 扱う学習内容

	No1	No2	No3	No4	No5	No6	No7	No8	No9	No10
実行										
エラー内容										
コメントアウト										
変数										
疑似乱数										
IF 文										
初期化とループ										
センサーの利用										
アクチュエータの利用										
関数										

また、ここでは配列は軽くは扱ったが、厚く扱ってはいない。配列は理解しやすい面白い題材が見つからなかったという理由もあるが、添字で理解に時間がどうしてもかかってしまうからである。配列は、実際に配列は配膳ロボを動かすときに登場させる予定である。

4.2.1.4. スキルの差の吸収

学習者のレベルの差はどうしてもつきものになる。このような場合であっても、全員が意欲的に参加できる授業設計が望まれる。これを実現するために、課題を3段階にレベル別に分けて提示する構成にした。星1つは全員がやるべき課題、星2つは挑戦してほしい課題、星3つは課題ができすぎて時間を持て余してしまう人用の課題、いわゆる難問である。星1つは写経するレベルの課題で、星2つは自分で考えながらプログラムを少し変えて行っていく課題、星3つは、より深く考えたり、Webで検索したりしながら行っていく課題である。以前、数学でこのような難易度を分けて授業プリントを作って教えられている先生と出会った際に、レベルの差は難易度別に作っていけばレベルの差を解決できると教わり、このように設計している。これら4つのことを考慮したPythonの基礎学習プリントを作り、10コマ利用し学習した。

4.2.2. 本題：配膳ロボのプログラミング

さて、Pythonの基礎学習が終わった後、ここから本題の配膳ロボのプログラミングに入っていく。前述したように、去年はプロトタイプ（試作品）的なものしかできなかったのですが、今年はより具体的な製品をイメージさせる教材開発に取り組んだ。題材は掃除ロボなど様々なアイデアを考えたが、近未来に普及しつつある配膳ロボを教材にしようと考えた。配膳ロボは「焼肉きんぐ」や「幸楽苑」などの飲食店に導入されており、注文した料理をロボットが運んでもらえるサービスである。

実際に配膳ロボの仕様を決定する際に、完璧な配膳ロボをあえて作らず、学習者の独自設計（再構成要素）いわゆるカスタマイズを入れられるような状態にし、学習者が配膳ロボの問題を自ら発見し、解決できるような学習要素を入れ込んだ設計にした。左側が基本動作のみを学習する配膳ロボの写真で、右側が基本編のプログラムをカスタマイズさせて学習する配膳ロボの写真である。これを二人一組になりグループで取り組んだ。

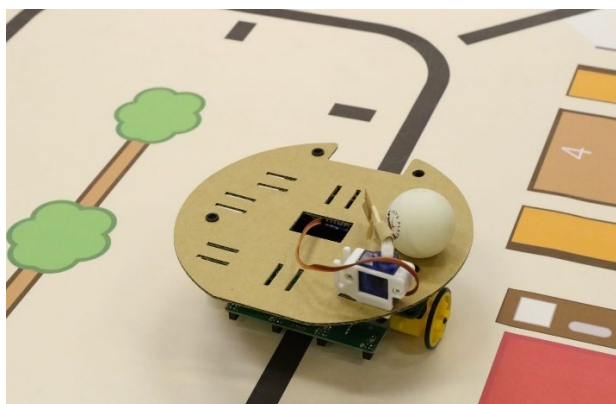


図 10 配膳ロボ写真

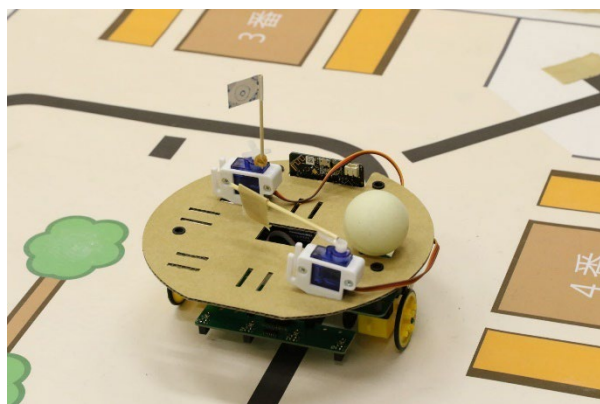


図 11 独自設計（再構成要素）

4.2.2.1. コース

プログラミングをし、試走をする際に、生徒が重なり合って試走に時間がかかってしまうことを避けたかった。そこで、コースは環状線のような状態にし、8人で一つの店舗を同時に使えるようなコースを制作し、それを5つ分作った。このようにすることで、テスト走行時の学習者の同時参加人数を増やせるようにした。また、教員に見せるための本番コースは、テーブルや椅子やレジやドリンクコーナーなど紙粘土や木工で

色々と装飾し、よりリアルにこだわり、学習者のやる気を出させるようにした。コースの印刷は学校の大判プリンタを利用した。

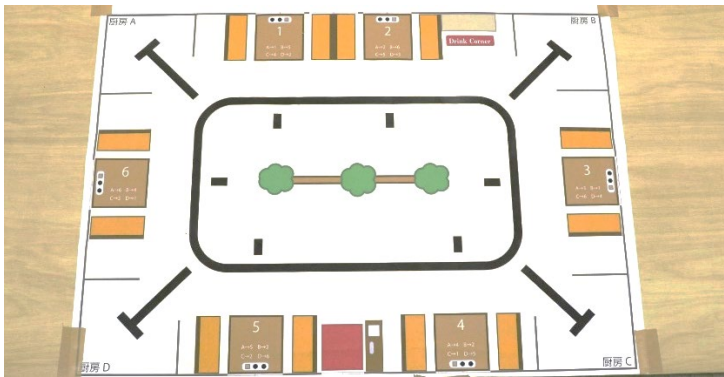


図 12 テスト用コースの写真

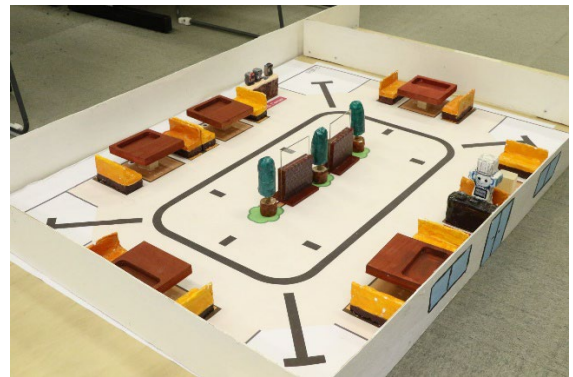


図 13 本番用のコース

4.2.2.2. 配膳ロボ基本編（目標に準拠した評価）

プログラムの多くは、様々なイベントが複数絡み合い成り立っている。また、開発する際も、完成形のコードを上から順番に書いていくのではなく、イベントを小分けにして少しずつ膨らましてコーディングしていくのがほとんどである。そのため、配膳ロボットの基本の動作のプログラミングも小さく分割して、様々なイベントを小目標としてクリアしていくことによってプログラミング的思考力を身につける構成にしようと考えた。以下がその目標内容である。こちらを5コマの時間を割いて行った。

- STEP1. コードはコメントアウトを使い行為を分解しておく。
- STEP2. モジュールを読み込み、まっすぐ走る。
- STEP3. 環状線をライントレースする。
- STEP4. テーブル番号のセットをする。
- STEP5. テーブルセットのプログラムをコメントアウトし、配膳テーブル番号を4（番目）にしておく。
- STEP6. テーブルをカウントするプログラムを書く。
- STEP7. 指定したテーブルで停まり、配膳する。
- STEP8. キッチンをカウントするプログラムと定数の宣言。
- STEP9. 環状線からスタートし、自分のキッチンに戻る。
- STEP10. 自分のキッチンからスタートし、T字路で左折し、4番目のテーブルで配膳し、自分のキッチンに戻る。
- STEP11. テーブル番号のプログラムをアンコメントし、T字路にもどって半回転させるプログラムを書いて完成。

発展編

- STEP12. 厨房 B,C,D からスタートしても、厨房 A の番号のところに届ける。
- STEP13. この配膳ロボの問題点を発見して改善案を報告。

学習者間のスキル差を吸収するために、発展編を作り、難問を用意した。

4.2.2.3. 配膳ロボカスタマイズ編（学習者の独自設計、再構成要素）

一通り走らせて機能させるところまでを学習したあとは、4 コマ使い配膳ロボットのカスタマイズに挑んだ。この時点で学習者の進捗がバラバラになったので、基本編の完成プログラムを全員に一回渡して、それを使ってカスタマイズ編に臨んだ。配膳ロボットのベンチャー企業の社長と副社長になったという想定のもと、自分たちで開発し、自分たちで売り込むという内容にした。自分の想像力でさらなる配膳ロボのカスタマイズに挑んで、プログラミングし、再構成させる。再構成には、モータに指定させた角度で止まることができるという 180 度サーボモータ、低速モータ（360 度サーボモータ）、LCD（液晶ディスプレイ）、LED の NeoPixel を利用する。さらに、竹ひご、竹串、画用紙、水性ペンなどの文房具でそれらを装飾して配膳ロボをデザイン的にもカスタマイズさせる。

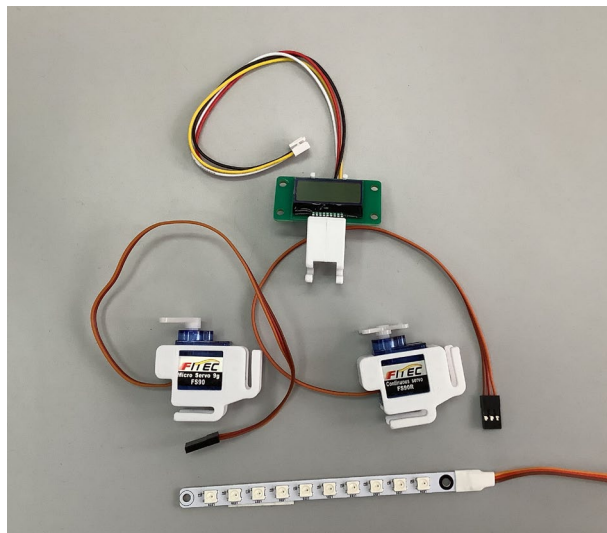


図 14 カスタマイズ部品
（上、左、右、下の順に
LCD、180 度サーボモータ、360 度サーボモータ、Neopixel）

Python の基礎課題の星 1 つの課題や配膳ロボの基礎編の課題に関しては、画一的な解答が決められているので、そう難しくはないが、ここから自分で問題を発見し、解決までを行わなければならない。ただ、自分で作りだしていくのは難しいので、各部品のサンプルコードは提示するようにしたが、より資料を読み込んでもらえるようにあえて雑に作った。

また、店舗は、以下の 7 つのシチュエーションが存在するという仮定の下、カスタマイズについて考えてもらった。

7つのシチュエーション

1. 家族連れで、子供がよく来るお店
2. 照明を落とした薄暗い感じのお店
3. 色々なメニューがありすぎて、どのメニューがオススメかわからないお店
4. その日のおすすめが多くあり、時価で出すお店
5. ビジネスマンが一人で来るお店
6. 女性が来づらいお店
7. COVID-19 で売り上げが落ち込んでいるお店

4.2.2.4. プレゼンテーション発表（ループリック評価）

配膳ロボの発表については、2人1組のグループでスライドを作り、グループの中で一人が発表、一人がiPadを使って動画を取り、それをロイロノートに提出させた。これを行うことで発表の時間を削ることができる。ロイロノートをあとで共有しておけば、プレゼンテーション動画を全員と共有することもできる。スライドの内容は、①トップページ、②会社紹介、③社会的背景とデータ、④商品紹介、⑤実演、⑥プログラム、⑦参考文献の要素を含むよう指示した。また、それに伴うループリック評価とその配点は以下の通りである。ループリック評価について、評価指標の重みがすべて一定であると、そう重要でないものと重要であるものの差がつかなくなってしまうため、重要なものの配点を上げ、そう重要でないものの配点は下げる工夫をした。また、プレゼンテーションの説得には、アリストテレスの言葉を拝借して、「説得力＝ロゴス×パトス×エトス」をプレゼンテーションの評価要素に盛り込んだ。以下が使ったループリック評価である（表3）。

表 3

	8	6	4	2
プレゼン声 (パトス, エトス)	聞こえやすい声で、抑揚をつけながら話している	聞こえやすい声で話している。	声が小さく、聞き取りにくい。	声が小さく、聞こえない。
説明のわかりやすさ (ロゴス)	説明が最初から最後まで首尾一貫している。	説明が最初から最後まで首尾一貫しているが、一部論理性にかけるものがある。	説明が曖昧でよくわからない部分が残る。	説明が支離滅裂でよくわからない。
データ、グラフなど (ロゴス)	自分で持ってきたグラフをわかりやすく編集し、効果的に使っている。	与えられたグラフを効果的に使っている。	自分で持ってきたグラフや与えられたグラフを使っているが効果的ではない。	グラフを使っていない。
問題発見・意識 (ロゴス)	様々な社会課題を発見し、問題意識を明確にしている。	様々な社会課題を発見しているが、問題意識が少し曖昧である。	社会課題を発見できていないが、問題意識はある。	社会課題を発見できておらず、問題意識もない。
デモ動画 (エトス)	デモ動画があり、適切に動いている。	デモ動画があるが適切に動いていない。	デモ動画があるが動いていない。	デモ動画がない。
解決方法・プログラム (エトス)	問題解決のための動作が適切に動いている。	問題解決のための動作が適切ではないが動く。	問題解決のための動作が適切ではあるが、動かない。	問題解決のための動作が適切ではなく、なおかつ動かない。
プログラムの貼り付け	当該部分が全て載せられており、わかりやすい。	当該部分が一部乗せられており、概要はわかる。	プログラムが載せられているが、エラーのあるプログラムである。	プログラムが載せられていない。
カスタマイズ 割り込みプログラム	割り込みプログラムを書いて文房具を使って装飾している	割り込みプログラムを書いているが、文房具で装飾していない。	割り込みプログラムを書いているが、文房具で装飾している。	割り込みプログラムを書いておらず、文房具で装飾していない。
解決策 (ロゴス)	問題意識からの解決策に論理性が十分に見られる。	問題意識からの解決策に論理性がやや見られる。	問題意識からの解決策が支離滅裂	問題を解決していない。
	4	3	2	1
熱意 (パトス)	カメラを見て情熱的に他者に訴えかけるプレゼンをしている。	時折情熱的なプレゼンテーションが見られる。	標準的な声でプレゼンテーションをしている。	カメラに目を合わせず、小さな声でプレゼンテーションをしている。
参考文献、引用文献 (ロゴス)	書籍、URL などメディアに応じて適切な引用の仕方をしている。	一部参考文献の書き方が適切でない。	URL が張られたままだったり、タイトルだけのものがある。	参考文献、引用文献がない。

また、3 番目のデータ、グラフに関しては、飲食店業界の人手不足を説明する棒グラフをサンプルで与え、グラフ表現は必須だが、このグラフを使っても使わなくてもどちらでもよいとの説明をし、自分でグラフを探して発表すると評価があがる旨を説明した。

4.3. 単元後の設計

3 学期は、データベースや Excel でモデル化とシミュレーションを行い、数学科と教科横断で、データの活用をする予定である。

5. プリント集

micro:bit No1 プログラムの記入と実行

組 番 氏名 _____

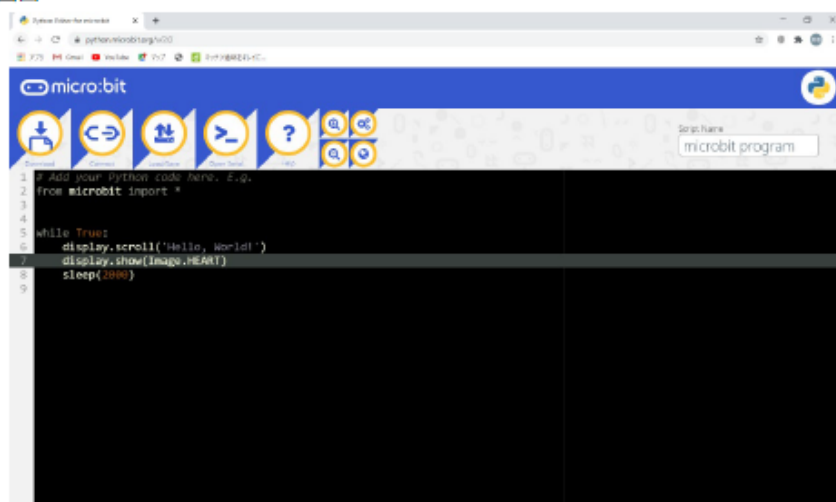
1. editor を検索

microbit python editor

検索

<https://python.microbit.org/v/2.0>

2. 編集画面



3. 初期のプログラム解説

	コード	内容
1.	# Add your Python code here. E.g.	①コメントアウト
2.	from microbit import *	②ライブラリを読み込
3.		
4.	while True:	③インデントがさがった部分をずっと行う :
5.	display.scroll('Hello, World!')	④ディスプレイに「Hello, World!」とスクロールする
6.	display.show(Image.HEART)	⑤ディスプレイに「HEART」というイメージを出す
7.	sleep(2000)	⑥2000 ミリ秒の間、動きを止める

- ①コメントアウト 実際のプログラムの動作に影響しない文字列。
- ②マイクロビットのプログラムのライブラリ（プログラムコード集）を呼び出して使えるようにする。
- ③while で始まる文は「～の間：以降の事をする」という意味。この場合はずっと：以降の動作を繰り返すということになる。
- ④ディスプレイにかっこの中の文字列をスクロールさせる（日本語は使用できない）。
- ⑤ディスプレイにイメージを出す。この場合は、micro:bit のライブラリに収録されている「HEART」というイメージを出している。
- ⑥sleep()はかっこの中にある時間、プログラムの動作を止める。かっこ内の単位はミリ秒。1秒は1000ミリ秒。

課題 1-1★ ディスプレイにスクロールさせる文字列を変えてみよう。（プログラムの画面の写真提出）

課題 1-2★ イメージを 10 秒間表示させてみよう（プログラムの画面の写真提出）

課題 1-3★★ HEART 以外にも以下のイメージがライブラリに収録されている。気になるものを表示させてみよう。（microbit の LED 画面の写真提出）

Image.HEART	Image.HEART_SMALL	Image.HAPPY	Image.SMILE	Image.SAD
Image.CONFUSED	Image.ANGRY	Image.ASLEEP	Image.SURPRISED	Image.SILLY
Image.FABULOUS	Image.MEH	Image.YES	Image.NO	Image.CLOCK*
Image.ARROW_**	Image.TRIANGLE	Image.TRIANGLE_LEFT	Image.CHESSBOARD	Image.DIAMOND
Image.DIAMOND_SMALL	Image.SQUARE	Image.SQUARE_SMALL	Image.RABBIT	Image.COW
Image.MUSIC_CROTCHET	Image.MUSIC_QUAVER	Image.MUSIC_QUAVERS	Image.PITCHFORK	Image.XMAS
Image.PACMAN	Image.TARGET	Image.TSHIRT	Image.ROLLERSKATE	Image.DUCK
Image.HOUSE	Image.TORTOISE	Image.BUTTERFLY	Image.STICKFIGURE	Image.GHOST
Image.SWORD	Image.GIRAFFE	Image.SKULL	Image.UMBRELLA	Image.SNAKE

*1 から 12 までの数字を指定できる。

** N, NE, E, SE, S, SW, W, NW が指定できる。

課題 1-4★★★★ これまでの知識を使い、ある文字をスクロールさせたあと、3 回連続同じイメージが点滅する方法を考えてみよう。（プログラムの画面の写真提出）

micro:bit No2 エラーの見方と直し方（間違い探し）

組 番 氏名

1. プログラムの記述を開始する場所は決まっている。また、正確に記述しなければならない。

		解説
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.		
4.		
5.	while True:	
6.	display.scroll('Hello, World!')	インデントが一つ（タブが一つ）
7.	display.show(Image.HEART)	インデントが一つ（タブが一つ）
8.	sleep(2000)	インデントが一つ（タブが一つ）

一文字でも間違えたり、記入開始場所を間違えるとエラーになる。

2. 間違い探し

課題 2-1★ 配布されたファイルをテキストエディタに張り付けて、どこが違うかを探し、正しい答えを書いてみよう。（このプリントの写真提出）

AttributeError（属性エラー）…ドット（.）の後の単語が間違っているとこうなる可能性大

行目の が間違っていて が正しい

SyntaxError（構文エラー）…特殊記号が間違っているとこうなる可能性大

行目の が間違っていて が正しい

NameError（名前エラー）…単語やドット（.）の前の単語が間違っているとこうなる可能性大

行目の が間違っていて が正しい

IndentationError (インデントエラー) …タブやスペースが間違っているとこうなる可能性大

_____行目の_____が間違っていて_____が正しい

※tab一つ分は⇐と書いてもらえれば結構です。

また、タブは半角スペース4つ分を入力しても同じ解釈になります。

ImportError (インポートエラー) …ライブラリやモジュールが間違っているとこうなる可能性大

_____行目の_____が間違っていて_____が正しい

そのほかにも色々エラーがあります。この簡単なプログラムでさえ、様々なエラーがあることを知っておこう。

コラム

プログラムで一番厄介なのは、このエラーです。何度もエラーを出すとへこたれます。実は私も教材研究の段階でエラーを出したりして、いくつか授業案を作っていると、ダウンロードしたファイルの数がすぐ100を超えてしまいました。エラーはつきものだという考えのもと、勉強してもらえれば幸いです。ただ、やはりエラーは嫌なので、エラーが出たときにすぐ対処できるような想像力や知恵を身に付けていきましょう。

論理エラー

論理エラーとは、プログラムの実行結果が意図した通りにならないエラーのことである。論理エラーは、転送が正常に行われ、プログラムの実行途中で異常終了することなく動作するものの、アルゴリズムが論理的に正しくない場合のことを意味する。例えば以下のコードは論理エラーである。

```
1 # Add your Python code here. E.g.
2 from microbit import *
3
4
5 while True:
6     display.scroll('Hello, World!')
7     display.show(Image.HEART)
8     sleep(2000)
9
```

これを書くと、ずっと Hello,World が流れ続け、ハートが表示されません。

裏へ

3. 間違い探し問題作り

課題 2-2★★ 最初のプログラムの一部を自分で削除してみたり、何かを追加してみたり、あるいは大文字にしてみたり、小文字にしてみたりして、どのようなエラーが起こるかどうを確認してみよう。5 種類のエラーを確認できたらポイント UP！（このプリントの写真提出）

_____行目の_____を_____に変えると、_____エラーになる。

_____行目の_____を_____に変えると、_____エラーになる。

_____行目の_____を_____に変えると、_____エラーになる。

_____行目の_____を_____に変えると、_____エラーになる。

_____行目の_____を_____に変えると、_____エラーになる。

_____行目の_____を_____に変えると、_____エラーになる。

_____行目の_____を_____に変えると、_____エラーになる。

_____行目の_____を_____に変えると、_____エラーになる。

4. エラー名把握

課題 2-3★★★ 上に出した以外にもエラーがある。どのようなエラーがあるか調べてみよう。「python エラー 一覧」などと調べてみましょう。内容がわからなくても、エラー名はわかったりするので、エラー名とその説明を書いてください。

エラー名	説明
Error	
Error	
Error	
Error	
Error	

No2 で配布したエラープログラム一覧

Attribute Error

```
# Add your Python code here. E.g.  
from microbit import *
```

```
while True:  
    display.croll('Hello, World!')  
    display.show(Image.HEART)  
    sleep(2000)
```

SyntaxError

```
# Add your Python code here. E.g.  
from microbit import *
```

```
while True  
    display.scroll('Hello, World!')  
    display.show(Image.HEART)  
    sleep(2000)
```

NameError

```
# Add your Python code here. E.g.  
from microbit import *
```

```
while True:  
    display.scroll('Hello, World!')  
    display.show(image.HEART)  
    sleep(2000)
```

IndentationError

```
# Add your Python code here. E.g.  
from microbit import *
```

```
while True:  
    display.scroll('Hello, World!')  
display.show(Image.HEART)  
    sleep(2000)
```

ImportError

```
# Add your Python code here. E.g.  
from microbit import random
```

```
while True:  
    display.scroll('Hello, World!')  
    display.show(Image.HEART)  
    sleep(2000)
```

答え

Attribute Error 6 行目の croll が間違っていて， scroll が正しい	SyntaxError 5 行目の True が間違っていて， True:が正しい
NameError 7 行目の image が間違っていて， Image が正しい	IndentationError 7 行目の display が間違っていて⇨display が正しい
ImportError 2 行目の random が間違っていて， *が正しい	

micro:bit No3 変数代入とコメントアウトとキー操作

_____組_____番 氏名_____

プログラムを書いていると、どこまではうまくいっていて、どこからうまくいかないのかを切り分けるために、コメントアウトをよく使います。その操作の練習をしよう。

コメントアウト, アンコメント Ctrl + /

元に戻りたい場合 Ctrl + z

※複数行の場合、カーソルの位置から Shift を押しながら ↓ を押して複数行を選択して Ctrl + /

	コード	解説
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.		
4.		53 という整数値を a という変数に代入する。
5.	a = 53	※プログラムは右の値を左に代入するという表現をよく使います。
6.	b = 59	
7.	c = 67	
8.	d = 4661	13 と "13" の違いは数値の 13 か文字列の 13 か
9.	e = "13"	f という変数に 40 という文字列を入れる。
10.	f = "40"	
11.	g = "0"	
12.		
13.	ans = a + b	
14.	ans = a - c	
15.	ans = b * c	
16.	ans = d / b	数値と文字列を足してみたらどうなるか？
17.	ans = a + e	文字列と文字列を足したらどうなるか？
18.	ans = e + f	文字列から文字列を引いたらどうなるか？
19.	ans = f - g	
20.		
21.	while True:	
22.	display.scroll(ans)	

課題 3-1★ 次の①～⑨の課題を行い、マイクロビットに書かれている表示を記入してみよう。エラーが出る場合、何エラーなのかも書いてください。(このプリントの写真提出)

- ① まずは 13 行目から 19 行目までコメントアウトしてください。
- ② 13 行目だけアンコメントし実行した際に、どんな表示になるか。 表示_____
- ③ 14 行目だけアンコメントし実行した際に、どんな表示になるか。 表示_____
- ④ 15 行目だけアンコメントし実行した際に、どんな表示になるか。 表示_____
- ⑤ 16 行目だけアンコメントし実行した際に、どんな表示になるか。 表示_____
- ⑥ 17 行目だけアンコメントし実行した際に、どんな表示になるか。 表示_____
- ⑦ 18 行目だけアンコメントし実行した際に、どんな表示になるか。 表示_____
- ⑧ 19 行目だけアンコメントし実行した際に、どんな表示になるか。 表示_____
- ⑨ 13～19 行目をアンコメントし実行した際に、どんな表示になるか。 表示_____

課題 3-2 わかったこと★★ 少なくとも三つは記述 (このプリントの写真提出)

- ・割り算の答えのこと。
- ・文字列と数値の足し算のこと
- ・文字列と文字列の足し算のこと。
- ・文字列と文字列の引き算のこと。

課題 3-3★★★★ 自分で上のプログラムを以下のような行為を自分でプログラミングし、どのような表示がでたかを書いてみよう。(このプリントの写真提出)

- ① 0 で割ってみる。
- ② $10 / 3$ をしてみた場合、小数点第何位まで表示されるか。
- ③ $a = a + b$ をし、21 行目の `display.scroll(ans)` を `display.scroll(a)` に変えてみたらどうなるか。

左側には計算部分の 1 行分のコードを記入

- ① _____ 表示結果_____
- ② _____ 表示結果_____ 第 () 位まで表示
- ③ _____ 表示結果_____

課題 3-4★★★★ 変数 x に 7、変数 y に 3 を代入し、その変数同士の足し算の答えを z という変数に入れ、表示で「No10」となるようなプログラムを書いてください。(プログラムの写真提出)

ヒント

- ① 計算結果を整数値ではなく、文字列の型にしたい場合、 $z = \text{str}(z)$ とすれば、文字列に変換されます。
- ② それを 3-1 の 18 行目のような処理をすれば表示できます。

micro:bit No4 LED イメージの自作

_____組 _____番 氏名_____

LED のディスプレイに表示されるイメージはプリセットとして予め登録されていますが、自分でイメージを自作したい場合があります。そんな場合にどうすればよいか勉強していきましょう。

Microbit の LED ディスプレイは縦 5 個、横 5 個の赤色の LED からできています。LED を光らせる明るさは 0（オフ）から 9（最大の明るさ）の 10 段階で LED ごとに指定できます。

	コード	解説
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.		
4.	jan = Image("00000:"	行ごとに区切って各 LED の明るさを指定
5.	"00000:"	コピー＆ペーストをしながら、編集すると楽です
6.	"09999:"	(Ctrl+C, Ctrl+V)。ここでは最大の明るさの 9 を指定
7.	"99999:"	
8.	"99999")	一番最後はコロン（:）なし
9.		という内容を jan という変数名にしておく
10.	jan = Image("09090:"	上の内容をコピーし、適宜書き換えていく。
11.	"09090:"	
12.	"09090:"	
13.	"09999:"	
14.	"99999")	
15.		
16.	jan = Image("90909:"	
17.	"90909:"	
18.	"90909:"	
19.	"99999:"	
20.	"99999")	
21.	display.show(jan)	ディスプレイに「jan」を表示させる。
22.		

課題 4-1★ グー、チョキ、パーを表示させてみよう。変数名は同じでいいです。コメントアウトを用いて切り替えてください。（プログラムの画面の写真提出）

※上のプログラムのままだと jan という変数が上書きされていき、16 行目の変数 jan の内容が表示されるので、16 行目から 20 行目までをコメントアウトし、チョキの表示を出してみたりしよう。

課題 4-2★★ 自分が作った LED イメージを点滅させたいこともあると思います。そんな場合は、`display.show` と `display.clear()` を交互に繰り返し、間に `sleep` 時間を設ければ点滅ができます。(プログラムの画面の写真提出)

	コード	解説
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.		
4.	# jan = Image("00000:")	
5.	# "00000:"	
6.	# "09999:"	
7.	# "99999:"	
8.	# "99999")	
9.		
10.		
11.	略	行数に注意してください。20 行目までとびます。
20.	"99999")	
21.		
22.	while True:	
23.	display.show(jan)	jan という変数を表示させる。
24.	sleep(200)	200 ミリ秒とめる。
25.	display.clear()	消す。
26.	sleep(200)	200 ミリ秒とめる。
27.		

課題 4-3★★ 点滅させるプログラムはどんな時に使うと効果的か簡単に意見を書いてください。

課題 4-4★★ 自作イメージをたくさんつくり、アニメーションにすることもできます。(micro:bit の LED 画面の動画提出)

	コード	解説
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.		
4.	rock = Image("00000:"	
5.	"00000:"	
6.	"09999:"	
7.	"99999:"	
8.	"99999")	
9.		
10.	scissors = Image("09090:"	上のものをコピーし、適宜書き換える
11.	"09090:"	
12.	"09090:"	
13.	"09999:"	
14.	"99999")	
15.		
16.	paper = Image("90909:"	
17.	"90909:"	
18.	"90909:"	
19.	"99999:"	
20.	"99999")	
21.	jan = [rock, scissors, paper]	jan という名前で リスト集 を作る。 ※プログラムの的には配列という。 [や]はエンターキーの近くにあります。
22.	while True:	
23.	display.show(jan, delay=200)	jan というリスト集を再生し、遅延時間 200 ミリ秒を設定する。
24.		

課題 4-5★★★★イメージの枚数や秒数も調整して自分なりのアニメーションを作成してみよう。(micro:bit の LED 画面の動画提出)。どんなアニメーションでもいいです。

micro:bit No5 疑似乱数と IF 文

組 番 氏名

ゲームなどでランダムな数字を発生させ、その値に応じてガチャのアイテムを引かせたりする場合があります。どのように実装しているのでしょうか？

	コード	解説
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.	import random	今回はここが追加されることに注意！
4.		
5.	while True:	インデント下がった部分をずっと行う。
6.	val = random.randint(1,6)	randint はランダム of 整数値を返し、変数 val に代入。 この場合 1～6 まで。int は INTEGER の略。
7.	display.scroll(val)	変数 val に入った値を表示させる。
8.		

課題 5-1★ 上記のプログラムでランダムが生成されるかどうか調べる。(プログラムの画面の写真提出)

疑似乱数 Tips

乱数とはいっても本当にランダムで発生しているわけではありません。実は一定の計算をして、表示させ、それを繰り返していくと、「全体的にランダムに実行されたように見える」という仕組みになっています。計算を読み解くと、次にでてくる値が予測できます。昔は線形合同法が有名でしたが、ランダム性がないと指摘されたので、現在メルセンヌツイスターという疑似乱数のアルゴリズムが使われることが多いです。microbit もメルセンヌツイスターを利用しています。

著作権配慮のため省略

そろそろ知っておきたいプログラムの意味。

オブジェクトとメソッド ○○○.△△△というプログラムをよく見かけます。オブジェクトとは、○○○にあたる部分でモノと抽象化され、メソッドとは△△△にあたる部分でオブジェクトをどう扱うかを定めた方法です。display.scroll はディスプレイ (モノ) をスクロールさせる (方法) という意味です。

	コード	解説
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.	import random	
4.		
5.	while True:	
6.	val = random.randint(1,3)	
7.	# display.scroll(val)	表示がうまくいったらコメントアウトさせておこう
8.	if val <= 1:	<=は数値が今回3種類なので、==でも可能。
9.	display.scroll("GU")	
10.	elif val <= 2:	<=は数値が今回3種類なので、==でも可能。
11.	display.scroll("CHO")	
12.	else:	
13.	display.scroll("PA")	

課題 5-2★ IF文を使い値によって別々に表示させるようにしよう。(プログラムの画面の写真提出)

IF文の基本

2通りの場合	3通り以上の場合
<pre>if 条件式 1: 処理 A else: 処理 B</pre>	<pre>if 条件式 1: 処理 A elif 条件式 2: 処理 B else: 処理 C</pre>

条件式の比較演算子について

意味	比較演算子	意味	比較演算子
A と B は同じ	A == B	A は B 以上	A >= B
A と B は違う	A != B	A は B より小さい	A < B
A は B より大きい	A > B	A は B 以下	A <= B

※A や B は変数や数値が入る。=一つであると代入を意味するので、イコールは==と記入。

数値が多くなった場合の IF の使い方 疑似乱数が 1~100 の整数の中で 100 通り発生し、それぞれ処理 A の確率が 30%、処理 B の確率が 10%、処理 C の確率が 60%をさせたい場合、右のように記述する。	<pre>if x <= 30: 処理 A elif x <= 40: 処理 B else: 処理 C</pre>	考え方：30 以下の場合は処理 A (31~) 40 以下の場合は処理 B それ以外は処理 C というように記述する。
--	---	--

裏へ

課題 5-3★★ 上のじゃんけん表示において、No4 で行った LED でじゃんけんマークを表示させるようにしよう。(microbit の LED の動画提出) ヒント：5 行の変数代入と `display.show(〇〇)`、`sleep(〇〇)` を使います。

ヒント：プログラムのヒントです。

```
cho = Image("90909:"  
            "90909:"  
            "90909:"  
            "99999:"  
            "99999")  
display.show(cho)  
sleep(2000)
```

課題 5-4★★ ゲーム：チョキ：パーの出る割合が 1:2:3 になるようなプログラムを作ろう。ヒント：ランダムを 6 通り発生させ、条件式の比較演算子をうまく使おう。じゃんけん表示は LED を自作して、`display.show(gu)` を使ってもよいし、`display.scroll("gu")` を使ってもよい。(プログラムの画面の写真全行提出)

micro:bit No6 ボタンと IF 文と疑似乱数

組 番 氏名

IF 文はよく使うので繰り返し学習していきます。ボタンの使い方 IF 文です。

	コード	解説
1.	# Add your Python code here. E.g.	ランダムオブジェクトを使うためランダムを import。 じゃんけん変数を設定しておく。 今まで"gu"と書いていたが、gu という変数を扱いたい場合はそのまま gu と書く。
2.	from microbit import *	
3.	import random	
4.		
5.	gu = Image("00000:"	
6.	"00000:"	
7.	"09999:"	
8.	"99999:"	
9.	"99999")	
10.		
11.	cho = Image("09090:"	一回だけ実行される部分 ここに今後使う変数を書いていく。
12.	"09090:"	
13.	"09090:"	
14.	"09999:"	
15.	"99999")	
16.		
17.	pa = Image("90909:"	
18.	"90909:"	
19.	"90909:"	
20.	"99999:"	
21.	"99999")	
22.		乱数を発生させる ボタンが押されたという if 文。ここは was で設定 if 文の中にさらに if 文を書くことも可能。 gu という変数名を扱いたい場合は gu gu と表示させたい場合は"gu"
23.	while True:	
24.	x = random.randint(1,3)	
25.	if button_a.was_pressed():	
26.	#display.scroll(x)	
27.	display.scroll("janken")	
28.	if x <= 1:	
29.	display.show(gu)	
30.	elif x <= 2:	

※No6 は全角スペースを入れさせないように注意

31.	display.show(cho)	
32.	else:	
33.	display.show(pa)	
34.	if button_b.is_pressed():	ここまで書いたら下は書かずに一回実行してみる。 ここは is で設定してみる。
35.	display.scroll("aikode")	
36.	if x <= 1:	何回も繰り返される部分。
37.	display.show(gu)	インデント（タブ）を使いながら繰り返 したい処理を書いていく。
38.	elif x <= 2:	
39.	display.show(cho)	
40.	else:	
41.	display.show(pa)	

is と was の違い～挙動をそれぞれ確認してみよう～

is は前処理が完全に終わらないと反応しない。was は前処理が実行中でも押しておけば反応する。

課題 6-1★ 上記のプログラムを書く。a ボタン b ボタンはディスプレイの横にある。(LED の動画提出)

課題 6-2★★ 以下はランダムに数字を発生させ、その結果によってじゃんけんの文字をスクロールさせるプログラムである。以下のプログラムで間違いな部分はどこか答えなさい。(プリントの写真提出)

	コード	
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.	import random	
4.		(A.) 行目のコードを
5.	x = random.randint(1,3)	
6.	while True:	(B.) 行目と (C.) 行目の間に
7.	if button_a.was_pressed():	
8.	display.scroll("janken")	もっていかなければいけない。
9.	if x <= 1:	
10.	display.scroll("gu")	
11.	elif x <= 2:	
12.	display.scroll("cho")	
13.	else:	
14.	display.scroll("pa")	

課題 6-3★★ a ボタンが押されたら、さいころ 1 個をふった目（・など）を表示させよう。変数名は先頭に関してアルファベットしか使えません。one,two,などの変数名を設定しよう。display.show と sleep も使います。(LED の動画提出)

課題 6-4★★★★ a ボタンが押されたら、さいころ **1 個をふった目（・など）** を表示させ、b ボタンが押されたらさいころ **2 個ふった目の数（1～12）** を表示させよう。また、さいころ 2 個振った場合は、同様に確からしい状態に表示させること。（プログラムの写真提出。すべての行を撮って提出すること。）

ヒント 1：さいころ 1 個の値を変数 x,y にそれぞれ入れて、足した値をさらに変数 z などに代入し、変数 z を表示させること。

ヒント 2：1～9 までは IF を使って、display.show でいけるが、10,11,12 は別の方法で表示させる必要がある。LED の表示を 10,11,12 は自作し、IF 文などを使って表示させていこう。

ヒント 2 の例

10 の場合の LED の例

```
ten=("90999:"  
    "90909:"  
    "90909:"  
    "90909:"  
    "90999")
```

```
if z <=9:  
    display.show(z)
```

micro:bit No7 音楽利用とループと初期化

組 番 氏名

ここからは、ループに入った時のプログラムとループに入る前のプログラムでコードをどこに記述していけばよいのか考えてみよう。

タイマー 10

	コード	
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.	import music	今回はここが入ることに注意。
4.		
5.	count = 0	ループに入る前に count 変数を 0 にしておく (これを初期化という。)
6.	while True:	a ボタンが押された時 10 ずつ増やしていく
7.	if button_a.was_pressed():	
8.	count = count + 10	
9.	display.scroll(count)	
10.	if button_b.was_pressed():	b ボタンが押された時 ずっと繰り返す。 0 なら終了させる。 音楽を鳴らす。 リセットさせ、最初の状態に戻る 値を表示させ、 1 を引いていく。count -= 1 という書き方 でもよい。
11.	while True:	
12.	if count == 0:	
13.	music.play(music.RINGTONE)	
14.	reset()	
15.	display.scroll(count)	
16.	count = count - 1	

count 変数がループに入る前にあることに注意。最初に 0 にしておく。初期化という。while の中にあったりすると、ループする度に 0 になってしまい、プログラムとして成り立たなくなる。

課題 7-1★ 上記のプログラムを書く。(プログラムの画面の写真提出)

課題 7-2★★ 次のコードは上の 12~14 行目と 15~16 行目を入れ替えただけですが、プログラムのバグなプログラムです。

	コード	
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.	import music	

4.		
5.	count = 0	
6.	while True:	
7.	if button_a.was_pressed():	
8.	count = count + 10	
9.	display.scroll(count)	
10.	if button_b.was_pressed():	
11.	while True:	
12.	display.scroll(count)	
13.	count = count - 1	count -= 1 という書き方でもよい。
14.	if count == 0:	
15.	music.play(music.RINGTONE))	
16.	reset()	

なぜなのか理由を書いてみよう。(このプリントの写真提出)

検証方法：プログラムを転送したあと、a ボタンを押さずに b ボタンから押してみる。

課題 7-3★★ (人数チェッカーの制作) a ボタンを押したら、押した回数をカウントし表示させ、b ボタンが押されたら、その値を表示させる。a ボタンと b ボタンが同時に押されたら、リセットさせる。
 ヒント:a と b の同時押しは、if button_a.is_pressed() and button_b.is_pressed(): です。同時押しは、a ボタンを押しながら、b ボタンを押すと実現できます。リセットは本当にリセットされたかわからないので、直前で音楽を鳴らしたり、直前で display.scroll("reset")としたりして、わかりやすくしよう。(プログラムの写真全行提出)

課題 7-4★★★★ 以下のような表示になるためにはどのようにすればよいか、プログラムをインターネットで調べながらループ (while あるいは for) を使って書きなさい。(プログラムの画面の写真提出)

表示	ヒント
1 12 123 1234 12345	- python の for 文の書き方を検索して調べる。 - 数字を for 文で 1 ずつあげていく。 - あげた数字を文字列にして、文字列の変数に結合させる。(文字列の結語は文字列同士の+でいけます。)

micro:bit No8 様々なセンサー

組 番 氏名

microbit の中には様々なセンサーが入っています。それらの使い方について考えていきましょう。

加速度センサー

accelerometer オブジェクトを使い、様々なメソッドを使うことで実現できます。左右 x 軸では、get_x() メソッドを利用すれば、x 軸の傾きを取得することができ、同じく get_y(), get_z() も使用することができます。さらに、microbit へのジェスチャーがプリセットとして登録されており、current_gesture() メソッドを使うことで登録された行為を文字列で読み取ることができます。

傾きの読み取り

	コード	
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.		
4.		
5.	while True:	
6.	reading = accelerometer.get_x()	そのほかに get_y(), get_z() がある。
7.	if reading > 20:	数値が低いととても敏感に反応してしまうため、閾値として 20 ぐらいを設定しておく。
8.	display.show("R")	プラスなら右方向
9.	elif reading < -20:	
10.	display.show("L")	マイナスなら左方向
11.	else:	
12.	display.show("-")	

課題 8-1★ 上記のプログラムを書いてみよう。(プログラムの画面の写真提出)

ジェスチャーの読み取り

	コード	
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.		
4.	while True:	
5.	gesture = accelerometer.current_gesture()	
6.	if gesture == "shake":	他に up, down, left, right, face up, face down,

7.	display.scroll("furareta")	freefall, shake がある。
8.	else:	
9.	display.scroll(".")	

これが紹介されているサンプルプログラムですが、ずっと実行していると、前処理の時間が延長されていき、反応が鈍くなるので、IF 文だけにして、リセットさせるほうが現実的かもしれないです。

現実的なジェスチャープログラム

	コード	
1.	# Add your Python code here. E.g.	display.scroll("futta")と書きましたが、ここに何かのプログラムを書いていき、最後に reset() で再起動させます。
2.	from microbit import *	
3.		
4.	while True:	
5.	gesture = accelerometer.current_gesture()	
6.	if gesture == "shake":	
7.	display.scroll("futta")#何かの処理	
8.	reset()	
9.		

課題 8-2★ 上記のプログラムを書いてみよう。(プログラムの画面の写真提出)

温度センサー

	コード	
1.	# Add your Python code here. E.g.	一度、7 行目まで書いて、7 行目までをアンコメントして実行させる。
2.	from microbit import *	
3.		
4.	while True:	
5.	temp = temperature()	
6.	#display.scroll(temp)	
7.	if temp > 30:	
8.	display.scroll("HOT")	
9.	else:	
10.	display.scroll("COOL")	
11.		

課題 8-3★ 上記のプログラムを書いてみよう。(プログラムの画面の写真提出)

※裏側の左上の 7mm 程度の正方形のチップが温度センサー。7 行目までを書いて、**指で温めると数値が分かります**。カップラーメンの上などに置くなどのアイデアを使った作品の場合、開発段階では温度を再現できないので、8 行目の temp の比較演算子の数値を低くしたりして状況を疑似的に再現しよう。

裏へ

課題 8-4★★ microbit を右に傾けたら"right"と表示させたあと、おみくじがでるアプリを作ろう。おみくじの確率は大吉：中吉：小吉：凶：大凶=6:4:3:2:1 になるように設定しよう。

(プログラムの画面の写真提出。複数枚になる場合、複数枚の写真提出。) ヒントは疑似乱数を 1 から 6+4+3+2+1 の合計の値の間で発生させ、IF 文で条件分けをしていこう。

ヒント

1:2:3 のグーチョキパーの場合 (6 種類の整数で)	
<pre>if x == 1: display.show(gu) elif x <= 3: display.show(cho) else: display.show(pa)</pre>	①1 の場合に実行される。 ②3 以下の場合に実行される。1 は上で実行されるため、事実上 2,3 の場合で実行される。 ③それ以外の時に実行される。事実上、4,5,6 の時に実行される。

課題 8-5★★★ microbit を左に傾けたら"left"と表示させたあと、30 度以上になったら 10 からカウントダウンするプログラムを書く。指で温めるとカウントが開始するように作る。(プログラムの画面の写真全行提出)

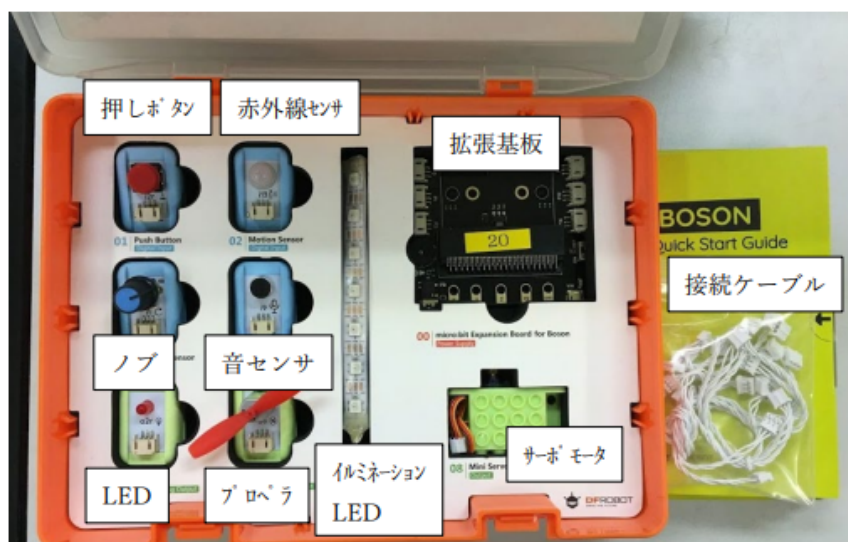
8-4 と 8-5 のヒント

	コード	
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.		
4.		
5.	while True:	
6.	gesture = accelerometer.current_gesture()	
7.	if gesture == "right":	
8.	display.scroll("right")	
9.	while True:	
10.	#おみくじプログラム 最後に reset()を使う	
11.	if gesture == "left":	
12.	display.scroll("left")	
13.	while True:	
14.	#温度を計測して、30 度以上なら 10 からカウントダウンするプログラム #最後に reset()を使う。	

micro:bit No9 拡張キットの利用

組 番 氏名

今回より micro:bit に接続して使える拡張キット (Boson Starter Kit for micro:bit) を使っていきます。この拡張キットは 2 人で 1 つずつ貸し出します。授業後は返却してもらい、またほかのクラスの授業で使うので大切に使用してください。以下が内容の紹介です。



Boson Starter Kit for micro:bit

拡張部品を使用するときは、まず拡張基板を micro:bit につながします。
次に部品を接続ケーブルで P0～P16 のいずれかに接続します。
ただし、P0 は音楽と共用しているので、音楽を使うときは P0 は利用できません。

外部モジュールへの伝達方法

write_digital、read_digital … 0(ロー)か、1(ハイ)かしか表現できない。
write_analog、read_analog … 0(0.0V)から 1023(1.0V)の間で表現できる。

赤外線センサー（デジタル）、ノブ（アナログ）、プロペラ（デジタル、アナログ両方）

赤外線センサーを P0 に、ノブを P1 に、プロペラを P2 に接続した場合

1. a ボタンを押すと、赤外線センサーが反応した時に、プロペラを 3 秒回す。
2. b ボタンを押すと、ノブを回した時、回した大きさによってプロペラの回転数を上げる。

	コード	
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.		
4.		
5.	while True:	
6.	if button_a.was_pressed():	
7.	while True:	
8.	sensor = pin0.read_digital()	赤外線センサーの値を読み取り sensor という変数に代入
9.	# display.scroll(sensor)	ここまで書いてアンコメントさせてテスト
10.	if sensor == 1:	センサーが反応したら
11.	pin2.write_digital(1)	端子をハイにする。
12.	sleep(3000)	3 秒間
13.	pin2.write_digital(0)	端子をローにする。
14.	if button_b.was_pressed():	
15.	while True:	
16.	val = pin1.read_analog()	ノブの値を val という変数に代入
17.	# display.scroll(val)	ここまで書いてアンコメントさせてテスト
18.	pin2.write_analog(val)	val の値の大きさにプロペラの回転数を制御

9-1★ 上のプログラムを書く。（プログラムの画面提出）

指示のある個所でテストを行い、すべて書いた後は実行して反応を確かめよう。

※赤外線センサーは机の上においてそっとしておくと 0 の状態に戻ります。ランプで確認してください。

※アナログのプロペラは値が大きくても回らない場合があります。そういう場合は手でアシストしてあげると動き出します。

LED (ディジタル)、赤ボタン (ディジタル)、音センサー (アナログ)

LED を P0 に、ボタンを P1 に、音センサーを P2 に接続した場合。

1. a ボタンを押すと、赤いボタンを押したら、2 秒 LED が光る。
2. b ボタンを押すと、うるさかった場合、ANGRY の顔文字になり、そうでない場合は、ASLEEP の顔文字になる。

	コード	
1.	# Add your Python code here. E.g.	
2.	from microbit import *	
3.		
4.		
5.	while True:	
6.	if button_a.was_pressed():	
7.	while True:	ボタンという変数に pin1 の赤ボタンの
8.	button = pin1.read_digital()	ディジタル値を読み取る。
9.	#display.scroll(button)	ここまで書いてアンコメントさせてテスト
10.	if button == 1:	赤ボタンが押されたら
11.	pin0.write_digital(1)	端子をハイにする。
12.	sleep(2000)	2 秒光らせる。
13.	pin0.write_digital(0)	端子をローにする。
14.	if button_b.was_pressed():	
15.	while True:	
16.	sound = pin2.read_analog()	端子のアナログ値を読み取る。
17.	#display.scroll(sound)	ここまで書いてアンコメントさせてテスト。
18.	if sound > 300:	300 という値で雑音判定する。
19.	display.show(Image.ANGRY)	
20.	else:	
21.	display.show(Image.ASLEEP)	

9-2★ 上のプログラムを書く。(プログラムの画面提出)

指示のある個所でテストを行い、すべて書いた後は実行して反応を確かめよう。

このように使い方を見てみると、write_digital、read_digital か write_analog、read_analog をそれぞれ部品によって使い分けていけばいいですね。

NEOPIXEL – RGB LED Strip

RGB LED Strip を P8 に接続する。Neopixel というライブラリを読み込む必要がある。

上から 3 つ、次の 3 つ、最後の 1 つが順番に光るサンプルプログラム

	コード	
1.	# Add your Python code here. E.g.	

2.	from microbit import *	
3.	import neopixel	neopixel ライブラリ
4.		を読み込む
5.	np = neopixel.NeoPixel(pin8, 7)	7 つピクセル Neopixel ボードが P8 に接続
6.		されていることを設定
7.	while True:	
8.	np[0] = (255,0,0)	最初が 0 であること
9.	np[1] = (0,255,0)	に注意
10.	np[2] = (0,0,255)	
11.	np[3] = (0,0,0)	
12.	np[4] = (0,0,0)	
13.	np[5] = (0,0,0)	合計 7 つなので 6 で
14.	np[6] = (0,0,0)	終わる
15.	np.show()	点灯させる
16.	sleep(1000)	1 秒間止まる = 持続
17.	np[0] = (0,0,0)	
18.	np[1] = (0,0,0)	
19.	np[2] = (0,0,0)	
20.	np[3] = (255,255,0)	
21.	np[4] = (0,255,255)	
22.	np[5] = (255,0,255)	
23.	np[6] = (0,0,0)	
24.	np.show()	点灯させる
25.	sleep(1000)	1 秒間止まる = 持続
26.	np[0] = (0,0,0)	
27.	np[1] = (0,0,0)	
28.	np[2] = (0,0,0)	
29.	np[3] = (0,0,0)	
30.	np[4] = (0,0,0)	
31.	np[5] = (0,0,0)	
32.	np[6] = (255,255,255)	
33.	np.show()	点灯させる
34.	sleep(1000)	1 秒間止まる = 持続

9-3★★ 上のプログラムを書く。(プログラムの画面提出)

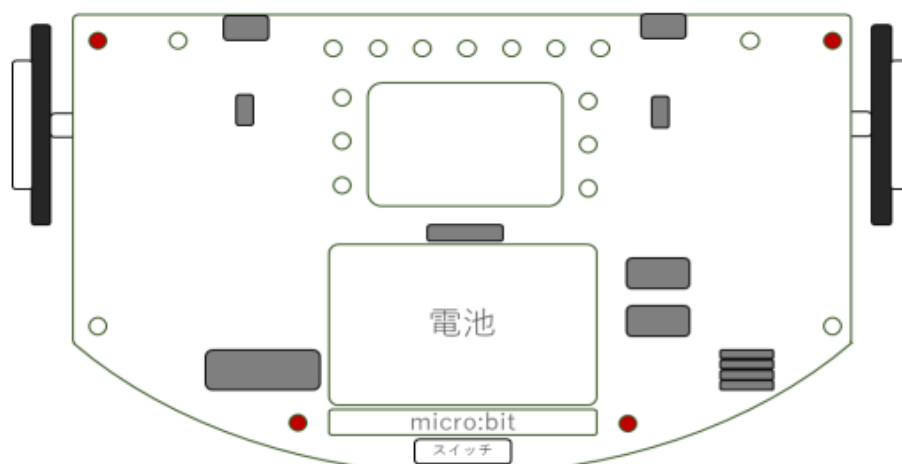
9-4★★★★ 上のプログラムを参考に、好きなタイミングで色や LED を光らせる。(プログラムの画面提出)

micro:bit No10 サーボモータの利用

組 番 氏名

前半は配膳ロボの準備

1. 赤の部分のネジを止める。また、棒の先端に段ボールの受けのための部品を取り付ける。



サーボモーターを使用してみよう

サーボモーターは、一般的なモーターと異なり、動作に必要な基盤やギヤが装着されている状態のモーターです。また、あらかじめ決められた回転角度の範囲内で回転し、その角度を維持するように動作させることもできます。

	コード	
1.	from microbit import *	
2.		
3.	pin2.set_analog_period(20)	電流を流す間隔を指定する
4.		
5.	def conv(angle):	angle から数を受け取る関数 conv を定義
6.	k1 = 33 + 0.44*angle	このモータは 33 の状態で 0 度になり、0.44
7.	return(k1)	につき 1 度動く。angle に対する必要な電流
8.		の計算結果を k1 とし、k1 を答えとして返す。
9.	while True:	
10.	if button_a.is_pressed():	ボタン a を押した時は
11.	pin2.write_analog(conv(180))	180 度の位置になるように電流を流す
12.	sleep(500)	500 ミリ秒止まる

13.	<code>elif button_b.is_pressed():</code>	ボタン b を押した時は
14.	<code>pin2.write_analog(conv(0))</code>	0 度の位置になるように電流を流す
15.	<code>sleep(500)</code>	500 ミリ秒止まる
16.	<code>else:</code>	
17.	<code>pin2.write_analog(conv(90))</code>	ボタンが押されていないときはハートを表
18.	<code>display.show(Image.HEART)</code>	示させ、90 度の位置に戻す電流を流す

10-1★ 上のプログラムを書き実行してみよう。(プログラムの画面提出)

自分の microbit に電池ボックスをつなぎ、さらに拡張基盤を差して、P2 とサーボモーターとをケーブルでつなぐ。

！注意！ サーボモーターを動かすためには大きな電力が必要です。電池ボックスをつなぎ忘れたり、スイッチが OFF になっていると、LED の表示が点滅したり、モーターが動かなかったりします。

10-2★★ タイマー10 で計測が終了したときに、サーボモーターが動くようにしよう。(プログラムの画面を提出)

ヒント タイマー 1 0

	コード	
1.	<code># Add your Python code here. E.g.</code>	
2.	<code>from microbit import *</code>	
3.		
4.	<code>count = 0</code>	ループに入る前に count 変数を 0 にしておく (初期化)
5.	<code>while True:</code>	
6.	<code>if button_a.was_pressed():</code>	a ボタンが押された時
7.	<code>count = count + 10</code>	10 ずつ増やしていく
8.	<code>display.scroll(count)</code>	
9.	<code>if button_b.was_pressed():</code>	b ボタンが押された時
10.	<code>while True:</code>	ずっと繰り返す。
11.	<code>if count == 0:</code>	0 なら終了させる。(a ボタンを押されてない場合)
12.	<code>display.scroll("finish")</code>	
13.	<code>reset()</code>	リセットさせ、最初の状態に戻る
14.	<code>display.scroll(count)</code>	値を表示させ、
15.	<code>count = count - 1</code>	1 を引いていく。count -= 1 という書き方でもよい。

時間があまったら前回の課題を行おう

6. 分析と考察

ここからは、これまでに行ってきたプログラミング教育に関する分析と考察を行っていく。

6.1. タイピング速度

1学期のタイピング学習では、秒間2.0タイプ以上打てることを目標にして、キーボードの「中段」「中段と上段」「中段と上段と下段」そして、数字が含まれる「中段と上段と下段と最上段」の4段階に分けて学習した。その結果、「中段」合格者は97.2%で「中段と上段」合格者83.3%で「中段と上段と下段」合格者61.1%、「中段と上段と下段と最上段」合格者は27.8%であった。2学期のはじめに再び計測する予定であったが、緊急事態宣言のため、オンライン授業が長引き、時間の余裕がなくなり計測はできなかった。ただ、プログラミング学習を始める前段階でタイピング速度を向上させることはとても重要なことではないかと考えている。

また、このタイピングの向上と一緒にショートカットキーやコマンドを学習していけるとより一層指導が楽になると考えている。プログラム記述の際に、似たような部分はコピー&ペーストして、コピー&ペーストした後で、少し書き換えていく。このようにすることで、変更した差分を頭の中で理解する時間を確保できるようになると考えている。

以上のようにタイピングやショートカット、コマンド理解は欠かせないものであると考える。

6.2. 事前アンケート

学習の事前と事後のプログラミングに対しての考え方の「差異」を測るために、アンケートを行うことにした。

内容は学習指導要領解説の第一章の総説「3、資質・能力の三つの柱に沿った情報活用能力の整理」から、①知識及び技能と②思考力、判断力、表現力等、③学びに向かう姿力、人間性等の3つの項目に分けて独自に考えた。以下がそのアンケート項目である。

プログラミング行為をするのに際し、必要な資質や能力はどのようなものであると考えられるか。5段階評価で答えてください。

①知識及び技能

Q1,プログラミングでは、PCの基本操作ができることは大切である。

Q2,プログラミング周りの基本操作（コメントアウト、コマンド利用）ができることは大切である。

Q3,正確にテキスト（文字）を入力することは大切である。

②思考力、判断力、表現力等

Q4,プログラムすべてを書いてから実行するのではなく、小さく実行していくことは大切である。（分解）

Q5,順序を考えてプログラミングしていく思考力は必要である。（順序立て）

Q6,バグや論理エラーを直すための思考力、想像力、判断力は必要である。（分析）

Q7,情報を収集して、使えそうな情報を取捨選択する力は必要である。

Q8,行為の本質をとらえ、共通点を見つけ他のいらない部分は捨てながらプログラムする能力は必要である。(抽象化)

Q9,プログラムを関数にしたり、クラスにしたりして行為を一般化する力は必要である。(一般化)

③学びに向かう姿力、人間性等

Q10,プログラムを実行し、トライアル&エラーを繰り返すことは大切である。

Q11,プログラミングで新しい価値を生み出そうとするチャレンジ精神は必要である。

Q12,仲間と協働してプログラミングし、解決する力は大切である。

この項目について、回答を 1,まったく思わない 2,あまりそう思わない 3,どちらともいえない 4,ややそう思う 5,とてもそう思うに分類し、答えてもらった。そして、それぞれの項目の数値について、平均値や最頻値を出し、それを分析することとする。

また、アンケート内容も、プログラミング言語記述方法習得については、割愛した。学習指導要領解説からも、「プログラミング言語を使用するとともに、アルゴリズムやプログラムの記述方法の習得が目的にならないよう取扱いに配慮する。」と書かれているからである。実際に一つのプログラミング言語ができると、他の言語の記述方法の習得は検索して調べたりすることで解決できるからである。

また、プログラミング的思考力という5つの要素を盛り込んだ。Q4,Q5,Q6,Q8,Q9 の「分解」「順序立て」「分析」「抽象化」「一般化」である。筆者の仮説から考えるに、Q8 の抽象化と Q9 の一般化という行為は時間がかかると考えている。経験の上に積み重なる考え方ととらえているからである。

そして最後の③の学びに向かう姿力、人間性等の内容については評価が難しい。プログラミングの授業を受けて、グループ内の仲間と喧嘩してしまい協働が難しくなったりして低くなってしまったりする可能性があるが、質問を設定することにした。

以下が事前アンケートの結果である(図 15)(表 4)。

事前アンケート

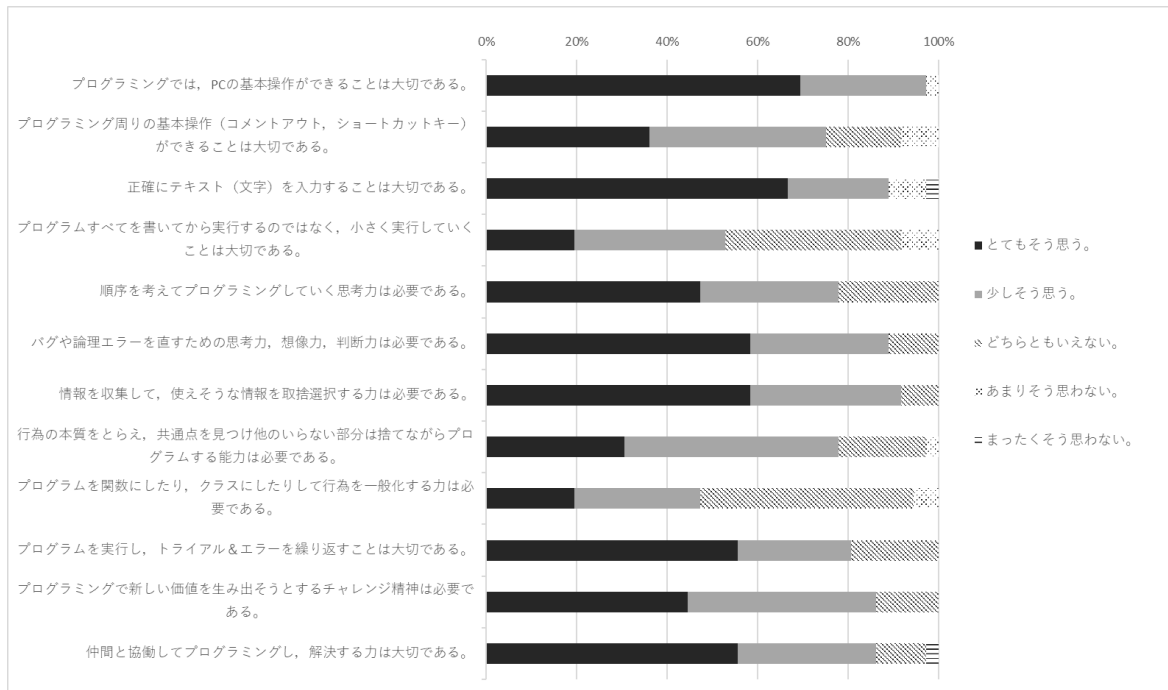


図 15

表 4

	平均値	最頻値	とてもそう思う。 頻度	少しそう思う。 頻度	どちらともいえない。 頻度	あまりそう思わない。 頻度	まったくそう思わない。 頻度
Q1 プログラミングでは、PC の基本操作ができることは大切である。	4.64	5	25	10	0	1	0
Q2 プログラミング周りの基本操作（コメントアウト、ショートカットキー）ができることは大切である。	4.03	4	13	14	6	3	0
Q3 正確にテキスト（文字）を入力することは大切である。	4.42	5	24	8	0	3	1
Q4 プログラムすべてを書いてから実行するのではなく、小さく実行していくことは大切である。	3.64	3	7	12	14	3	0
Q5 順序を考えてプログラミングしていく思考力が必要である。	4.25	5	17	11	8	0	0
Q6 バグや論理エラーを直すための思考力、想像力、判断力が必要である。	4.47	5	21	11	4	0	0
Q7 情報を収集して、使えるような情報を取捨選択する力が必要である。	4.50	5	21	12	3	0	0
Q8 行為の本質をとらえ、共通点を見つけ他のいらない部分は捨てながらプログラムする能力が必要である。	4.06	4	11	17	7	1	0

Q9 プログラムを関数にしたり、クラスにしたりして行為を一般化する力は必要である。	3.61	3	7	10	17	2	0
Q10 プログラムを実行し、トライアル&エラーを繰り返すことは大切である。	4.36	5	20	9	7	0	0
Q11 プログラミングで新しい価値を生み出そうとするチャレンジ精神は必要である。	4.31	5	16	15	5	0	0
Q12 仲間と協働してプログラミングし、解決する力は大切である。	4.36	5	20	11	4	0	1

通常このような5段階評価の順序尺度において、平均を出すことは意味をなさないが、参考程度に出した。では、アンケートについて考察していく。

まず、Q2,Q4,Q8,Q9 の質問に関して、重要度はそう高くないと感じているようである。たしかにこれらはプログラミングを経験しないと体験できないことである。また、Q12 に関してはとても意気込みを感じることが読み取れる。これが高い位置にあるのもグローバルコースならではの生徒の特長と考えている。そして、筆者が個人的にプログラミングにおける欠かせない資質・能力は Q4 の「分解」であると考えている。コンピュータリテラシー全般にもいえることだが、何かうまくいかない場合、そのほとんどは、積み重なりの中での一部欠如から起こることが多い。例えば、ネットワーク（インターネット）に接続されない場合、IP アドレスはもらえているのか、DNS は正しく設定されているか、アクセスポイントに正しく接続されているか、ケーブルは繋がっているかなどである。これらが正しく積み重なって初めてネットワークに接続できる。プログラミングに関しても同様に、積み重なることが多いので、それらを自分の頭の中で分解してそれぞれ考えられる能力が必要であると考え。事後アンケートではこの「分解」という部分について、動向を注目していくとともに、詳しく見ていくことにする。

6.3. Python の基礎課題と進捗状況

Python の基礎学習のために行った課題について分析、考察をしていく。評価は、一つの問題につき3点を上限とし、点数をつけていった。正しく動くプログラムは3点、ちょっとしたバグは2点、プログラムを少しだけ書いたものは1点と点数をつけていった。以下が課題における平均点一覧である。

表 5

1-1	1-2	1-3	1-4	2-1	2-2	2-3	3-1	3-2	3-3	3-4	4-1	4-2	4-3	4-4	4-5	5-1	5-2	5-3	5-4
3	2.21	2.36	1	2.71	1.71	1.11	2.86	2.24	1.41	0.25	2.48	2.15	1.39	1.45	0.91	2.62	2.45	1.34	0.31
6-1	6-2	6-3	6-4	7-1	7-2	7-3	7-4	8-1	8-2	8-3	8-4	8-5	9-1	9-2	9-3	9-4	10-1	10-2	
2.9	1.8	0.57	0	2.87	1.68	1.71	1.43	2.7	2.7	2.5	1.03	0.48	2.81	2.71	2.32	0.29	2.69	1.07	

一コマごとの課題については、後にいけばいくほど難易度があがるので、点数が低くなる傾向にあった。難易度が生徒のスキルの差をうまく吸収できていたと考えている。

次にコマ毎の考察をしていく。No5 や No6 では乱数を扱うために、ランダムモジュールを必要とする。3行目に `import random` とかなければいけないが、事前に説明したものの、それを記述し忘れる生徒がある程

度はでてしまう。ここは机間巡視しながら、指摘してあげる必要があった。No6では、タブやスペースの入力で全角スペースを入力したり、字下げが適切に行われていないケースが目立った。No6はコード量も多く、正確に記述するにはとても難易度が高かった。全角スペースを入力すると Syntax エラーになり、字下げをしないと、Indentation エラーや論理エラーになる。また、全角スペースを入力されると、発見するのに時間がかかってしまう。こちらのほうは、授業導入時に特に生徒に注意を促すべきであると考えている。そして、No8 や No9 のセンサーを使ったプログラミングや、拡張キットを使ったプログラミングは、生徒たちはとても楽しそうに実習しているように見えた。ハードウェアでのプログラミングは生徒の目に映るインパクトが強いらしく、能動的に取り組んでいた。赤外線センサーを使ったプログラミングにおいて、「自動ドアはこのセンサーの大きいものがついているよ」と話したところ、興味深そうに話を聞いていた。

ただ、プログラミングの学習が進み、難易度があがるにつれて、複数の要素が絡み合ったプログラムがどうしても多くなってくる。複数の要素が絡み合うプログラムの場合、すべて記述してから実行させるのではなく、途中まで記述し、実行させ、うまく動作したら次へ、というように、ふくらましながらコーディングするというのが授業では円滑に進める手助けになると考える。いわゆる「分解」という考えである。この分解の考えを得るために、教材の順序や構成は考えておかなければいけない。

6.4. 配膳ロボ基本編

基本編の最中は、配膳ロボに愛着をもつ生徒が多く、とても意欲的に学んでいた。この点はとても評価できる。ただ、エラーやバグに陥ると、直す作業に時間を取られることが多く、能力のある子でも直すのに時間がかかり、課題が進まない状況も見られた。また、授業時間に関しても基本編に割く時間があまりなく、基本動作のプログラムを授業内で完成したグループは全体の 15%ほどだった。しかし、放課後に補習を用意したところ、多くの生徒がPC 教室に足を運んで、プログラミングを楽しんでいた。最終的に 60%ぐらいのグループが基本編をすべて終えることができた。また、これまで micro:bit にプログラムを転送する作業は Python 基礎課題と合わせると 100 回以上になっている生徒が多く、トライアル&エラーをたくさんしていたのも評価できる。

基本編において、論理エラーの場合、バグを簡単には見つけられないなどの問題があった。論理エラー以外の場合は、転送時に micro:bit にエラー表示が現れて、何が間違っているかがわかりやすいが、論理エラーの時はそのままエラー表示がでないのどこがエラーかわかりにくい。特にインデントが適切に行われていないと、論理エラーになるケースが多かった。論理エラーを出された時に、バグを見つけるためには生徒の画面のコードを見に行かなければならない。

例えば以下のコードを書かれた場合、コードの閲覧の注視をしなければならず時間がかかり対応に時間をとられてしまう。まずは正しいコードから記載する(コード 1)。

コード 1

82	#⑦メインプログラム（配膳行為のプログラミング）#
83	display.show(table_marker_count)
84	
85	while True:

86	#motor.set_speed(50,50)
87	trace_line()
88	check_table_marker()
89	check_kitchen_marker()
90	
91	#⑧メインプログラム（指定のテーブルを見つけたら配膳を行う）#
92	if table_marker_count == table_number and serving == True:
93	motor.set_speed(0,0)
94	sleep(2000)
95	servo.set_angle(0)
96	sleep(2000)
97	servo.set_angle(90)
98	serving = False

上記のように、走行中に指定のテーブルを見つけたら配膳を行うプログラムにおいて、次のように書くと、エラーが出ずに走り続けてしまう(コード 2)。

コード 2

82	#⑦メインプログラム（配膳行為のプログラミング）#
83	display.show(table_marker_count)
84	
85	while True:
86	#motor.set_speed(50,50)
87	trace_line()
88	check_table_marker()
89	check_kitchen_marker()
90	
91	#⑧メインプログラム（指定のテーブルを見つけたら配膳を行う）#
92	if table_marker_count == table_number and serving == True:
93	motor.set_speed(0,0)
94	sleep(2000)
95	servo.set_angle(0)
96	sleep(2000)
97	servo.set_angle(90)
98	serving = False

92 行目からのコードのインデントが右に一つ下がっていないと、85 行目から 89 行目だけがループされ続

け、92 行目からが反映されず論理エラーになる。このような例が多く頻度で現れれば、教員側の適応力もあがってくるが、最初は時間をかけなければならないこともあると感じた。

6.5. 配膳ロボカスタマイズ編

Python の基礎課題や配膳ロボの課題に関しては、画一的な解答が決められているので、そう難しくはないが、ここから自分で問題を発見し、解決までを行わなければならない。カスタマイズのプログラムはループ評価からもわかるように主に二段階から成り立っている。まずは、いつでも動くようにプログラムを実行させる方法、もう一つは、他のプログラムが実行されているときに、そこに割り込んでプログラムを実行させる方法である。特に割り込みプログラムでは以下のようなコードを書いて割り込ませなければならない。これは、180 度サーボモータを 2 秒おきに 0 度と 90 度を行き来するプログラムである（コード 3）。

コード 3

	略
21	record_time = 0 #起点となる時間を記録する変数
22	TIME_INTERVAL = 2000 #繰り返し動作を行う時間間隔
23	motion_number = 0 #異なる動作を繰り返す際の動作番号
	略
98	while True:
99	#motor.set_speed(50,50)
100	trace_line()
101	check_table_marker()
102	check_kitchen_marker()
103	
104	if running_time() - record_time > TIME_INTERVAL:
105	record_time = running_time()
106	if motion_number == 0: #motion_number フラグが 0 なら
107	servo180.set_angle(0) #サーボモータの角度を 0 度にする。
108	motion_number = 1 #motion_number を 1 に変える。
109	elif motion_number == 1: # motion_number フラグが 1 なら
110	servo180.set_angle(90) #サーボモータの角度を 90 度にする。
111	motion_number = 0 # motion_number を 0 に変える。
112	
113	略

このコードは実行時間を計測して、2000 ミリ（2）秒おきに motion_number のフラグを変えていき、そ

それぞれの動作を記述するというアルゴリズムである。この記述を資料には書いたが、実行できた生徒は2グループほどしかおらず、ほとんどが、割り込みプログラムを書かないコーディングをしていた。また、部品や配膳ロボ个体によっては生徒たちや教員でも直せないエラーも一部発生したりしており、割り込みプログラムは少し敷居が高いと感じた。扱う部品には、多くの動作確認が必要であるとも感じた。

6.6. プレゼンテーション

一人ひとりが皆の前でプレゼンテーションを行うと、授業時間数が超えてしまうのでその時間を削る方法考えた。プログラミングの時間を極力確保するために生徒それぞれが自分のコンピュータの前でプレゼンテーションの動画を取り、ロイロノートに提出という方法をとった。提出後は共有をすることで、気になる友達の発表も生徒間で見るようにした。

次に、プレゼンテーションのループリック評価について言及する。①トップページ、②会社紹介、③社会的背景とデータ、④商品紹介、⑤実演、⑥プログラム、⑦参考文献のスライド要素の中で、③のデータ、グラフなどは、与えられたグラフ以外に自分でデータを持ってきて表現するグループがほとんどであった。売上高のデータ、人件費のデータ、DX戦略を打ち出すデータなどよく世の中のことを考察している点が見られた。また、熱意に関しても、通販番組のようなプレゼンテーションをするグループも現れ、本当に商品を売っているかのようにプレゼンテーションをしていた。ただ、カスタマイズについては緊急事態宣言のためのオンライン授業の影響で時間が足らず、そんなに深くは扱うことができなかった。組み込み系のプログラミングの場合、〇〇しながら△△するというような並列処理ができないため、時間ごとに区切って動作を分ける割り込みプログラムを書かなければいけないが、書いている生徒が少なかったのは反省点である。ここは来年度、資料は改善すべきであると考えている。

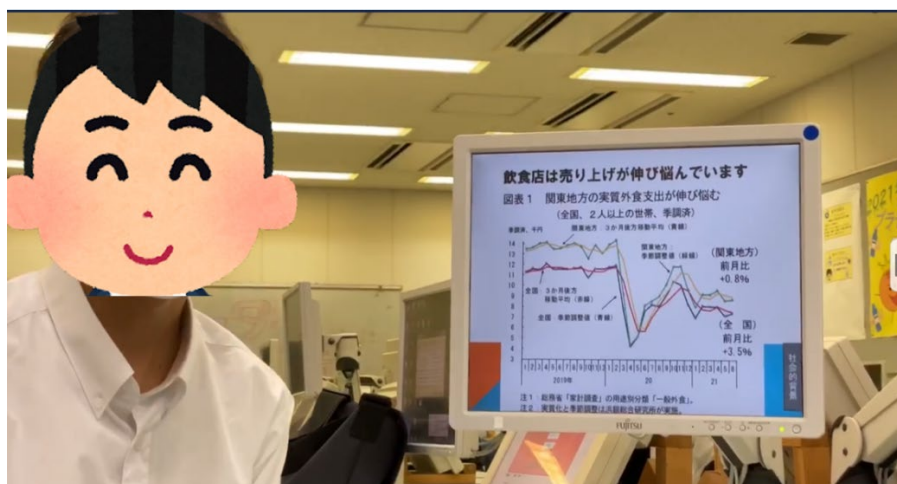


図 16 プレゼンテーションの様子（一部加工）[4]

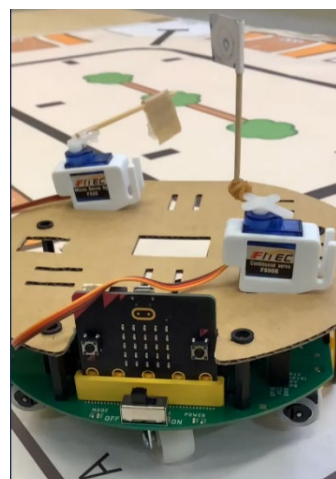
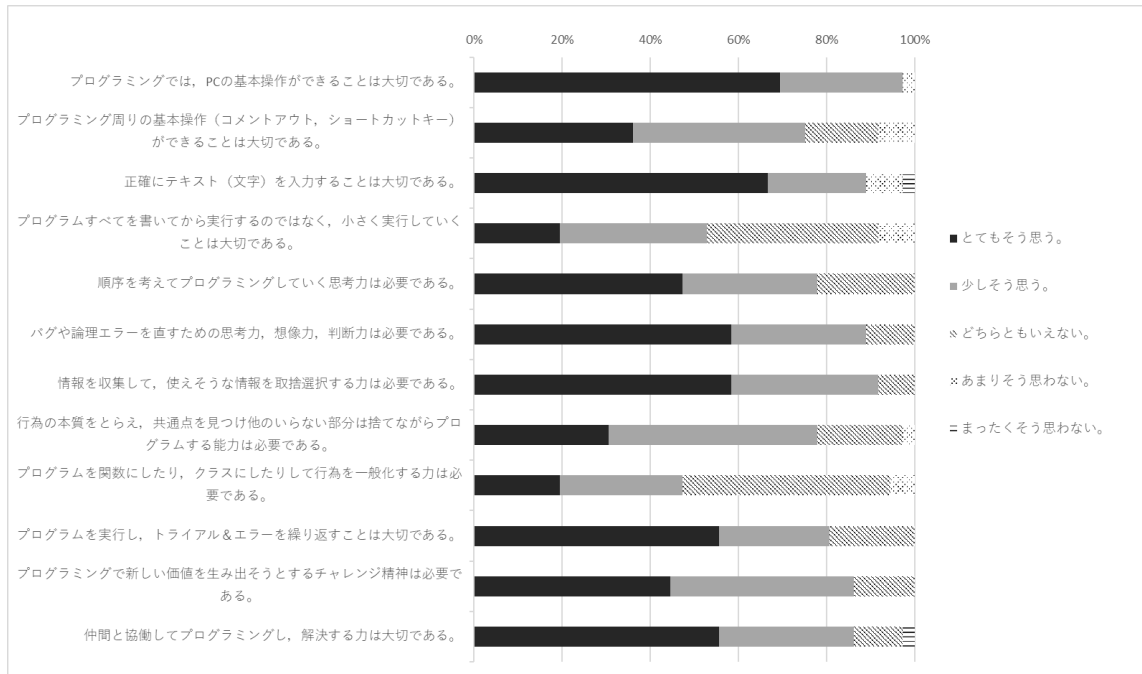


図 17 配膳ロボのカスタマイズ

6.7. 事後アンケート

さて、一通りの学習活動が終わった後、事後アンケートをとった。事後アンケートについては、事前アンケートと同様の質問を行った。結果が以下である(図 18)(表 6)。

事前アンケート（参考）



事後アンケート

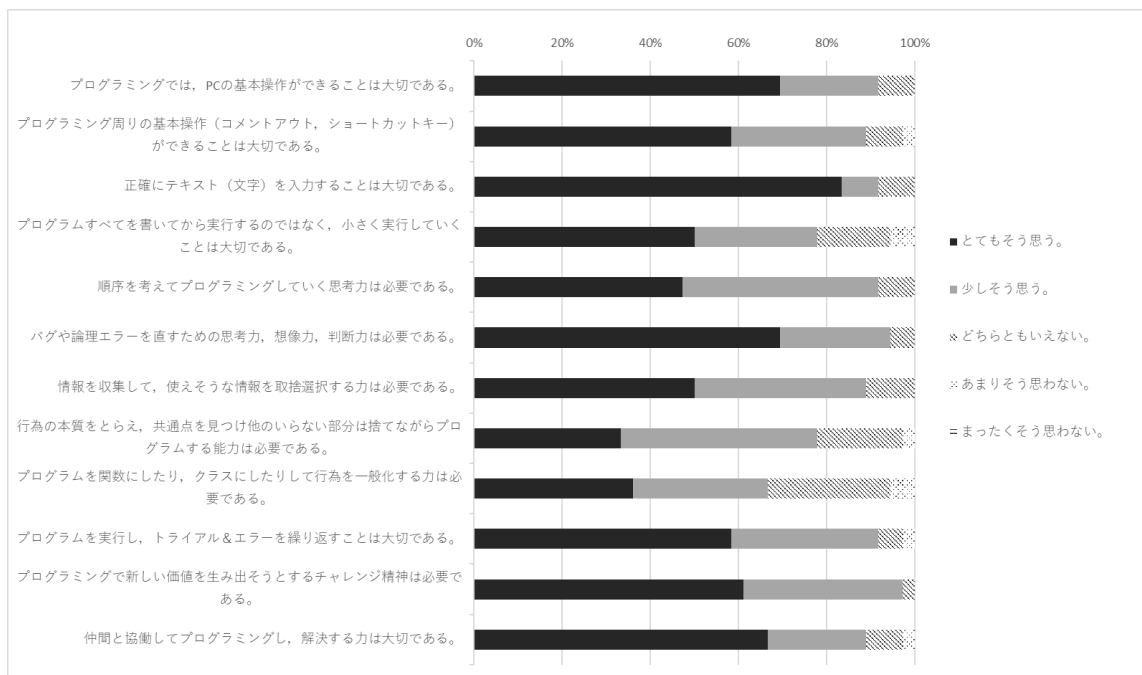


図 18

表 6

	平均値	最頻値	とてもそう思う。 頻度	少しそう思う。 頻度	どちらともいえない。 頻度	あまりそう思わない。 頻度	まったくそう思わない。 頻度
Q1 プログラミングでは、PC の基本操作ができることは大切である。	4.61	5	25	8	3	0	0
Q2 プログラミング周りの基本操作（コメントアウト、ショートカットキー）ができることは大切である。	4.44	5	21	11	3	1	0
Q3 正確にテキスト（文字）を入力することは大切である。	4.75	5	30	3	3	0	0
Q4 プログラムすべてを書いてから実行するのではなく、小さく実行していくことは大切である。	4.22	5	18	10	6	2	0
Q5 順序を考えてプログラミングしていく思考力は必要である。	4.39	5	17	16	3	0	0
Q6 バグや論理エラーを直すための思考力、想像力、判断力は必要である。	4.64	5	25	9	2	0	0
Q7 情報を収集して、使えそうな情報を取捨選択する力は必要である。	4.39	5	18	14	4	0	0
Q8 行為の本質をとらえ、共通点を見つけ他のいらない部分は捨てながらプログラムする能力は必要である。	4.08	4	12	16	7	1	0
Q9 プログラムを関数にしたり、クラスにしたりして行為を一般化する力は必要である。	3.97	5	13	11	10	2	0
Q10 プログラムを実行し、トライアル＆エラーを繰り返すことは大切である。	4.47	5	21	12	2	1	0
Q11 プログラミングで新しい価値を生み出そうとするチャレンジ精神は必要である。	4.58	5	22	13	1	0	0
Q12 仲間と協働してプログラミングし、解決する力は大切である。	4.53	5	24	8	3	1	0

集計をしてみたところ、最頻値は、Q8 以外すべて 5 になった。Q8 は「行為の本質をとらえ、共通点を見つけ他のいらない部分は捨てながらプログラムする能力は必要である。」でプログラミング的思考力の抽象化の内容にあたる。確かに、この内容は習得するのに経験や時間を要する必要がある、またその内容を今回は厚く扱わなかったため、ここで 4 になるのはやむを得ないと考える。ただ、この結果は真摯に捉え、次につなげていこうと考えている。

さて、このアンケートの中で一番注視していたのは、事前アンケートにも記述したとおり、「Q4 プログラムすべてを書いてから実行するのではなく、小さく実行していくことは大切である。」というプログラミング的思考力の「分解」の要素を含んだ質問である。プログラミングの中で分解という考え方がどれくらい浸透しているかを特に注視していた。分解を詳しく見てみると以下のようなグラフになった(図 19)。

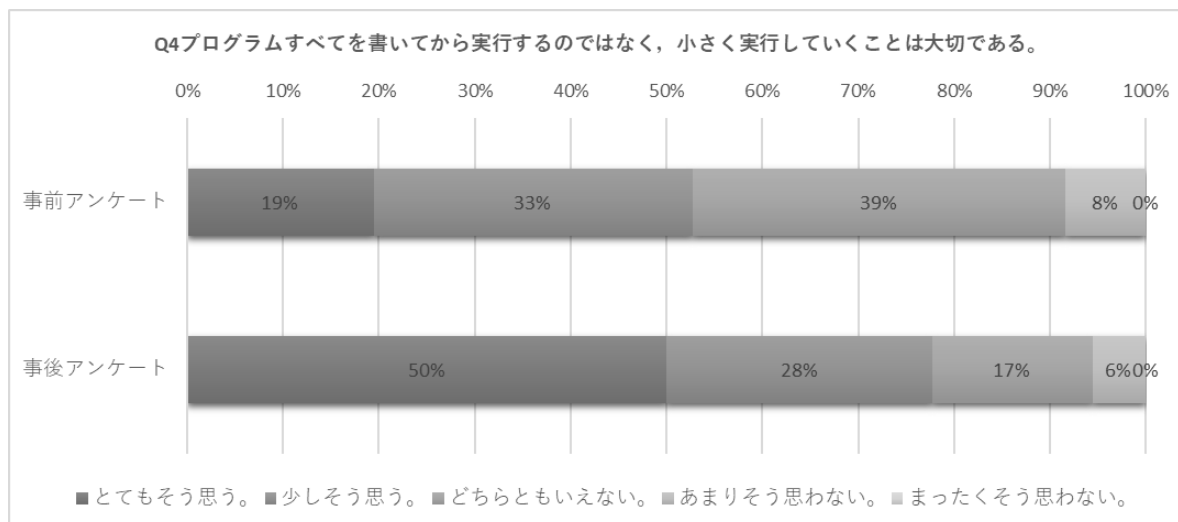


図 19

事後アンケートでは半分の生徒が「分解」の大切さを感じており、「とてもそう思う」の生徒は 50%, 「少しそう思う」も合わせると、78%ほどの生徒が大切さを感じているようだった。分解の考えさえ構築できれば、一人でプログラミングを行うことも可能だと考えるので、今後、この研究にさらに分解の研究を積み重ねて行っていきたいと考える。

また、生徒たちのその他の感想を以下にまとめた。

①Python 基礎課題 No10 までの感想を教えてください。

- ・ ためになりました
- ・ 最初は何がなんだか分からなかったけど徐々に慣れてわかってくると何が違うのかがわかって楽しくできた
- ・ とくに
- ・ 仕組みが部分的には分かった
- ・ 応用がほぼ自力で書くものなので難しかった
- ・ プリント全部はできなかったが、基本はおおむねできた
- ・ 基礎能力を身につけられた
- ・ 理解するのに苦労した。
- ・ エラーなどがたくさん出たがその分たくさん改善できた
- ・ 基礎工程を理解することで配膳ロボットの作業が進んだ。
- ・ エラーのオンパレードだった。
- ・ 所々自分で考えてプログラムをする必要があり、楽しかった。
- ・ 楽

- ・地道だが、合っているかどうか等やっているときはとても楽しめた。
- ・基礎課題で何エラーかによりどういうミスなのか把握できた。
- ・書いただけだったので、あまり工夫ができなかった
- ・難しかったが、考えたりタイピングの力がついたと思い、とても良かった。
- ・個人課題は問題が発見しやすく解決しやすいと思った
- ・難しかったができた
- ・Python の基本操作を学ぶことができてよかった
- ・一人での作業だったので少し大変だった。また、間違えないようにすることよりも間違いを見つけて正しく直すことが難しかった。
- ・コメントアウトを学べたことで、先のことまでプログラミングしてしまっても、コメントアウトをすることで無効化させることができた。
- ・難しかった
- ・だいぶ理解に苦しんだが、結構楽しかった。
- ・効率的に書くのが重要だと思いした。
- ・大変だった
- ・分からないところも先生に聞けばなんとなく分かった
- ・順序を理解することが大切だと思った
- ・大変だった
- ・最初のプログラミングは簡単だったけど、最後の方が難しかった。
- ・楽しかった
- ・パートナーと協力しとても楽しいものができた。
- ・Python の基礎的な高校生向けの教科書がない中でわかりやすいテキストで先生の努力などを感じられた
- ・ロボットを動かすための基本的な単語をたくさん知れた。また、それらの基本的な単語を使えばたくさんのプログラムができることも知った。

②配膳ロボットの良かった点を教えてください。

- ・実際に動くという事があったため楽しむことができた
- ・将来にいかせそう
- ・とくに
- ・分からない
- ・ちゃんと動いた時は感動した
- ・エラーの文字がわかりにくい
- ・ペアで協力して出来た
- ・段階的に考える力がつく
- ・何回も失敗したが最終的に成功できたところ。
- ・マイクロビットの理解をするのにとてもいい題材だった。
- ・試行錯誤が楽しかった

- ・苦勞してプログラムした実行内容が正確に作動して達成感を感じた。
- ・自分もトレースラインできた
- ・僕達でも簡単にカスタマイズできた点。
- ・自分のやりたいことをロボットにやらせられた
- ・自分で改造できた
- ・自分でカスタマイズできてとても面白く、興味深かった
- ・アイデアを出して具現化するまでが楽しい
- ・さまざまなプログラミングが体験できる
- ・Python の応用力を身につけることができて良かった
- ・実際に人を雇わなくても、自動で動いて配膳をしてくれるという便利さ。
- ・自分の好きなようにカスタマイズできる。また、音を鳴らす、光らせる、旗を回すなどの中から選んで自分で工夫してプログラムできるというところがよかった。
- ・スライドと説明がよくできた
- ・楽しかった
- ・プログラミングしたものが動いたこと
- ・色々なことができる
- ・作動するとプログラムが成功していると分かって嬉しい
- ・動いた瞬間が嬉しかった。
- ・変化が目に見える
- ・配膳ができるところ
- ・機能がたくさんあって、たくさんの事が出来た
- ・走る
- ・可愛い
- ・ステップごとに分かれたプログラミングテキストはわかりやすく楽しく段々と配膳ロボがタスクをこなしていく姿は見ていて可愛くて楽しかった
- ・様々なツールがあって楽しかった。ネオピクセルや LCD は楽しかった。

③2 学期のプログラミングの授業を受けての総合的な感想を教えてください。

- ・難しかった
- ・色々壊れているなど問題はあったがそれを直すのも楽しかった
- ・とくに
- ・仕組みが部分的には理解できた
- ・ロボットを自分で作って動かすのが新鮮で楽しかった。
- ・難しく誰かに聞かないとわからないことが多かった
- ・プログラミングの実践的な活用ができて良かった
- ・理解するのが大変だった。
- ・自分はタイピングがあまりできず、打つことが多く難しかったが二学期のプログラミングを通して少

しできるようになった。

- ・プログラミングの理解が深まり、とてもためになる授業だった。
- ・パソコンの基本操作について教えてほしい
- ・グループワークがあって、作業が滞ることもあったが、楽しかった。
- ・ちょっと難しい
- ・やり甲斐が感じられるが、とても大変で地道な作業だった。
- ・プログラミングが苦手の自分にとってとても困難ではあったが、成長を感じられた
- ・難しかったが、自分で工夫ができて楽しかった
- ・とても楽しかった。家でもやりたいと思った
- ・初めは原理を理解するのに時間がかかったが、取り組む中で理解できた
- ・色々できるが配膳ロボット自体が壊れたりするので難しかった
- ・思考力や行動力を身につけることができた
- ・一人の力でプログラミングを完成させられる能力が少しついたと思うのでよかったし、今回得た知識はこれからにも繋げていきたいと思った。
- ・自分が入力したプログラムがどのような役割を果たしているのかを理解することができた。また、コメントアウトやショートカットキーを使うことでスムーズにプログラミングを進めることができた。
- ・少し難しかったけど興味がある
- ・新たな体験ができて楽しかった
- ・たのしかったしコピーペーストの重要性を学べてよかった。
- ・色々な技術を学べて楽しかった
- ・タイピングが遅いと作業効率が下がる
- ・計画性が大切
- ・エラーを直すのが大変だった
- ・プログラミングは難しいものだと思っていたが、やってみるととても面白かった。
- ・プログラミングをもっと身につけていきたいです。
- ・結構楽しい
- ・とても楽しかったです。
- ・基礎的なプログラミングをロボットの動きの成長とともに学べて楽しかった
- ・ロボットを動かす仕組みが知れた。複雑なプログラミングではあるが、意外と簡単にロボットを動かせるのだなと思った。

このように見ると個人個人捉え方は違うものの、おおむね満足した様子であった。また、「分解」の考え方を身につけていそうな生徒の発言があったのはこちらの意図する授業ができたと考えている。

7. まとめ

今回の授業研究をするにあたり、学内の他教科の教員からもアドバイスを頂いた。アドバイスを頂く過程の中で、筆者が気づいたことは、プログラミングについて、まだ学習指導要領が曖昧な状態であるということである。例えば他教科の数学科では、関数の中に、一次関数、二次関数、三角関数、三次関数など様々な単元があるが、情報科のプログラミングはまだ「プログラミング」でしかない。プログラミングの中にも、線形探索、二分探索、ハッシュ探索、モンテカルロ法、ニュートン法など様々な代表的なアルゴリズムが存在する。これらのアルゴリズムを体系化し、学習する順序も決定してくるようになれば、プログラミングのアルゴリズムは高校分野でも大きく学問として成り立つのではないかと考える。

次にプログラミングの指導方法だが、プログラムを書けることを目的とした場合、タイピングやショートカットキーといったコンピュタリテラシーを早めに身につけておくことが学習前段階では必要になると考える。そして、プログラミングの授業全体の設計として重要なのは、エラーへの嫌悪感をいち早く取り除き、エラーの見方と直し方に関する知識や操作方法を早めに身につけておくことではないかと考えている。あとは「分解」に関する考え方を教材の中に盛り込んでいけばプログラムは書けるようになると思う。そして、プログラミングの難易度だが、筆者個人としては、高校1年生でそう高くは求めてはいけないうと考えている。まずは、楽しい、面白いということを中心に学習させ、モチベーションを高めていく。おそらく、多くの学校は高校1年時や2年時で「情報Ⅰ」を扱い、「情報Ⅱ」や「情報演習」を共通テスト対策として、高校3年生で設置すると思われる。高校3年生になって共通テストに必要な場合、難易度を上げて代表的なアルゴリズムを学んでいけばよいと筆者は考えている。ただ、プログラムを書けるようになることを目的とした場合は、このような方法をとるが、共通テストなどで点数を取らせたい場合、読めることに特化した指導方法にするというまた違う方法があるかもしれない。どちらがよいかは一概にはいえないが、そのようなことも考えていかなければならない。

以上のように述べてきたが、色々と一番頭を悩ませるのは、やはり授業の指導に関する時間配分である。本校の場合、今年度はタイピング学習時間が少し長かったので来年度は早めに教科書範囲に入り、もう少し教科書の内容の時間に余裕をもって確保できるようにする予定であるが、それでも時間は足りなくなると考えられる。時間がない中で全単元を消化するポイントは、やはり複数の単元を同時に扱っていくということではないだろうか。教科書の内容を行いながら情報デザインの内容を扱ったり、ネットワークを行いながらセキュリティを扱ったり、プログラミングを行いながら問題解決も扱っていくということである。

配膳ロボのプリントに関しては、まだお見せできるレベルにないと感じたので、掲載は割愛した。それでもこちらをご覧になりたい方は、佼成学園の岡野までご連絡いただければ、PDF ファイルで送付する。そして、全単元を細かく最後まで指導はしきれていないところに問題があるので、今後もこれらを改善する授業研究は続けていく。

8. 謝辞

今回、研究をするにあたり、多くの方々にお世話になった。学校長、教頭をはじめとする本校の教員、情報科の萩原先生、そして、株式会社スイッチエデュケーションの宗村様、木原様にこの場を借りて感謝申し上げます。

9. 参考文献

9.1. 引用文献

- [1] 高等学校学習指導要領(平成 30 年告示)解説 総則編,文部科学省
- [2] 高等学校学習指導要領(平成 30 年告示)解説 情報編,文部科学省
- [3]Consolidation of human skill linked to waking hippocampo-neocortical replay
ER Buch,Cell Reports Volume 35, Issue 10, 8 June 2021
- [4]いらすとや（高校生・中学生のイラスト）
https://www.irasutoya.com/2013/05/blog-post_6973.html（2021 年 12 月 17 日参照）

9.2. 参考文献

- [5]BBC micro:bit MicroPython documentation（English）
<https://microbit-micropython.readthedocs.io/en/v2-docs/>
- [6]BBC micro:bit MicroPython ドキュメンテーション（日本語版）
<https://microbit-micropython.readthedocs.io/ja/latest/>