

コンピュータとプログラミングにおける

思考力・判断力・表現力の指導方法と評価についての研究

ーゲームを制作するための箇条書きアルゴリズムー

1 単元や課題の設定理由・ねらい

共通教科情報科の目標の一つに「様々な事象を情報とその結び付きとして捉え、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する力を養う」と記載されている。プログラミングの分野では問題の発見・解決に向けて、試行錯誤と振り返り及び改善を行い、情報と情報技術を適切かつ効果的に活用する力を養う必要がある。

本研究では、ゲームを制作する際、コンピュータで情報が処理される仕組みを理解した上で、その目的に応じたアルゴリズムを、箇条書きで記述する授業の指導方法と評価について研究する。

2 研究内容

(1) 単元の目標

目的に応じたアルゴリズムを考え適切な方法で表現し、プログラミングを行う中でその過程を評価し、改善することで、コンピュータを効率よく活用するためにアルゴリズムを正しく表現する力やアルゴリズムの効率を考える力を養う。

(2) 学習活動に即した評価規準（「思考・判断・表現」の観点のみ）

分岐、反復構造及び乱数の要素を含んだ設計図を作成し、目的に応じたアルゴリズムを箇条書きで合理的・効率的に表現している。

(3) パフォーマンス課題及びその概要

ア パフォーマンス課題

分岐、反復構造及び乱数の要素を含んだ簡単なゲームを作ってみよう。

イ 授業の進め方

(ア) じゃんけんのアルゴリズムを、分岐、反復構造に着目し、コンピュータで情報が処理される仕組みを理解する。

(イ) アルゴリズムの表現方法に関する演習を行う。

(ウ) 各自の目的に合わせたゲーム制作に取り組む。

ウ 事前学習について

- ・ 順次、分岐、反復構造、変数について
- ・ アルゴリズムの表現方法について
- ・ プログラミング言語（ソフトウェアの操作方法）について

時 限	学習活動	指導上の留意点
1	○実習1 (30分) ・サイコロゲームを題材としたプログラムの設計図や配慮した点を記述し、プログラムを作成する。 ○自作ゲームのテーマ決定 (5分) ・自作のゲームを作るために、提示された条件をよく理解し、テーマを決定する。	・パフォーマンス課題に取り組むために、設計図やプログラミングの際の工夫や合理的に動かすために配慮した点を、必ず記述するように促す。 ・ループリックに触れておく。 ・例題を利用することも可能とする。
2	○実習2 (45分) ・自作ゲームのテーマやストーリー、提示された条件(分岐、反復構造及び乱数)を考えて設計図を書き、プログラムを作成する。 ・プログラムを作成する過程で、要素の使い方や動作の工夫など、合理的・効率的に動かすために配慮した点を記入する。	・条件(分岐、反復構造及び乱数)が設計図の中に含まれているか、確認するよう促す。 ・プログラムを作成する過程で、設計図や工夫した点などの記述に追加、変更がある場合は、必ずワークシートに反映するよう指示する。
	○まとめ (5分) ・実習の振り返りと自己評価を行う。	

(6) 評価の進め方 (評価方法)

「ゲームで行いたいこと(目的)」に合わせて、分岐、反復構造及び乱数を含むアルゴリズムを簡条書きで作成し、さらに、合理的・効率的に動かすために配慮した点が、ワークシートに記述されているかという点で評価した。

3 授業の状況

(1) 指導するに当たって、学校の状況に応じて留意したことやその理由

プログラミング言語による実習は、言語の習得が主となってしまふことが多く、学校によって使用する言語も異なる。アルゴリズムをフローチャートではなく、文章(簡条書き)の形にすることで、フローチャートのルールについて、時間をかけることなく、自分の考えを表現しやすいと考えた。分岐、反復構造及び乱数について分解して考えることに重点を置き、簡条書きでアルゴリズムを記述することで、生徒が実習に取り組みやすくなっていることを実感した。

(2) 授業実践後に協議して設定したルーブリックと典型的な作品例

達成度	観点の説明	生徒の作品例
<p>A (十分満足できる状況)</p>	<p>「要素の使い方や動作について工夫した点、合理的に動かすために配慮した点」の項目が記述でき、それが設計図にも反映されている。さらに、分岐、反復構造及び乱数の要素を含んだ設計図を作成し、目的に応じたアルゴリズムを簡条書きで作ることができる。</p>	<div data-bbox="719 349 1353 1077" style="border: 1px solid black; padding: 5px;"> <p>プログラミングの設計図(記入内容) ※本番用</p> <p>ゲーム名 ※ストーリーorゲームの要素や大まかな流れ</p> <p style="text-align: center;">敵討伐</p> <p>自分とCPがランダム1~10の数字をだし、 自分CPの数字が同じたらず、その数字を得意とし、10点1/10まで 2/10まで</p> <p>プログラムのための設計図(要素や動きの関係を記す)</p> <p>変数: <u>Score</u> (総得点), <u>My number</u> (自分の数), <u>CP number</u> (CPの数)</p> <p>手順</p> <ul style="list-style-type: none"> Scoreと0と弱 自分の数を決める(My number: 1~10の整数を乱数で) CPの数を決める(CP number: 1~10の整数を乱数で) 自分の数とCPの数同じ(My number = CP number) <p>★ Yes → <u>Score</u>: score + My number 実行</p> <p>★ No → <u>Score</u>: score + 0 実行</p> <p>要素の使い方や動作について工夫した点、合理的に動かすために配慮した点</p> <p>ただ敵と自分の数字が同じだと、それと得点と弱さで、5/10-3/10 1/10-2/10と弱さの差がある。下手、敵が同じな違うかというこに1/10に 5/10弱さ強さの差が弱くなる。</p> </div> <p>[評価に対する注釈]</p> <ul style="list-style-type: none"> 工夫した点や合理的に動かすために配慮した点が、設計図に一つでも記述してある。ただし、ゲームが成り立つかどうかは問わない。
<p>B (おおむね満足できる状況)</p>	<p>分岐、反復構造及び乱数の要素を含んだ設計図を作成し、目的に応じたアルゴリズムを簡条書きで作ることができる。</p>	<div data-bbox="699 1283 1369 1821" style="border: 1px solid black; padding: 5px;"> <p>ゲーム名 【 1/10 と強 】 のストーリーorゲームの要素や大まかな流れ</p> <p>プログラムの設計図(要素や動きの関係を記す)</p> <p>MyHPの CPHPの 0以下に なるとで くりかえす</p> <p>CPHPをかわす MyHP = 10 ランダムに My kazu を決める ランダムに CP kazu を決める もし My kazu < CP kazu ならば ★ Yes → 「自分と相手に1ダメージ」を表示 MyHP - 1, CPHP - 1 する もしそうではなく、My kazu < CP kazu ★ Yes → 「1ダメージをくらった3ダメージ」を表示 MyHP = 3 する もしそうではなく、My kazu > CP kazu ★ Yes → 「1ダメージをくらった相手に3ダメージ」を表示、CPHP - 3 する</p> <p>要素の使い方や動作について工夫した点、合理的に動かすために配慮した点</p> <p>CPHPをかわすにして、弱さ強さを強弱で できるからして。</p> </div> <p>[評価に対する注釈]</p> <ul style="list-style-type: none"> 例題を参考にして、自分の考えるゲームの設計図を作成しているが、工夫した点、配慮した点が分岐、反復構造及び乱数の要素に関する部分ではない。

(3) 「C (努力を要する状況)」と評価した生徒への指導の手だて

例題の数値を変更しただけの生徒には、具体的な派生について助言する。例えば、「サイコロの数を増やしたらどうなるか」「トランプに変更したらどう変わるのか」などの質問をして支援する。

要素(分岐、反復構造及び乱数)が足りない生徒には、ループリックを再度確認するよう促し、不足している要素は何かを考えたり、使い方を復習したりするよう促す。

4 まとめ及び考察

(1) 実習課題について(生徒の取組状況も含めて)

例題を提示することで、多くの生徒が必要な要素(分岐、反復構造及び乱数)を取り入れて、例題からどのような派生が考えられるか、試行錯誤する様子が見られた。オリジナルゲームの作成という身近な題材を用いたことや、実際のプログラミング言語を使うわけではなく、文章を用いてアルゴリズムを表現できたため、生徒の取組もよかった。

事前にゲームを考えていた生徒の中に、設計図や工夫した点について箇条書きで記述できていたものの、「工夫した点、合理的に動かすために配慮した点」の項目に記述されていなかった生徒がいた。逆に、「工夫した点、合理的に動かすために配慮した点」の項目に記述してあるものの、設計図に反映されていなかったり、設計図が完成しなかったりする生徒もいた。机間支援において、「どういったところを工夫したのか」などの声かけが適宜、必要であると感じた。

(2) 評価について

当初は、ループリックの達成度Aを「分岐、反復構造及び乱数の要素を含んだ設計図を作ることができる。要素の使い方や動作の工夫など、合理的・効率的に動かすために配慮がされている」としていたが、このループリックでは、設計図のみを重視し、「要素の使い方や動作について工夫した点、合理的に動かすために配慮した点」の項目の記載が不十分でも、評価Aとなる事象が発生していた。また、設計図のみでは評価しきれない、生徒の創意工夫や思考の過程について、評価に反映させることができない場面もあった。そのため、ループリックを「『要素の使い方や動作について工夫した点、合理的に動かすために配慮した点』の項目が記述でき、それが設計図にも反映されている。さらに、分岐、反復構造及び乱数の要素を含んだ設計図を作成し、目的に応じたアルゴリズムを箇条書きで作ることができる」と変更し、工夫した点や合理的に動かすために配慮した点の記載を受けて設計図が書かれているかどうか判断するループリックへと見直すこととなった。こうしたパフォーマンス課題に取り組む際には、設計図だけでなく、生徒の思考の過程が視覚化できるよう、目的に対してどのような工夫や配慮をしたかについて記述し、それが設計図に反映されているかどうか、生徒に声かけして確認するよう促していく必要がある。

評価を進めていく中で、ごく少数ではあるが例題の繰り返し回数などの変更のみしか行っていない生徒も見られた。そのため、達成度Bの注釈としてループリックに加えた。

(3) 授業実践の改善に向けて

本研究ではアルゴリズムの表記方法を箇条書きにするような形式で行った。しかし、フローチャートを授業できちんと行っている学校では、フローチャートで設計図の記述を行うとよい。箇条書きであると生徒は記述しやすいが、評価する上で教員側が設計図を読み取りにくいと感じるものもある。

今回のパフォーマンス課題では、「要素の使い方や動作について工夫した点、合理的に動かすために配慮した点」が設計図のどの部分に当たるのか分かりにくく評価しづらかった。また、生徒も工夫した点や配慮した点を明確に示すことができていなかった。「工夫した点、合理的に動かすために配慮した点」が設計図のどの部分に当たるのか、明記させる必要があった。紙媒体なので、色を変えて線や矢印でつなぐ等の対応をするとよい。

今回のパフォーマンス課題では、記述した内容をプログラムとして実行することまでは行っていない学校が多い。時間に余裕があれば、この設計図を基にプログラミング言語を用いて、実際にプログラムを作成することで、試行錯誤を繰り返し、アルゴリズムの合理的な考え方や自分のアルゴリズムの改善が図れると感じる。

(4) その他

今回の授業は、分岐構造、反復構造、変数について理解していれば実施することができる。そのため、1クラスは、分岐構造、反復構造、変数まで授業で学習した上で実施し、他のクラスでは、配列や関数まで学習した後に実施した。設計図の中身を比べたところ、配列や関数を用いて設計図を作成した生徒は1名で、ほとんど差は見られなかった。

今回のパフォーマンス課題では、記述した内容をプログラムとして作成して実行するところまでは行っていない。しかし、時間的に余裕のあるクラスではプログラムの作成まで行ったところ、そのクラスの方が、設計図の見直しや「工夫した点や合理的に動かすために配慮した点」をしっかりと記述している様子が見られた。今後は、プログラムの作成までを一つのタームとした授業展開を考えていきたい。

研究後に、実際にプログラミングまで行ったクラスでは、他者と情報共有を行ったり、情報検索を行って、試行錯誤したりしている生徒がいた。今回の研究で実施することはできなかったが、プログラムを作成し完成させた後に「工夫した点や合理的に動かすために配慮した点」を再度記入すると、同じルーブリックで評価したとしても、達成度A（十分な満足できる状況）となる生徒が格段に多くなるのではないかと考えられる。