４－３　プログラミング教育　研究のまとめ（平成３０年度）

１　事業概要

プログラミング教育を取り入れた複数教科・領域における教材の活用と指導のあり方について検証する。

２　研究の成果

２－１　効果的な教材を活用した研究事例

（１）効果的な教材の活用実践

今年度、センサーを搭載したロボットを５台配備した。車軸とタイヤがついたものでセンサーカーとも呼ばれる。このセンサーカーに、センサー類（ＬＥＤ赤、ＬＥＤ青、赤外線センサー、タッチセンサー、ブザー）を必要に応じて搭載させ、その機能に合わせたプログラムを作った。

まず、直進、後進、右折、左折、回転、停止の命令を組み合わせ、センサーカーを思い通りに動かした。さらに、タッチセンサーを組み合わせることで、壁に当たったときにバックし、向きを変えて直進させる動きにより、部屋の中を永久に走り続けるようなプログラムを作った。

また、赤外線センサーとＬＥＤを組み合わせ、防犯センサーを作ったり、ＬＥＤ（赤、青）とタッチセンサーを組み合わせ、ボタン式歩行者用信号機を作ったりした。

（２）成果と課題

センサーカーは、各種センサーを取りつけることにより、応用範囲が広がるという利点がある。ＬＥＤを用いた信号機のプログラミングでは、自動車用や歩行者用のプログラムを作った。そして、歩行者用の場合、押しボタン式や赤外線反応式といった応用もできた。信号機の他に、防犯ライトや衝突回避システムも作ることができた。このセンサーカーは、子どもの好奇心を揺さぶり、楽しくプログラミングを学ぶことができる教材であった。この教材の活用により、身の回りにある機械の仕組みを模擬体験することができ、プログラミングが身近なものであることを体感することができた。

課題は、一人につき一台ではないので、数人のグループでの作業となってしまうことである。特定の子どもだけが扱うのではなく、操作をする子ども、計画書を持ちプログラムを確認する子ども、センサーカーの動きを確認する子どもなどの係を決め、ローテーションで全ての作業ができるように役割交替するなどの配慮をする必要がある。また、ＬＥＤの点滅やブザーの音量、センサーカーのスピードを一定の範囲内に指定しないと健康被害や衝突による破損の原因にもなってしまう。事前の指示の吟味や予備実験をしておくことが大切である。さらに、授業のときは、故障やパソコンのフリーズの対応に備え、複数の教員で指導にあたる必要がある。

２－２　協働的な学びにつながる研究

（１）各教科学習における研究事例

ア　４年外国語活動「Hour of Code（アワーオブコード）」

アワーオブコード内の「モアナと伝説の海」を使ってプログラムを作った。

１９のステージがあり、どのステージからでもチャレンジできるのが利点である。

筏や主人公を指示通りに動かすプログラムを作る。最初は「move forward」「turn left」「turn right」だけだが、「repeat * times」「strike」「dodge」など、単語が増えてくる。

読み方や意味がわかるような指示カードを掲示し、英語への抵抗感を減らしながら進めていった。

スクラッチのようにブロックを並べていくだけなので、手順さえまちがえなければステージをクリアできるため、低学年でも取り組み可能な教材である。

イ　６年理科　センサーカーの利用

センサーカーのＬＥＤ（赤色のライト）を制御（点灯、消灯、点滅）するプログラムを作った。点滅は、点灯・消灯の間隔を決めるためにグループで話し合いながら試行を繰り返し、適切な時間設定を見つけていった。

応用として、赤外線センサーを取りつけ、防犯ライトになるようにプログラミングした。物体が一定の距離まで近づくとＬＥＤが点灯し、数秒後に消灯するのである。

プログラムが複雑になるので、全員で正しいフローチャートを作り、手順を確認した後にプログラムを作った。

（２）特別活動、総合的な学習での研究事例（対話型ロボット「Pepper（ペッパー）」をプログラミング）

ア　特別活動（クラブ活動）

クラブ活動で、「老人介護に役立つペッパー」のプログラムを作った。お年寄りに昔を思い出してもらうことと運動をいっしょにやってもらうことで、脳と体を活性化させるのがねらいである。

利用者であるお年寄りがペッパーの胸のディスプレイから昔の年代や動作の大小を選び、それらがランダムに発動するようなプログラムを作った。

昭和３０年頃の話題を話す、昭和５０年頃の話題を話す、大きな動作をまねしてもらう、小さな動作をまねしてもらう、といった四つの内容について、グループで内容を検討し、プログラムを作った。グループ内で意見を出し合い、お年寄りに笑顔になってもらうことをめざして、手順や内容を考えることができた。

実際のプログラミングの場面では、これまでクラブで学習した内容（話す、動く、選択、条件分岐、繰り返し、ランダム）を思い出しながら、新しい作品を工夫して作る姿が見られた。

イ　３年ほんざかタイム（総合的な学習の時間）

来校者を玄関から職員室へ案内するプログラム「おもてなしペッパー」を作った。

二人一組でミニボードにフローチャートを作り、他のグループと意見交換をして、来校者をスムーズにもてなすような流れになるように改善していった。その後、プログラムを作り、ペッパーを実際に動かし、動作の修正をした。

（３）成果と課題

アワーオブコード内の「モアナと伝説の海」は、１９のステージがあり、どこからでもチャレンジできるのが利点である。説明もプログラムの命令も英語なので、英語の学習とプログラミングの学習が同時にできた。このソフトウェアは、スクラッチのようなブロックを並べていくだけのビジュアルプログラミングソフトなので、日本語バーションに切り替えれば、低学年でもプログラミングの学習が可能である。

ステージがクリアできたとしても、それがよいプログラムとは限らないので、簡潔なプログラムにするようにグループで検討したことで、協働的な学びを推進することができた。

そして、センサーカーの制御は、実物が目の前にあり、プログラムの実行による動作確認が瞬時にできるので、興味を持続させながら学習をすることができた。センサーカーは５台あり、４人１組での実践になった。今後、各自の学習効率を上げたり、プログラミングの個性化を進めたりするために、一人一人がプログラミングに関わっていくことが必要である。

クラブ活動の「老人介護に役立つペッパー」のプログラミングでは、フローチャートを作ることで、自然な流れになるように議論する場ができた。繰り返しの部分や分岐した後の動き、乱数を発生させた後の動きなどをグループごとに作り、最後につなぎ合わせた。フローチャートの手順を追うことで、プログラムのうまく動作しなかった部分を発見するのができた。これにより、プログラミングはフローチャートの思考の流れが重要になることを実感させることができた。課題として、プログラミングの際、児童のプログラミング力に合ったコマンド（命令）の意味や使用例を示し、動きの確認をしておく必要がある。

３年生の「おもてなしペッパー」は、フローチャートをみんなで練り合うことに時間をかけた。実際の動きを確かめながらプログラムを作り、デバッグ作業をし、移動距離や待機時間のパラメーターを適切なものにしていった。会話の文章をキーボード入力ではなく音声入力で行ったため、時間を短縮することができた。ただ、ペッパーの機能面で問題が生じた。ペッパーのセンサーが１ｍ以内にある物に反応してしまい、狭い廊下では動かなくなってしまったのである。本来なら、玄関から職員室まで動くようにプログラムを作ったのだが、実際にはうまく作動せず、ロボットの限界や苦手部分に気づくことができた。その後、広い体育館でのデモンストレーションを行った。

総合的に見ると、これらの実践で、フローチャートやプログラムを作り、ロボットを思い通りに動かす活動により、プログラミング的思考を高めることができた。

２－３　プログラミング教育の指導

（１）子どもの論理的思考の変容について

ただプログラムを作らせるのではなく、次のようなプログラミングの手順を意識させた。

①目標や条件を理解する

②手順をフローチャートに表す

③プログラムを作る

④プログラムを実行し、うまくいかなかった部分を修正する

⑤より簡潔なプログラムにできるか検討する

このような手順の習得は、各教科の学習や日常生活にも生かされた。例えば、算数の問題を解くときも、先の手順と同様なものになっているからである。

①問題を理解する

②問題を解く方法を考える

③その方法で問題を解く

④答えを検討する

⑤よりよい方法がないか検討する

また、家庭科の調理実習で効率のよい手順を考えたり、理科の実験がスムーズにできたりするようになった。

このように、プログラミング教育は正しい手順の構築の練習につながり、論理的思考力を高めることができた。

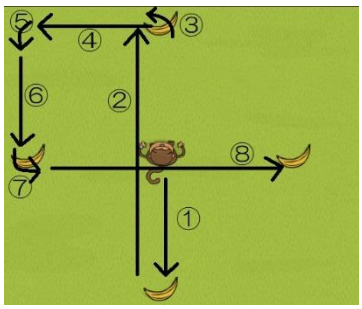
（2）効果的な指導方法に関する研究提案

プログラミングをする際にまず必要なことは、到達点を明確にして目標を定め、条件を確認し、正しい手順を考えることである。フローチャートを作り、流れを確認する。次にそれをプログラムにしていく。作ったプログラムを実行し、思った通りの動きができたか確認する。できていなかったときは、プログラムを修正する。

これがプログラミングの一連の流れであるが、よりスムーズに学習を進めるために、ビジュアル化した指導が効果的である。

例えば、コードモンキーの場合、学習するステージの状況を紙に印刷し、その紙面上で、印刷したモンキーを動かして、プログラムを実行する前に確認していく。そして、プログラムを実行し、思った通りの動きになったか確認する。うまくできない児童もいるだろうが、友達と協力して学習させることで、理解を深めることができる。

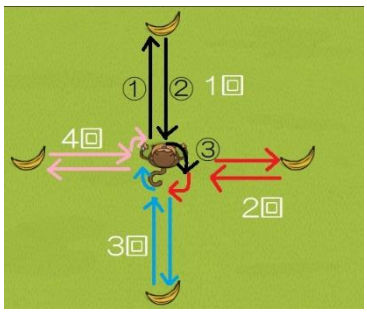
ただ、プログラムが正しく実行できたとしても、不必要な記述がある場合もあるため、精選されたプログラムに直すことも大切にしたい。




```
step -10
step 20
turn left
step 10
turn left
step 9
turn left
step 20
```

例えば、左図のように、バナナを４つゲットする動きをさせるプログラムを８行で作ったとする。

下図のようにすれば、４行で済むのである。




```
4.times ->
    step 10
    step -10
    turn right
```

算数の問題を解いた後に見直しをするように、プログラミングの際も見直しをすることが大切である。できるだけ短い単純なコードでプログラムを作る習慣づけが欠かせない。

複雑で記述が長くなるプログラムについては、すべてをプログラミングするのではなく、プログラムの一部分を取り出し、その部分の手順を考えさせたり、修正させたりするのも有効である。例えば、自動販売機のプログラミングでは、投入するコインの種類を増やした場合や缶が売り切れになった場合のみを考えることが児童の発達段階には有効であると考える。

今後は、児童の実態に合わせた柔軟な学習展開について研究を深めていきたい。