第２章　プログラミングの実際

５　リモートコントロールに挑戦　～「ポケコン搬送車」と散歩しよう～

問題　次のルール（操作方法）に合う「ポケコン搬送車」の動きを考えよう。

答えは，10 進数で書くこと。ヒント：remote を実行してみよう

①「前進」　→　4

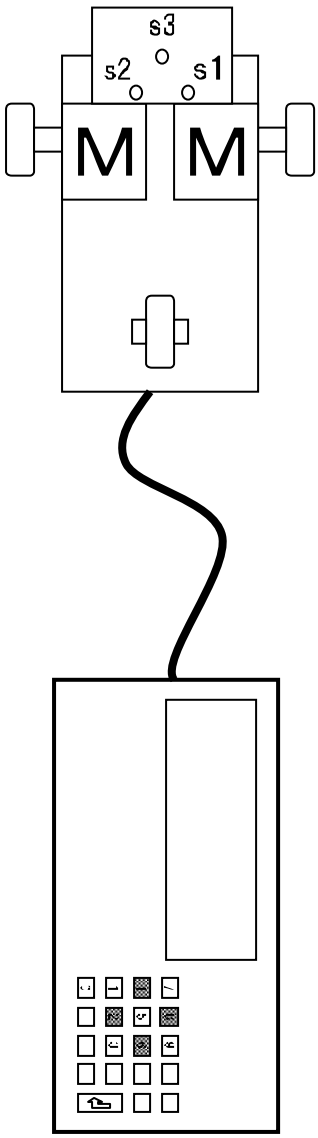
②「左旋回」→2　　8←③「右旋回」

6　←④「後進」

⑤その他のキー　・・・　「ブレーキ」

↓

※何も押されていないときも含む



ルール①　→　４が入力されたら「前進」

左モータ（　正転　）／右モータ（　正転　）

↓

モータ出力　＝　ＯＵＴ　＿＿＿＿

ルール②　→　２が入力されたら「左旋回」

左モータ（ブレーキ）／右モータ（　正転　）

↓

モータ出力　＝　ＯＵＴ　＿＿＿＿

ルール③　→　８が入力されたら「右旋回」

左モータ（　正転　）／右モータ（ブレーキ）

↓

モータ出力　＝　ＯＵＴ　＿＿＿＿

ルール④　→　６が入力されたら「後進」

左モータ（　逆転　）／右モータ（　逆転　）

↓

モータ出力　＝　ＯＵＴ　＿＿＿＿

ルール⑤　→　その他のキーでは「ブレーキ」

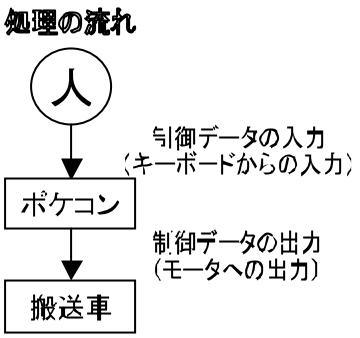
左モータ（ブレーキ）／右モータ（ブレーキ）

↓

モータ出力　＝　ＯＵＴ　＿＿＿＿

★「リモートコントロール」するプログラムを作ろう（ＢＡＳＩＣによる制御）

＜プログラムの考え方＞



＜プログラム化（その１）＞

① 手順１ キー入力 ⇨ 入力待ちにならないキー入力

手順２ 前進の定義 ⇨ ＩＦ文（もし，4 だったら）「前進」

手順３ 手順１へ戻る

または，

② 手順１ キー入力 ⇨ 入力待ちにならないキー入力

手順２ 左旋回の定義 ⇨ ＩＦ文（もし，2 だったら）「左旋回」

手順３ 手順１へ戻る

または，

③ 手順１ キー入力 ⇨ 入力待ちにならないキー入力

手順２ 右旋回の定義 ⇨ ＩＦ文（もし，8 だったら）「右旋回」

手順３ 手順１へ戻る

または，

④ 手順１ キー入力 ⇨ 入力待ちにならないキー入力

手順２ 後進の定義 ⇨ ＩＦ文（もし，6 だったら）「後進」

手順３ 手順１へ戻る

または,

⑤　**手順１**　キー入力　⇨　**入力待ちにならないキー入力**

　　手順２　その他のキー　⇨　**「ブレーキ」**

　　手順３　**手順１**へ戻る

①が「前進」，②が「左旋回」，③が「右旋回」，④が「後進」の定義である。そして，⑤の「ブレーキ」が，「その他のキーが押されたとき」の定義であり，また同時に，「何も押されていないとき」の定義でもある。(重要)

①～⑤の処理を，一つにまとめると，以下のようになる。

＜プログラム化（その２）＞

手順１　キー入力　⇨　**入力待ちにならないキー入力**

①　**手順２**　前進の定義　⇨　**ＩＦ文**（もし，4 だったら）**「前進」**

　　手順３　**手順１**へ戻る

②　**手順２**　左旋回の定義　⇨　**ＩＦ文**（もし，2 だったら）**「左旋回」**

　　手順３　**手順１**へ戻る

③　**手順２**　右旋回の定義　⇨　**ＩＦ文**（もし，8 だったら）**「右旋回」**

　　手順３　**手順１**へ戻る

④　**手順２**　後進の定義　⇨　**ＩＦ文**（もし，6 だったら）**「後進」**

　　手順３　**手順１**へ戻る

⑤　**手順２**　その他のキー　⇨　**「ブレーキ」**

　　手順３　**手順１**へ戻る

※キー入力により，手順２のどれか一つ（①～⑤）が選択されます。

問題 プログラムを作ってみよう。ヒント：remote を実行してみよう

手順１　　10　X　＝　ＡＳＣ（ＩＮＫＥＹ$）－ 48

① 手順２　　20　ＩＦ　X＝ 4　ＴＨＥＮ　＿＿＿＿＿＿＿＿

手順３（20 行目に続く）　　：　ＧＯＴＯ　10

② 手順２　　30　ＩＦ　X＝ 2　ＴＨＥＮ　＿＿＿＿＿＿＿＿

手順３（30 行目に続く）　　：　ＧＯＴＯ　10

③ 手順２　　40　ＩＦ　X＝ 8　ＴＨＥＮ　＿＿＿＿＿＿＿＿

手順３（40 行目に続く）　　：　ＧＯＴＯ　10

④ 手順２　　50　ＩＦ　X＝ 6　ＴＨＥＮ　＿＿＿＿＿＿＿＿

手順３（50 行目に続く）　　：　ＧＯＴＯ　10

⑤ 手順２　　60　＿＿＿＿＿＿＿＿

手順３　　70　ＧＯＴＯ　10

※自分の「ポケコン」と「搬送車」で試してみよう。

気付いたことをまとめよう。

☆10 行目のＡＳＣ（ＩＮＫＥＹ$）－ 48 は，入力待ちにならないキー入力です。
何も押されていないとき（キー入力なし）や，4，2，8，6 以外のキーが押されたときには，60 行目の処理（ブレーキ）が実行されます。

［考察］魔法の引き出しを使って，手順２を整理しよう

＜数字（0～9）キーの値と制御データの関係＞

①キーの値［4］のとき，制御データ「前進」　＝ ＯＵＴ　4
②キーの値［2］のとき，制御データ「左旋回」＝ ＯＵＴ　6
③キーの値［8］のとき，制御データ「右旋回」＝ ＯＵＴ　5
④キーの値［6］のとき，制御データ「後進」　＝ ＯＵＴ　3

表にまとめると，以下のようになる。

キーの値	0	1	2	3	4	5	6	7	8	9
ＯＵＴ			6		4		3		5	

○この関係を，魔法の引き出し（＝「配列」という）の引き出しの「名前」と「中身」に当てはめる。

「名前」	D(0)	D(1)	D(2)	D(3)	D(4)	D(5)	D(6)	D(7)	D(8)	D(9)
「中身」			6		4		3		5	

☆魔法の引き出し（＝「配列」）とは：

D(2) を指定すると，　6　が取り出せる。
↑　　↑
引き出しの名前　　引き出しの中身（＝値）
＝「キー入力」　　＝「モータ出力」

→　プログラムの中でよく使われる便利な引き出しで，D(0) のような名前で呼ばれる。
名前は，必ず 0 番から始まり，D(1)，D(2)，・・・のように， 1 番，2 番と続く。

◎この魔法の引き出し（＝「配列」）を使うときには，大事な約束事があります。

①引き出しの大きさは， ＤＩＭ　D(9) という『オマジナイ』で指定する。
②引き出しの中身（＝値）を，必要に応じて設定する。 →　（例）D(2) ＝ 6

この約束事は，以下のように，前処理という形で実行します。

```
前処理①    10  ＤＩＭ  D( 9 )      ← 引き出しの大きさの指定
前処理②    20  D( 4 ) ＝ 4         ← 4 が入力されたら，ＯＵＴ 4
〃          30  D( 2 ) ＝ 6         ← 2 が入力されたら，ＯＵＴ 6
〃          40  D( 8 ) ＝ 5         ← 8 が入力されたら，ＯＵＴ 5
〃          50  D( 6 ) ＝ 3         ← 6 が入力されたら，ＯＵＴ 3
```

＜プログラム化（前処理の続き）＞

手順１　キー入力　⇨　入力待ちにならないキー入力

① 手順２　ＩＦ文（もし，4，2，8，6 だったら）魔法の引き出しで指定

手順３　手順１へ戻る

② 手順２　その他のキー　⇨　「ブレーキ」

手順３　手順１へ戻る

※キー入力により，手順２のどちらか一つ（①か②）が選択されます。

問題　前処理の続きのプログラムを作ってみよう。

ヒント：remote を実行してみよう

※前のページの 50 行目の続き

手順１　60　Ｘ ＝ ＡＳＣ（ＩＮＫＥＹ$）－ 48

① 手順２　70　ＩＦ　Ｘ＝ 4　ＯＲ　Ｘ＝ 2　ＯＲ　Ｘ＝ 8

ＯＲ　Ｘ＝ 6　ＴＨＥＮ　＿＿＿＿＿＿

手順３（70 行目に続く）　：　ＧＯＴＯ　60

② 手順２　80　＿＿＿＿＿＿

手順３　90　ＧＯＴＯ　60

※自分の「ポケコン」と「搬送車」で試してみよう。

気付いたことをまとめよう。