速習！情報 ～共通テスト対策講座～

プログラミング(反復構造)

反復構造(繰り返し構造)、ループ

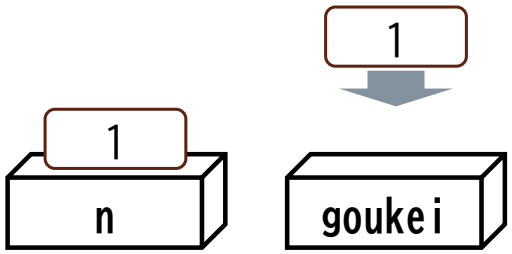
反復構造＝条件が満たされる間、処理を繰り返す

1～□までの数を足していき合計を表示する　　1＋2＋3＋4＋……＋□

反復構造＝条件が満たされる間、処理を繰り返す

1～□までの数を足していき合計を表示する　　1+2+3+4+……+□＝goukei
n

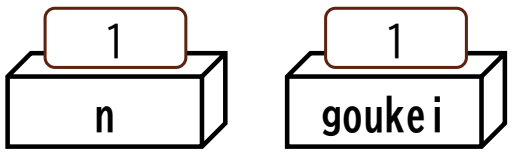
イメージ



反復構造＝条件が満たされる間、処理を繰り返す

1～□までの数を足していき合計を表示する　　1+2+3+4+……+□＝goukei
n

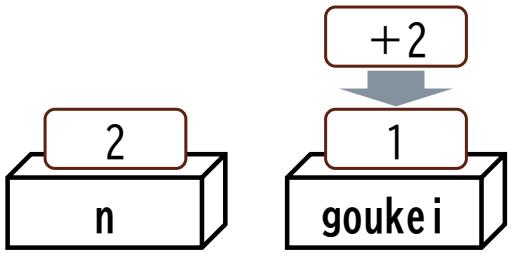
イメージ



反復構造＝条件が満たされる間、処理を繰り返す

1～□までの数を足していき合計を表示する　　1＋2＋3＋4＋……＋□＝goukei
n

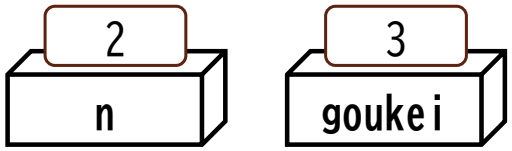
イメージ



反復構造＝条件が満たされる間、処理を繰り返す

1～□までの数を足していき合計を表示する　　1＋2＋3＋4＋……＋□＝goukei
n

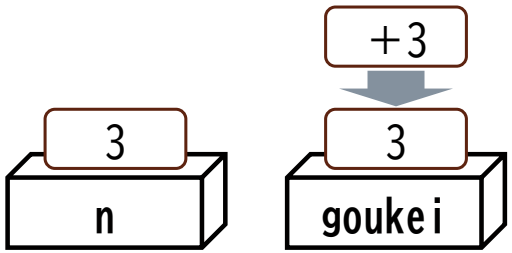
イメージ



反復構造＝条件が満たされる間、処理を繰り返す

1～□までの数を足していき合計を表示する　　1＋2＋3＋4＋……＋□＝goukei
n

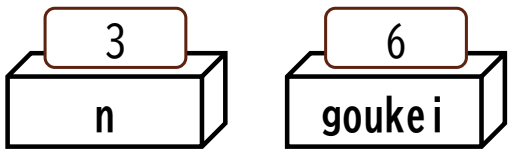
イメージ



反復構造＝条件が満たされる間、処理を繰り返す

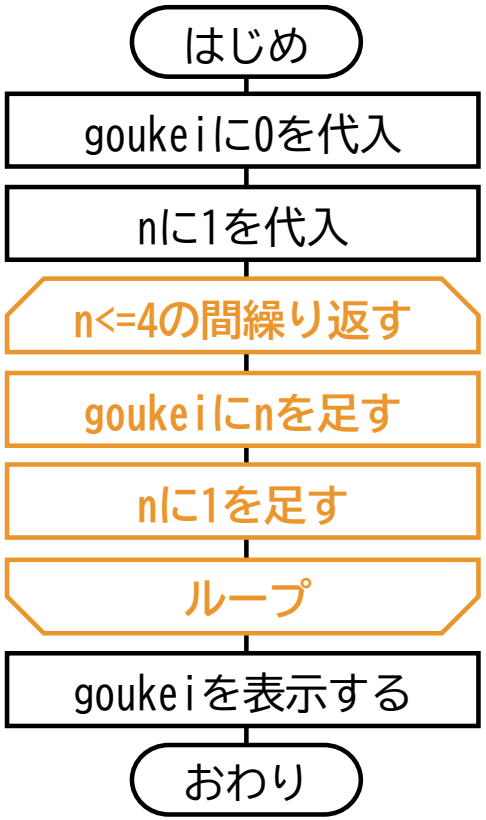
1～□までの数を足していき合計を表示する　　1＋2＋3＋4＋……＋□＝goukei
n

イメージ



反復構造＝条件が満たされる間、処理を繰り返す

1～4までの数を足していき合計を表示する

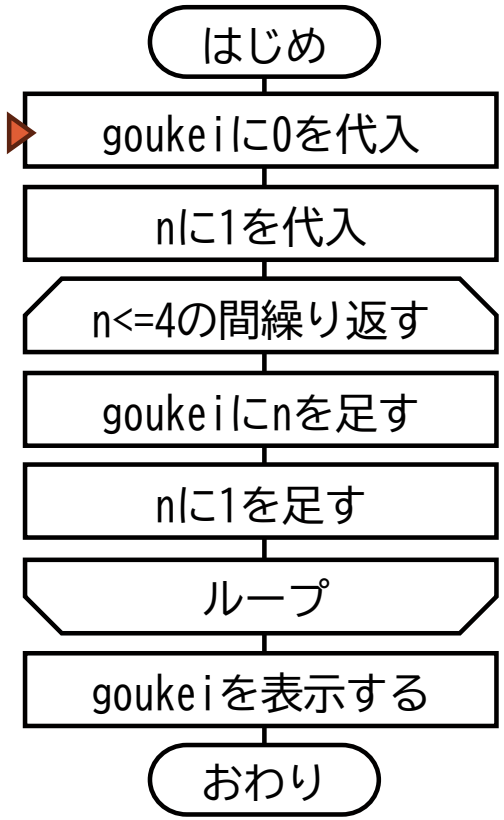


1+2+3+4

反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない

1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

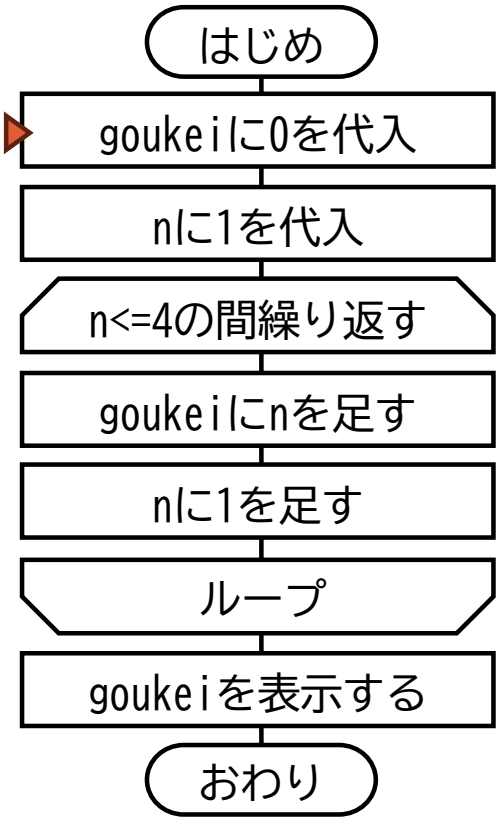
```
goukei = 0
```

1＋2＋3＋4

反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない

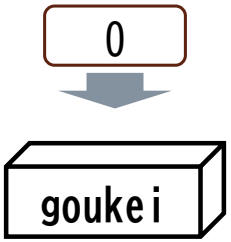
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
```

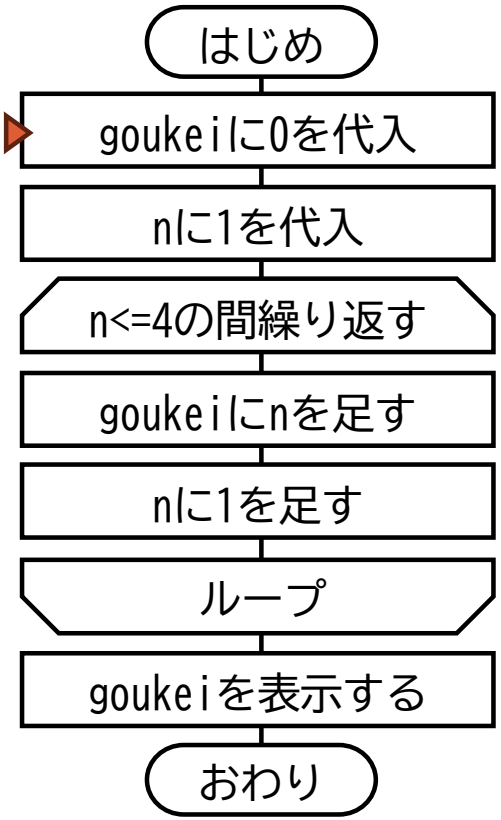
1＋2＋3＋4



反復構造＝条件が満たされる間、処理を繰り返す

・「＝」は代入の意味　※等式ではない

1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

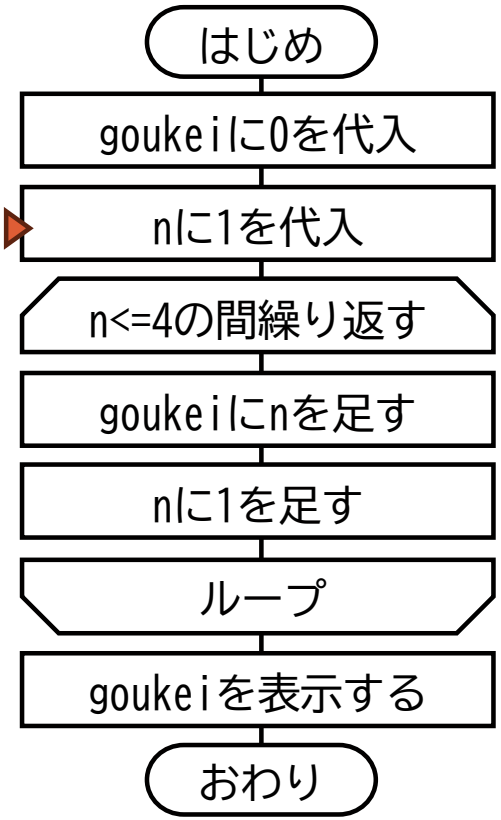
```
goukei = 0
```

1＋2＋3＋4

反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない

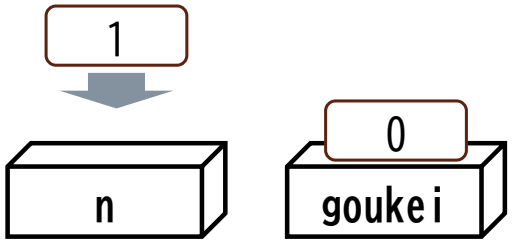
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
```

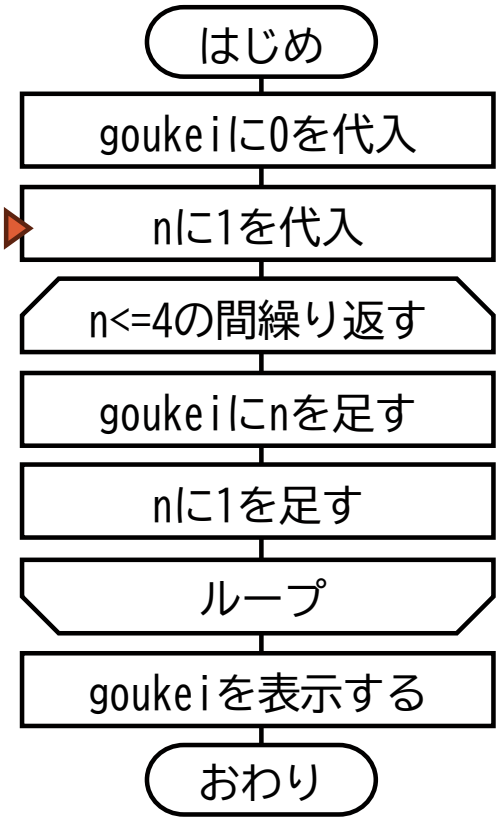
1＋2＋3＋4
n



反復構造＝条件が満たされる間、処理を繰り返す

・「＝」は代入の意味　※等式ではない

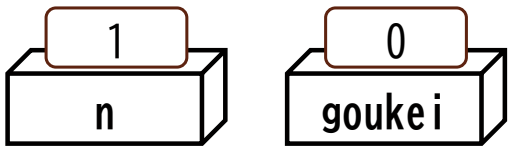
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
```

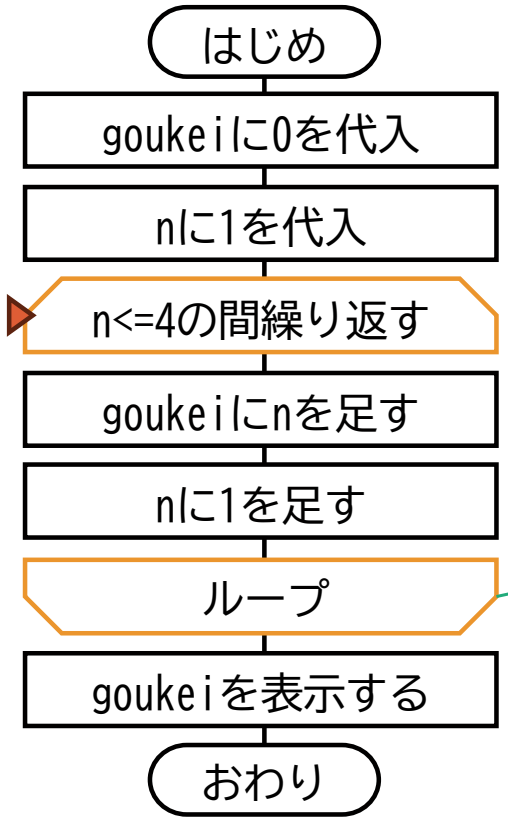
1＋2＋3＋4
n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない

1～4までの数を足していき合計を表示する

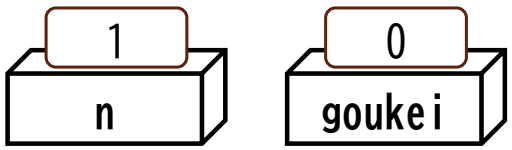


共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：

```

反復処理の終了部分を意味する

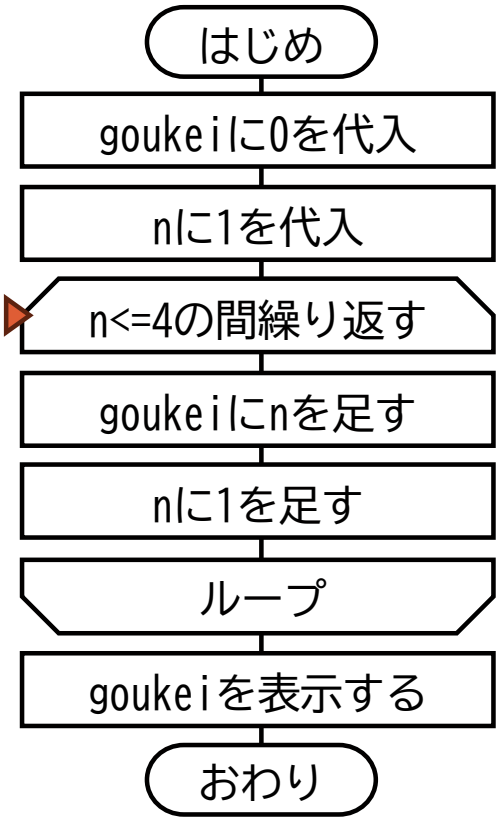
1+2+3+4
n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない

1～4までの数を足していき合計を表示する

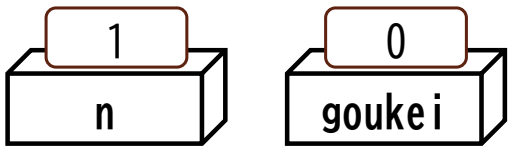


共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
```

1+2+3+4
n

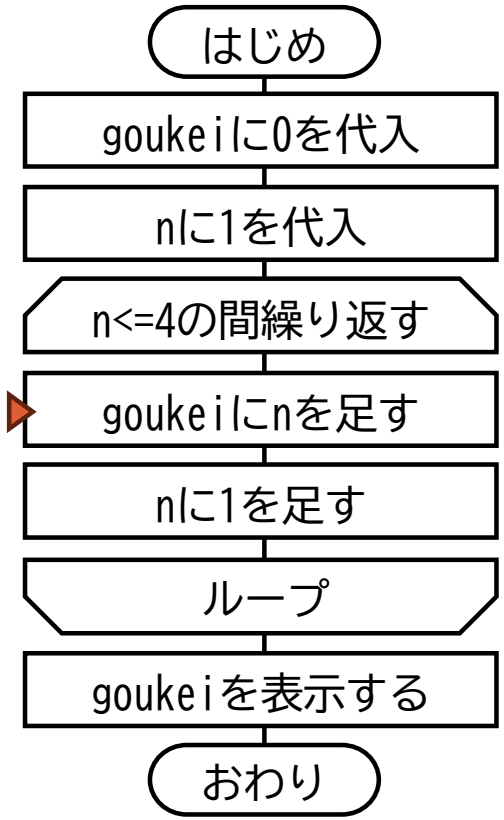
いまのnは1で、条件を満たす



反復構造＝条件が満たされる間、処理を繰り返す

・「＝」は代入の意味　※等式ではない

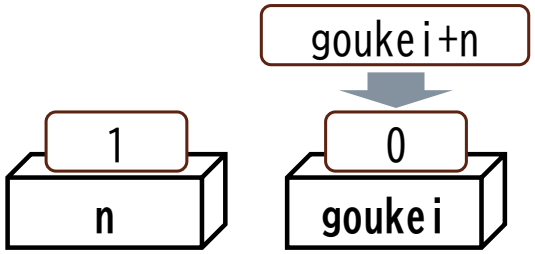
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
```

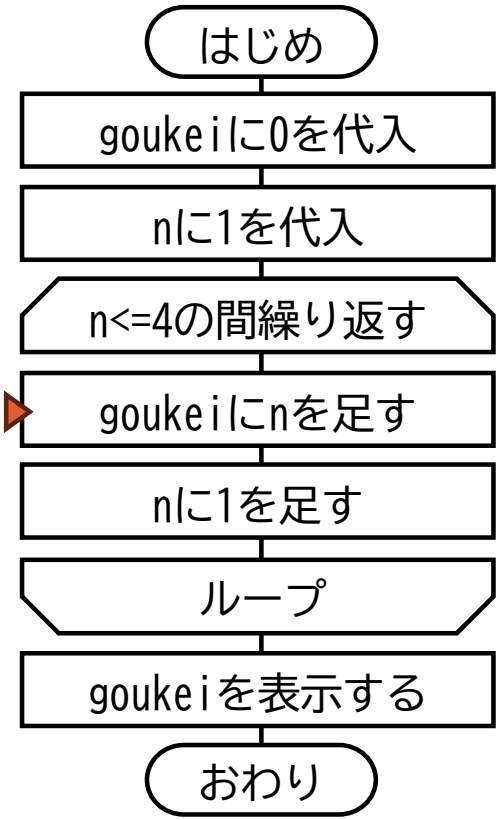
1+2+3+4
n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

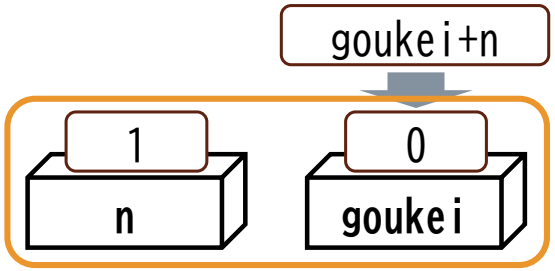
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
```

1+2+3+4
n

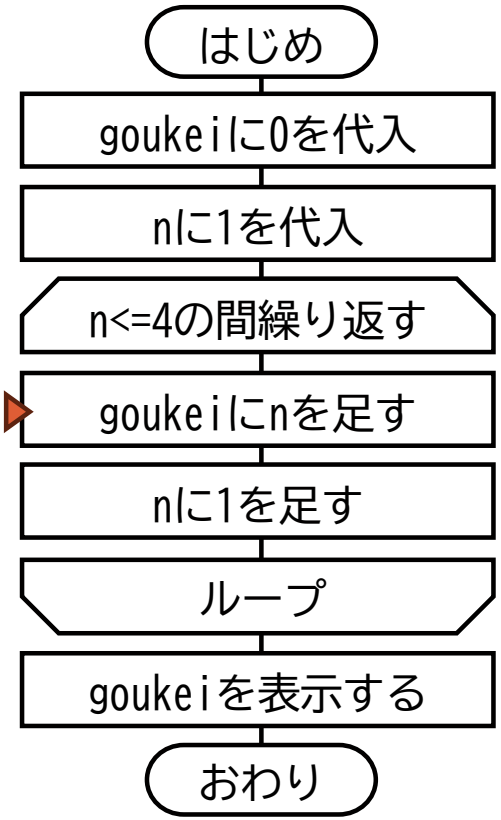


いまのgoukeiは0
いまのnは1

反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

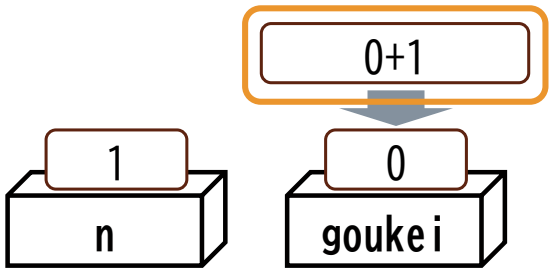
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
```

1+2+3+4
n

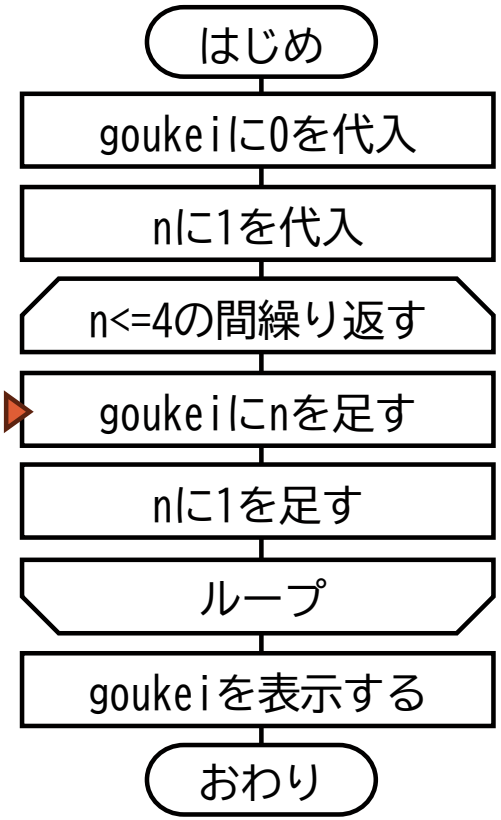


いまのgoukeiは0
いまのnは1

反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

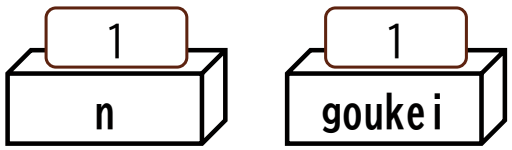
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
```

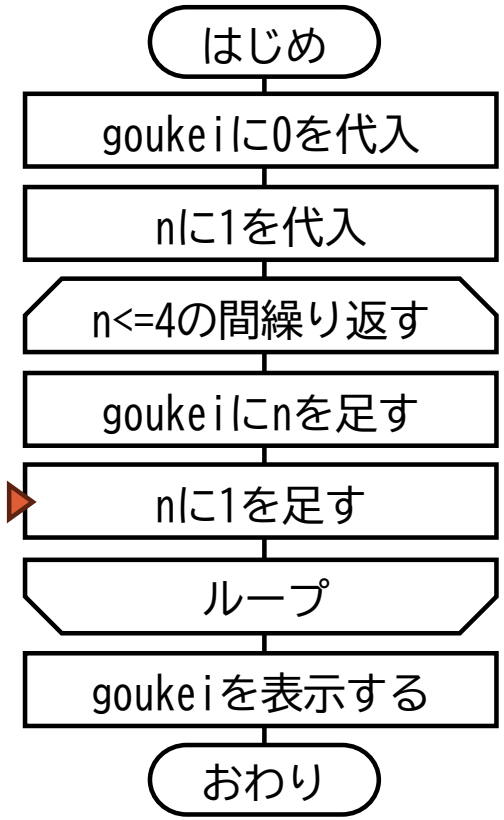
1+2+3+4
n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

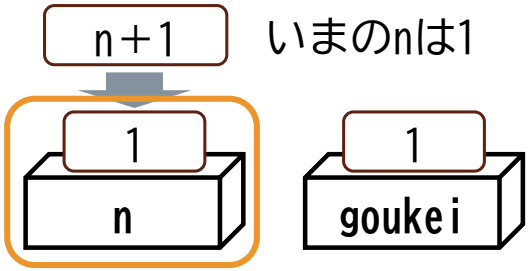
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

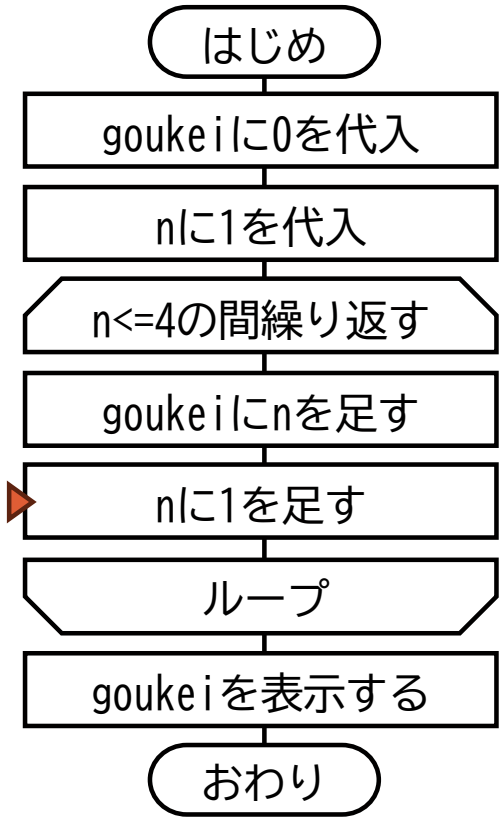
1＋2＋3＋4
　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

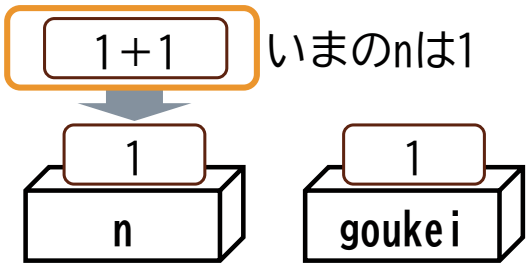
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

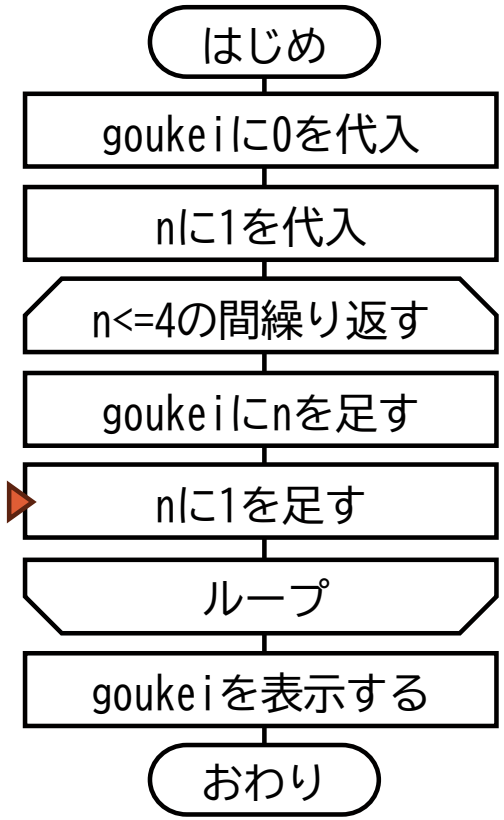
1＋2＋3＋4
　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

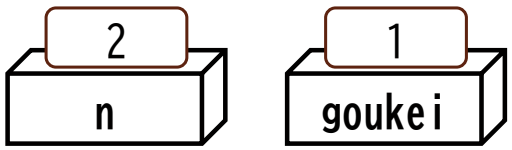
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

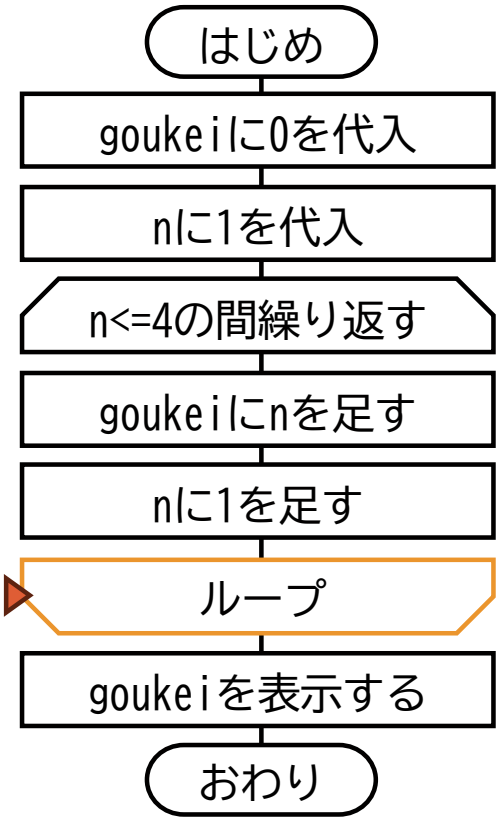
1＋2＋3＋4
　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

1～4までの数を足していき合計を表示する

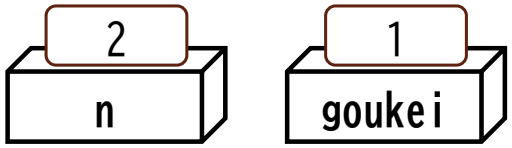


共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

反復処理の終了部分を意味する

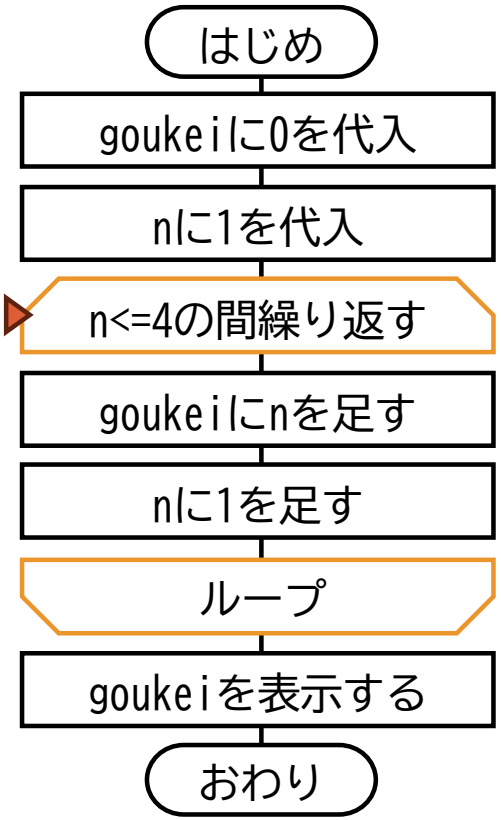
1＋2＋3＋4
　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

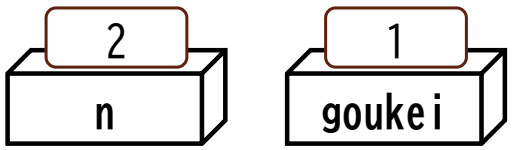
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

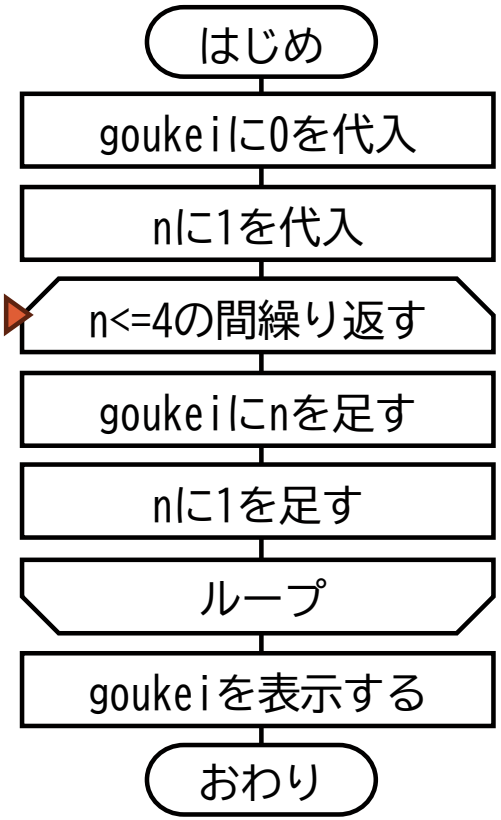
1＋2＋3＋4
　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

1～4までの数を足していき合計を表示する

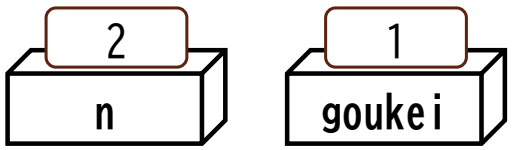


共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

1＋2＋3＋4
　n

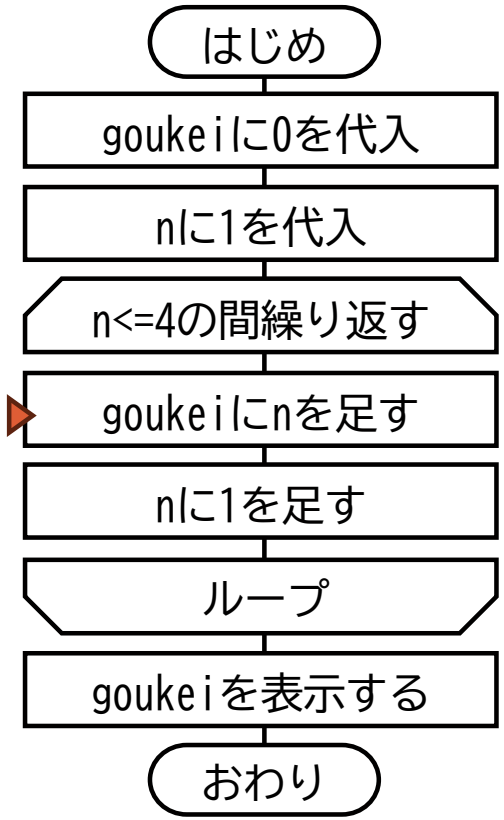
いまのnは2で、条件を満たす



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

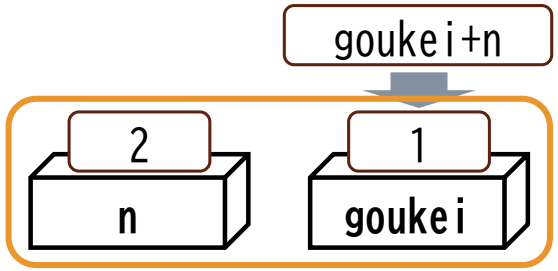
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

1＋2＋3＋4
　n

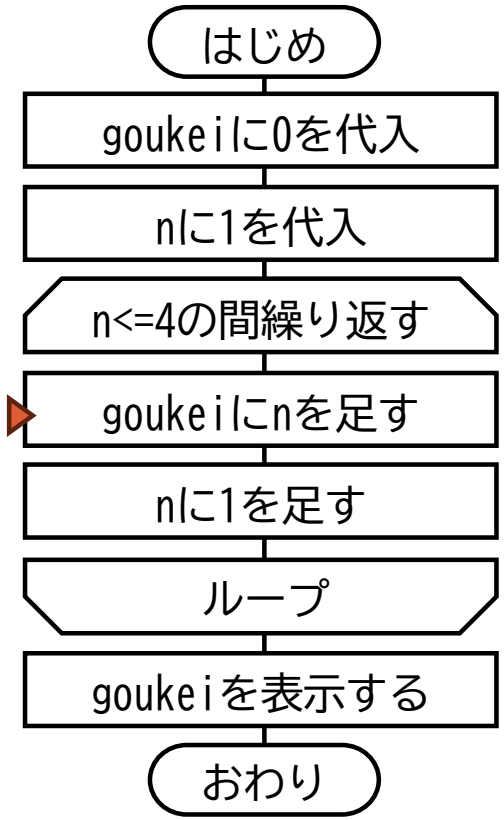


いまのgoukeiは1
いまのnは2

反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

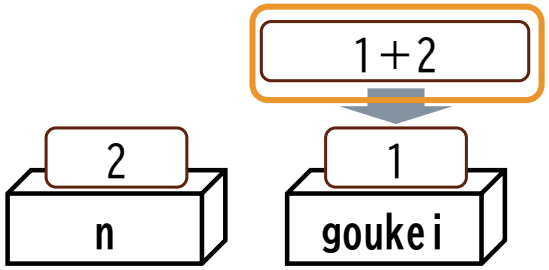
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

1＋2＋3＋4
　n

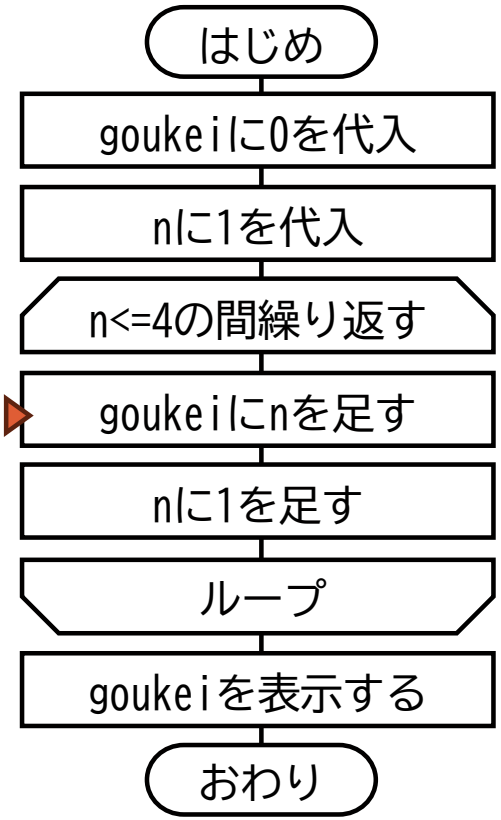


いまのgoukeiは1
いまのnは2

反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

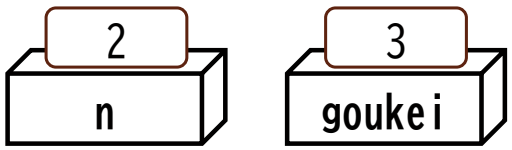
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

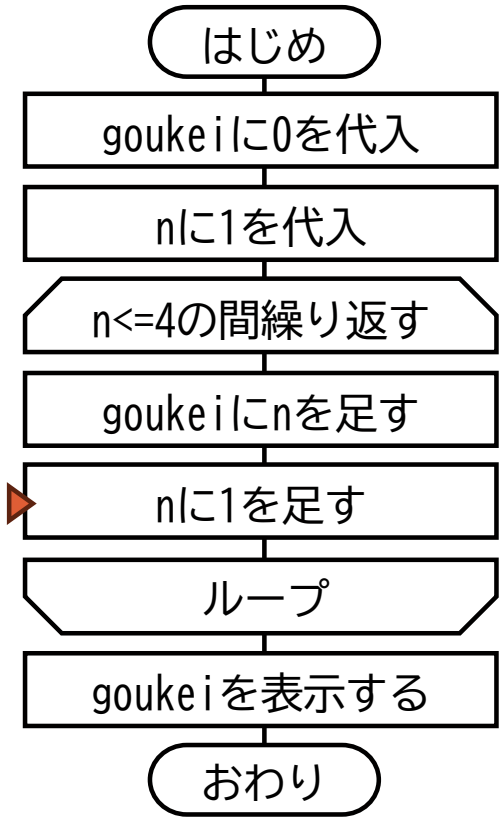
1＋2＋3＋4
　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

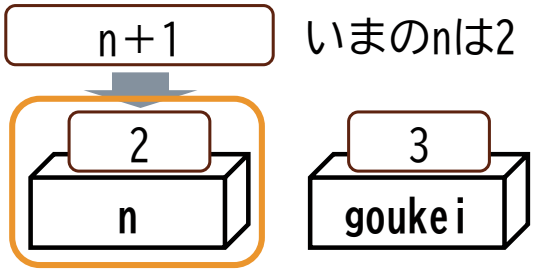
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

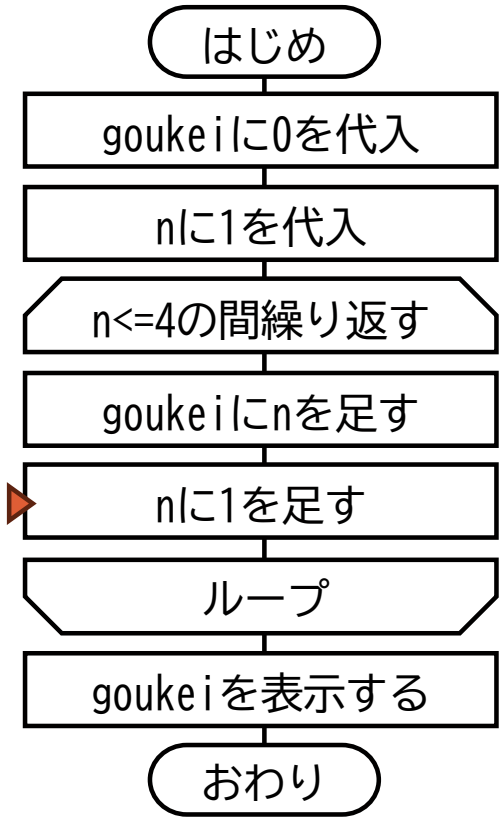
1＋2＋3＋4
　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

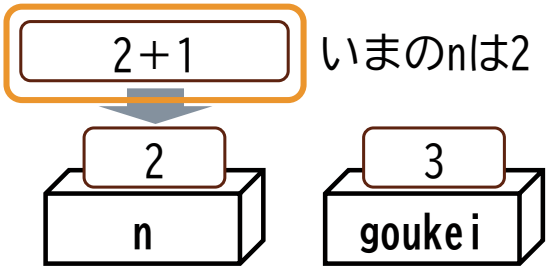
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

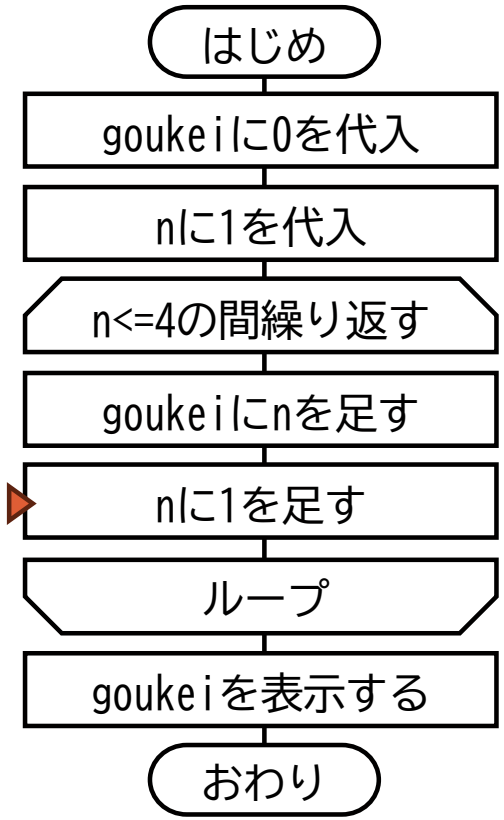
1＋2＋3＋4
　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

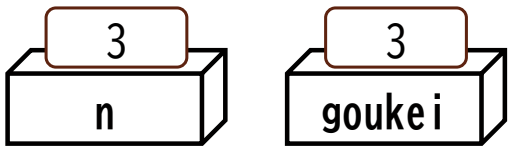
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

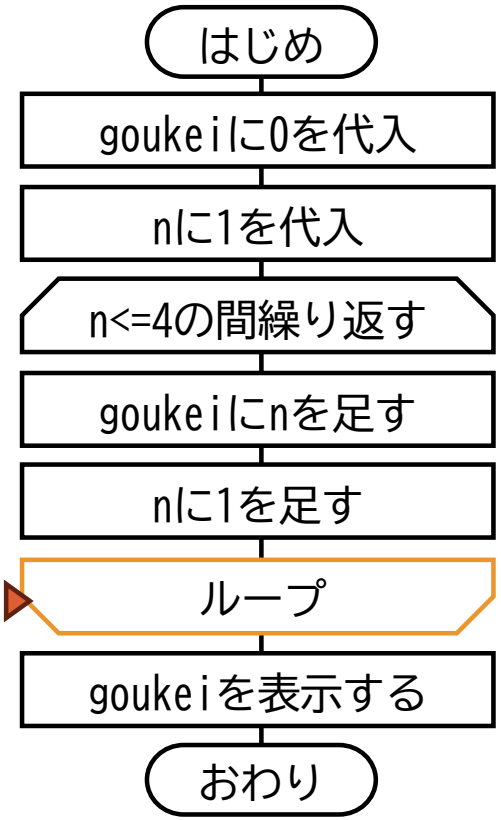
1＋2＋3＋4
　　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

1～4までの数を足していき合計を表示する

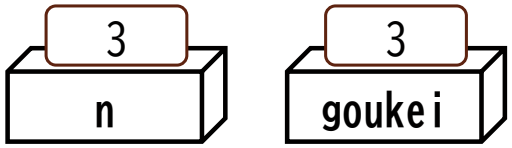


共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

反復処理の終了部分を意味する

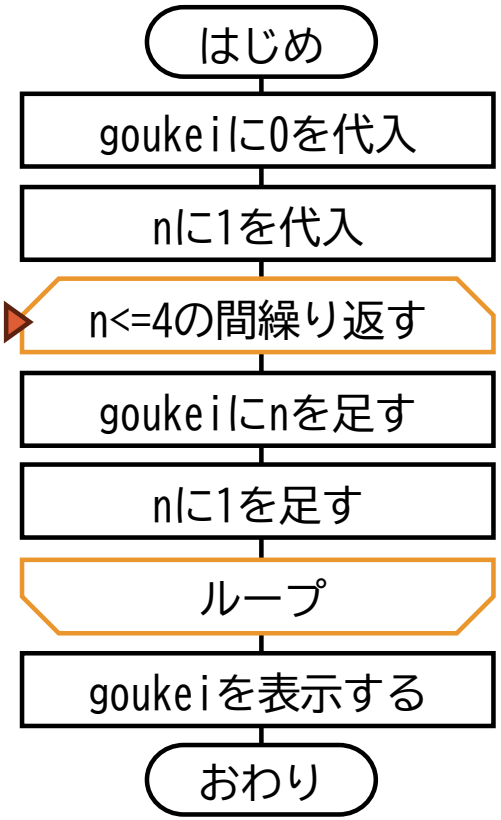
1＋2＋3＋4
　　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

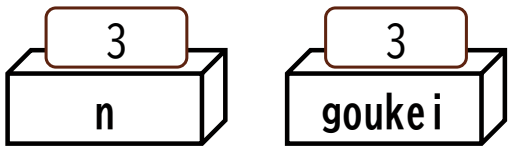
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

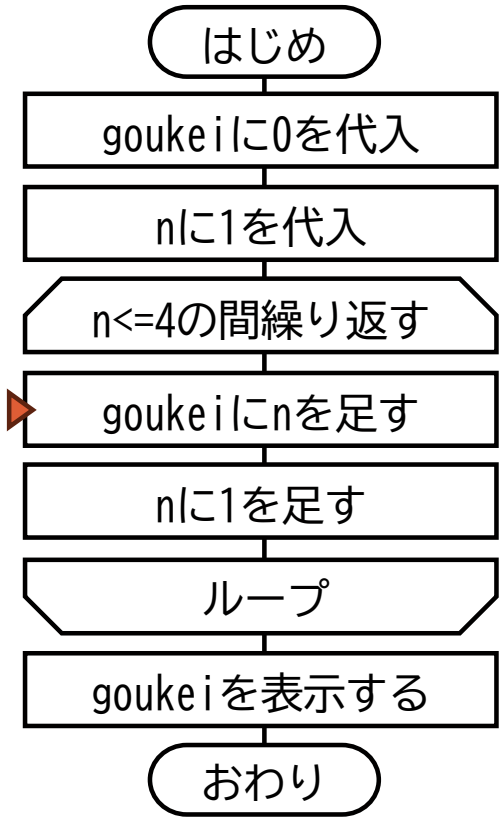
1＋2＋3＋4
　　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

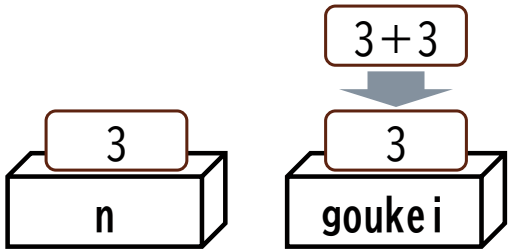
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

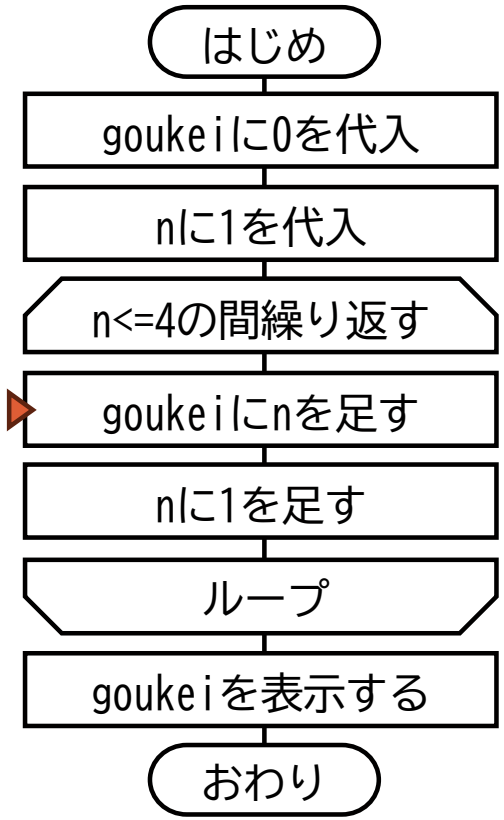
1＋2＋3＋4
n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

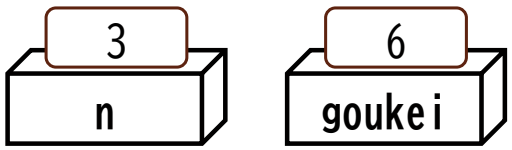
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

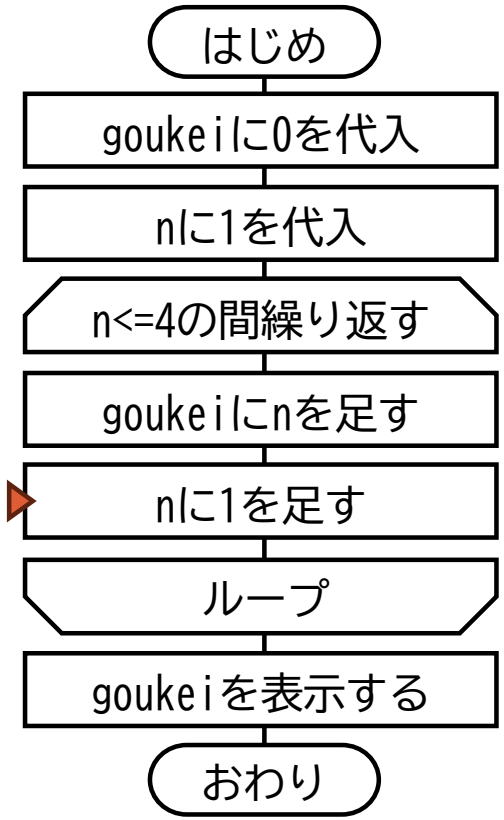
1＋2＋3＋4
　　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

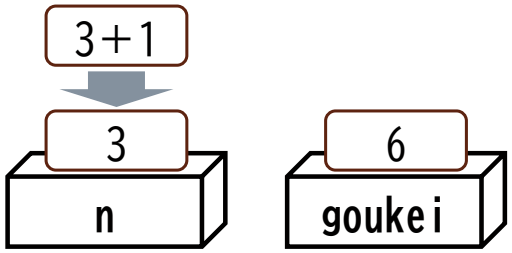
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

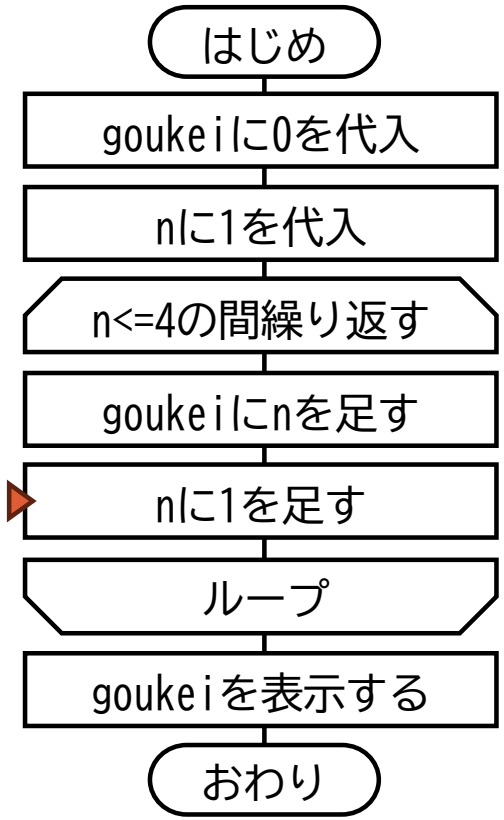
1＋2＋3＋4
　　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

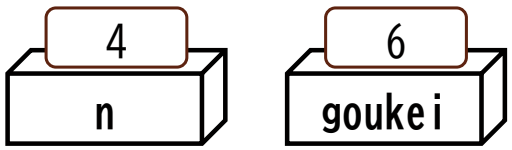
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

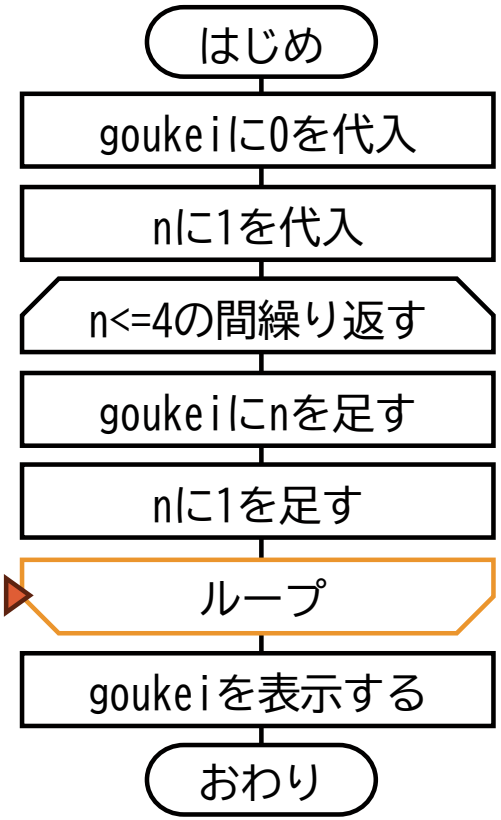
1＋2＋3＋4
　　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

1～4までの数を足していき合計を表示する



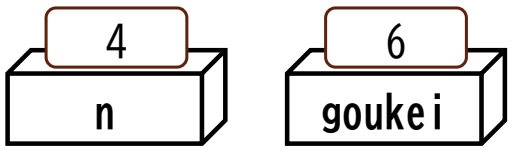
共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

反復処理の終了部分を意味する

1＋2＋3＋4

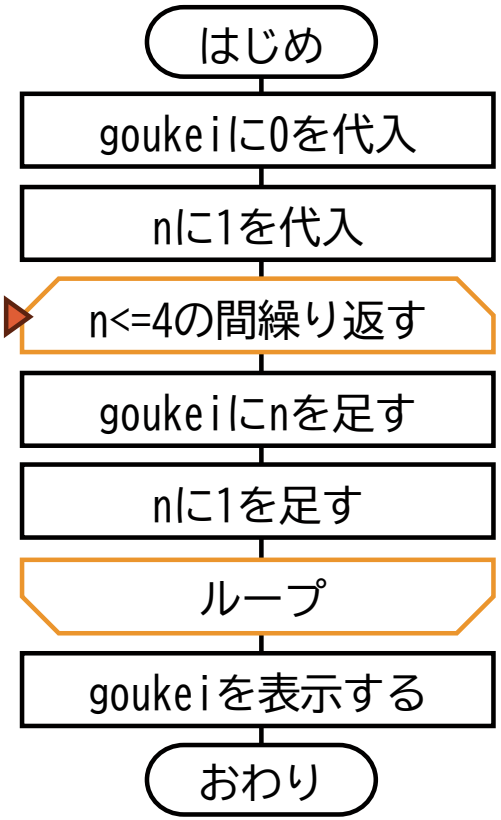
n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

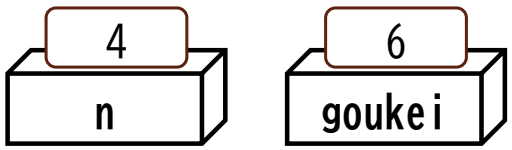
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

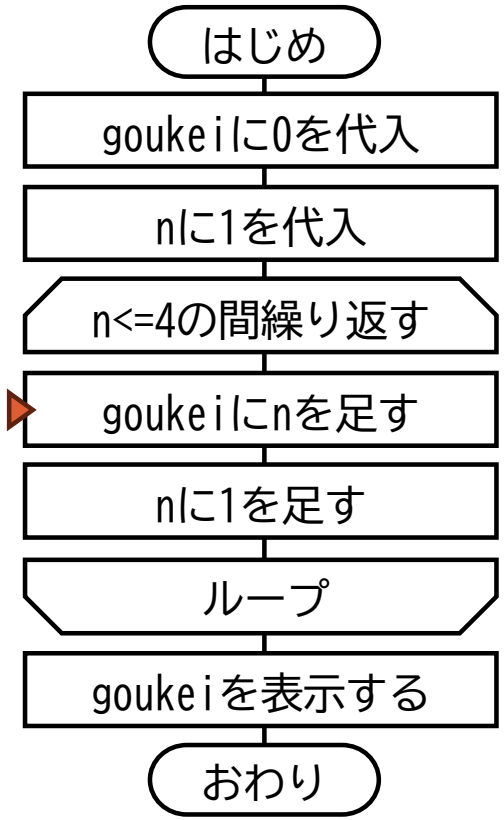
1＋2＋3＋4
　　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

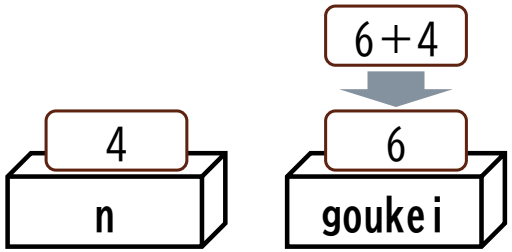
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

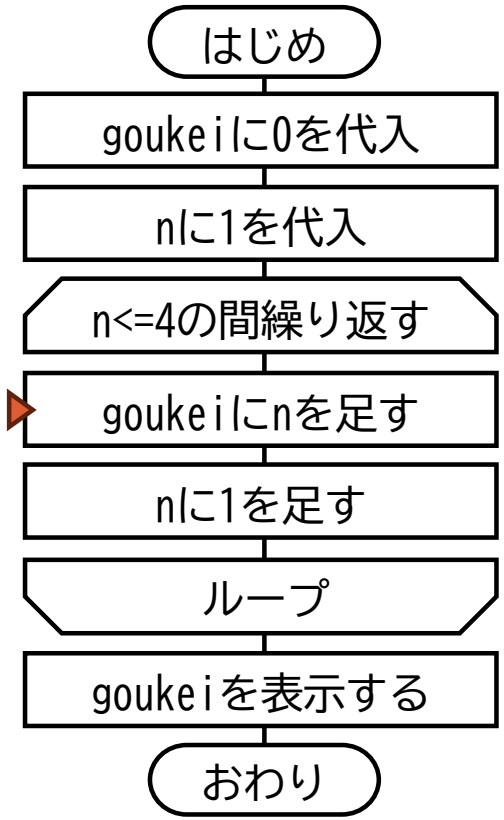
1＋2＋3＋4
n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

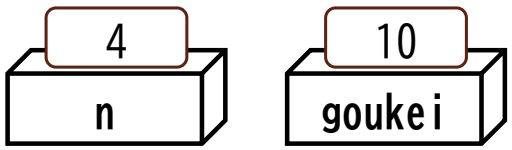
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

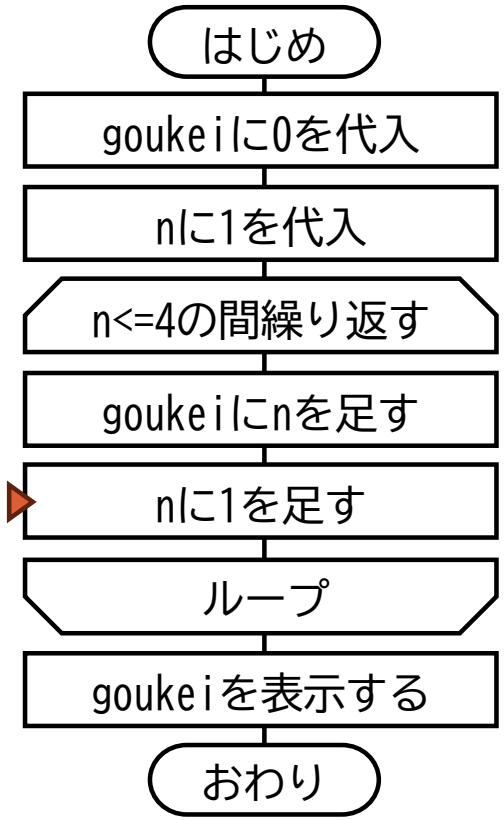
1＋2＋3＋4
　　n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

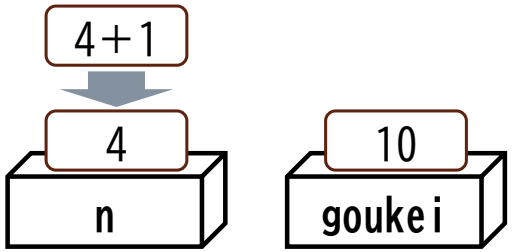
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

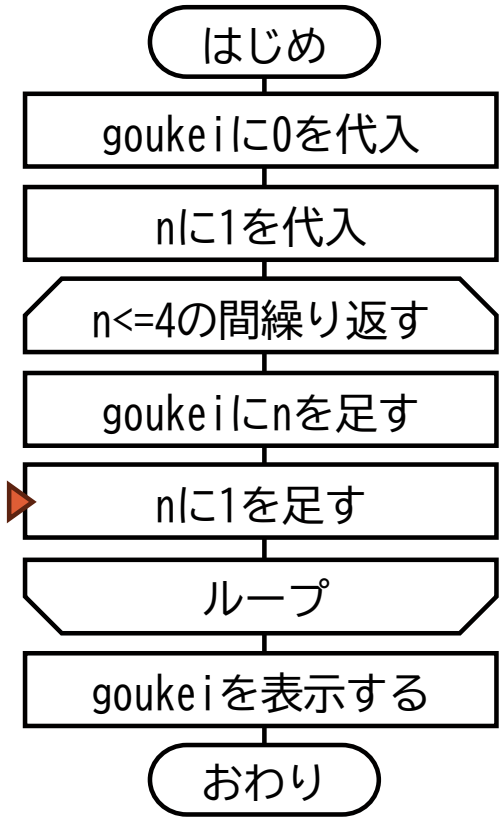
1＋2＋3＋4
n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

1～4までの数を足していき合計を表示する

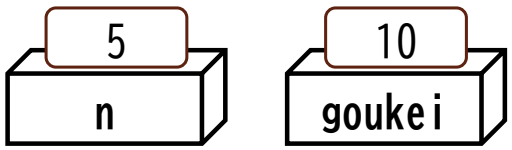


共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

1＋2＋3＋4 □

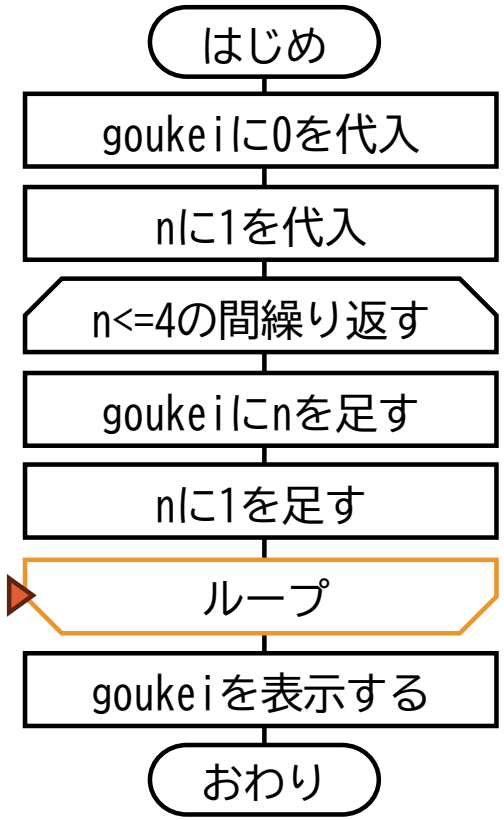
n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

1～4までの数を足していき合計を表示する

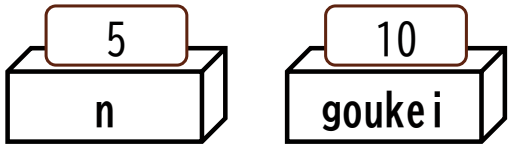


共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

反復処理の終了部分を意味する

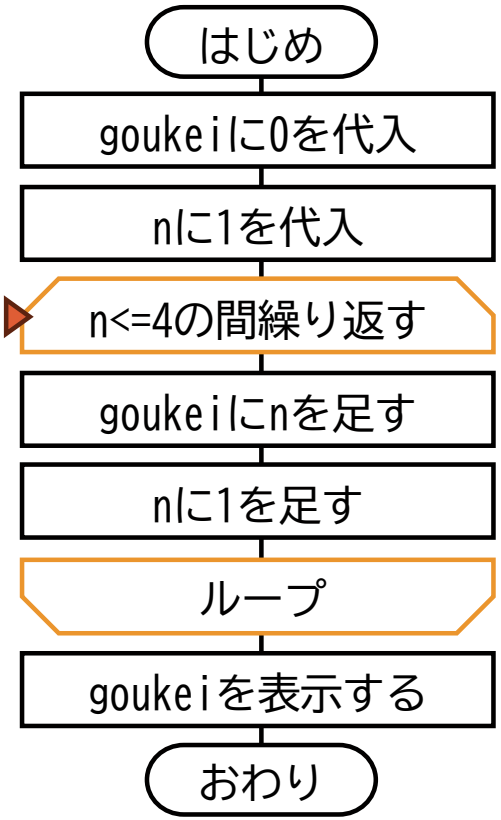
1＋2＋3＋4



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

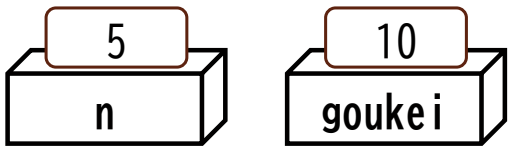
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

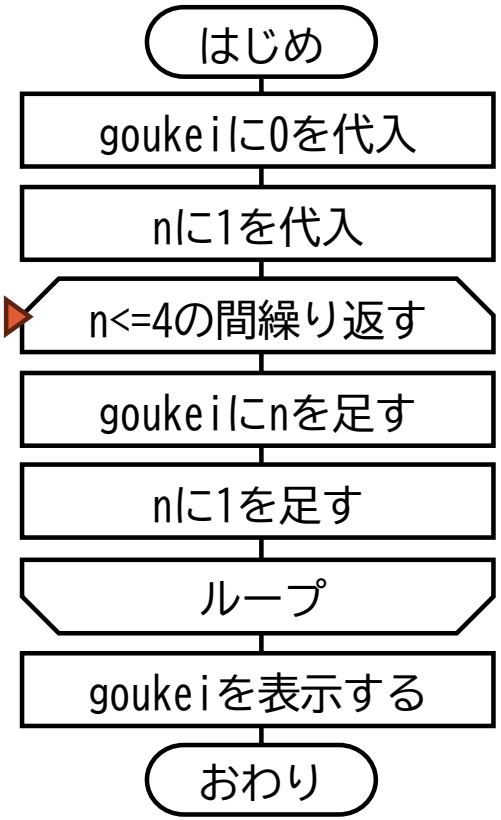
1＋2＋3＋4



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

1～4までの数を足していき合計を表示する



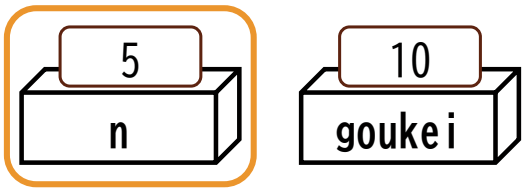
共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

1＋2＋3＋4 □

n

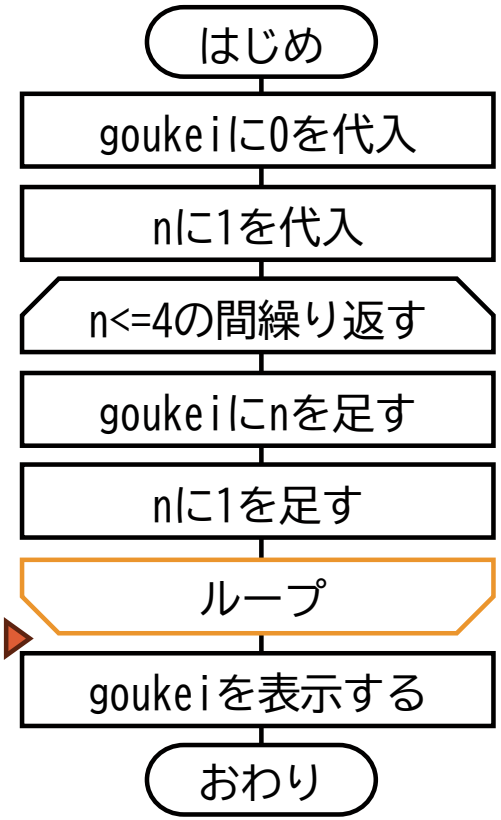
いまのnは5で、条件を満たさない



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する

1～4までの数を足していき合計を表示する



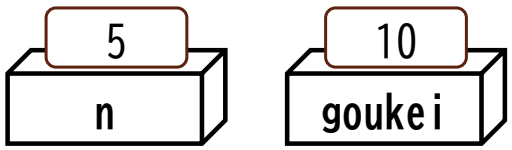
共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
```

反復処理の終了部分を意味する

1＋2＋3＋4

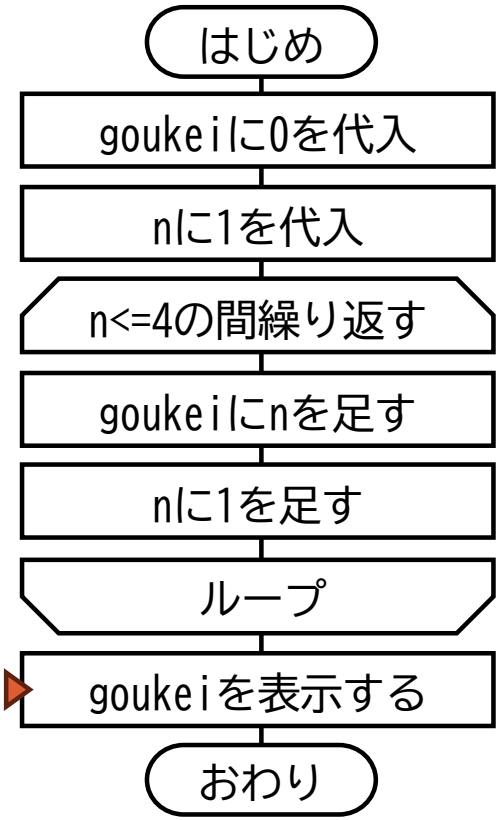
いまのnは5で、条件を満たさない



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する
  - 変数は値に置き換える
  - 「”」で囲われたら文字列そのまま

1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

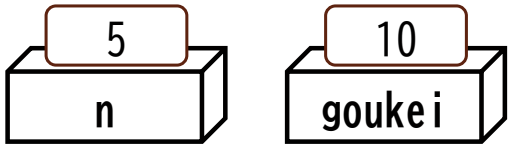
```
goukei = 0
n = 1
n <= 4 の間繰り返す：
  goukei = goukei + n
  n = n + 1
表示する(goukei)
```

いまのnは5で、条件を満たさない

いまのgoukeiは10

1＋2＋3＋4

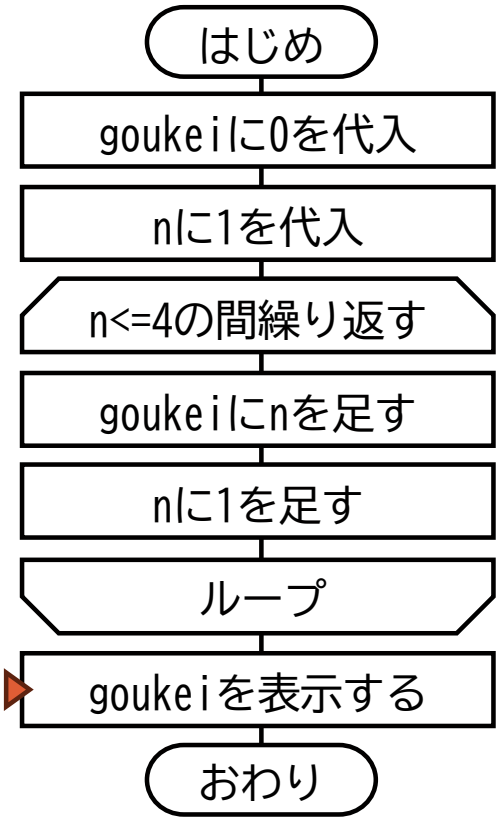
n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する
  - 変数は値に置き換える
  - 「”」で囲われたら文字列そのまま

1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
  goukei = goukei + n
  n = n + 1
表示する(goukei)
```

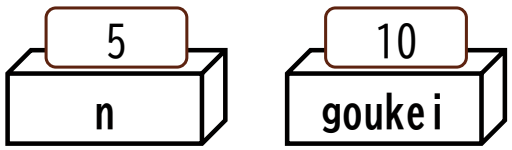
いまのnは5で、条件を満たさない

「10」と表示される

いまのgoukeiは10

1＋2＋3＋4

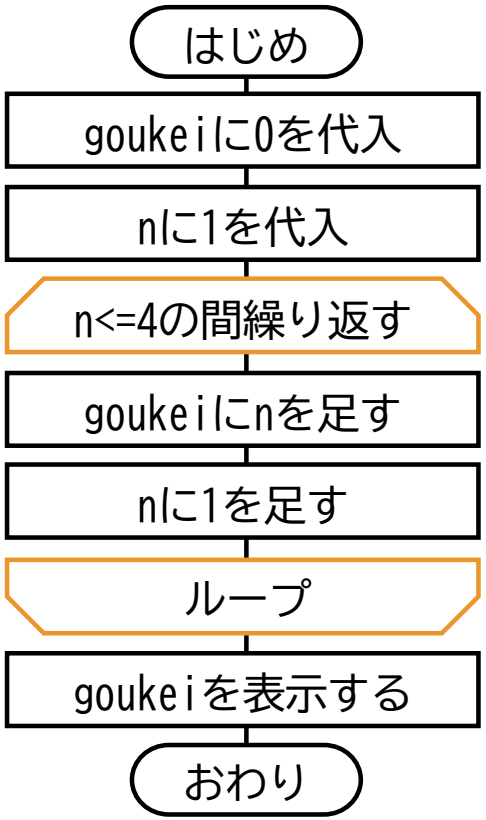
n



反復構造＝条件が満たされる間、処理を繰り返す

- 「＝」は代入の意味　※等式ではない
- 計算をしてから変数に代入する
  - 変数は値に置き換える
  - 「”」で囲われたら文字列そのまま

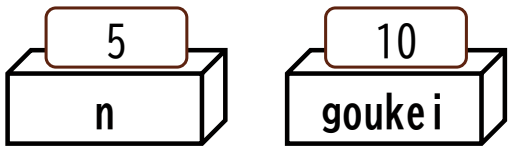
1～4までの数を足していき合計を表示する



共通テストで使用されるDNCL

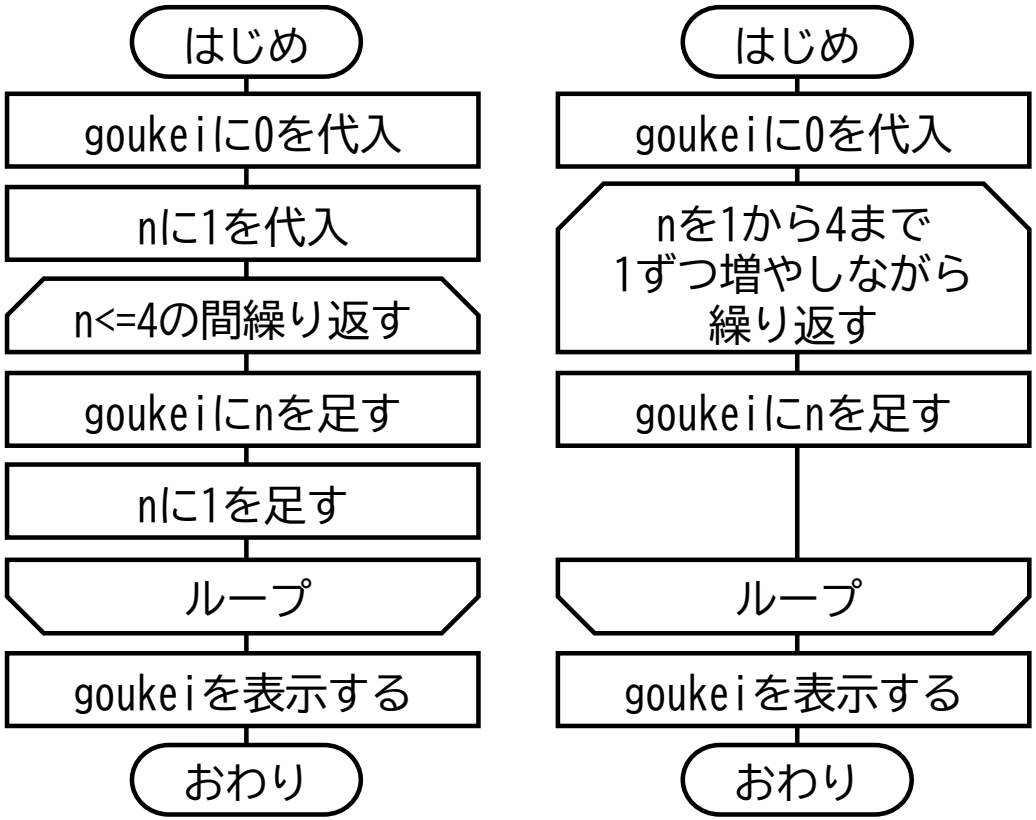
```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
表示する(goukei)
```

反復構造の1つ目は「～の間繰り返す」と終了記号で表す



反復構造には2通りの方法がある

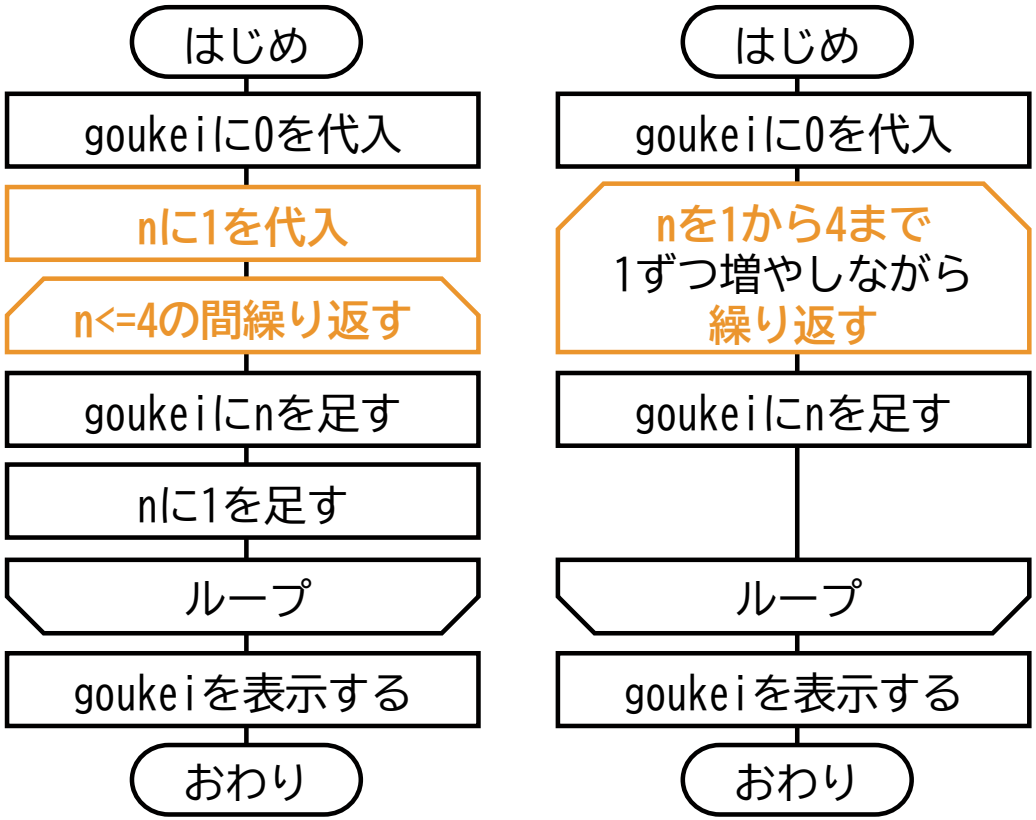
1～4までの数を足していき合計を表示する



- 反復構造の1つ目は「～の間繰り返す」と終了記号で表す
- 反復構造の2つ目は「～ながら繰り返す」と終了記号で表す

反復構造には2通りの方法がある

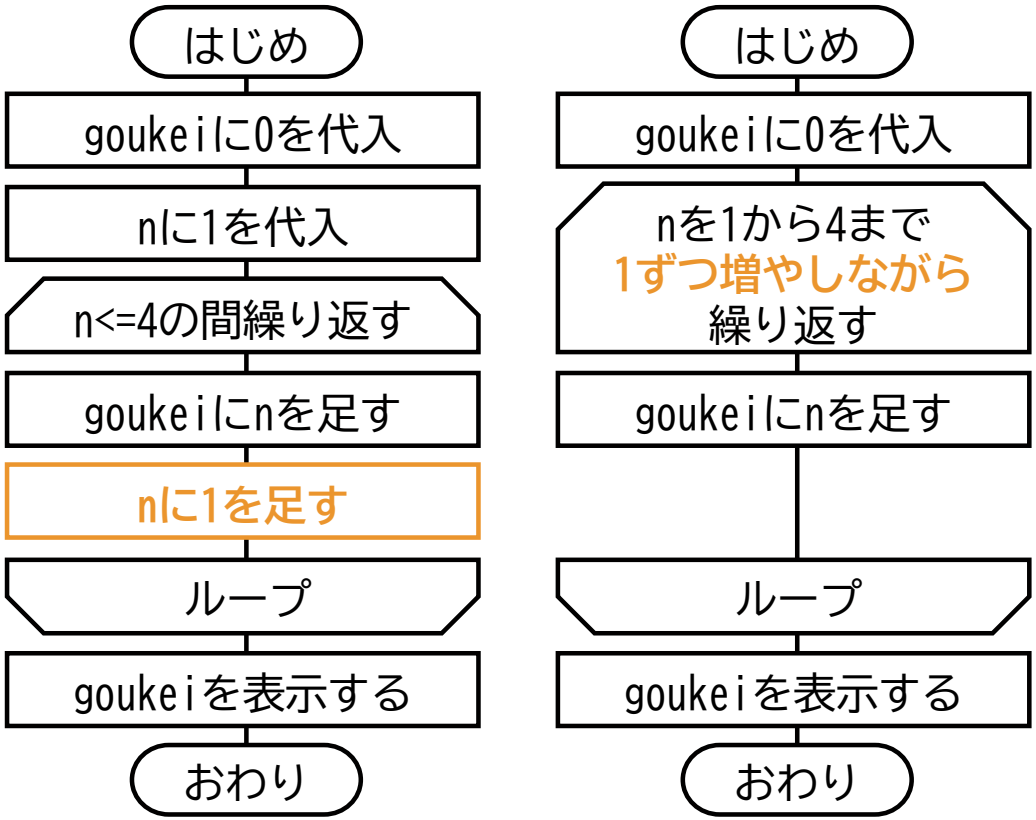
1～4までの数を足していき合計を表示する



- 反復構造の1つ目は「～の間繰り返す」と終了記号で表す
- 反復構造の2つ目は「～ながら繰り返す」と終了記号で表す

反復構造には2通りの方法がある

1～4までの数を足していき合計を表示する



- 反復構造の1つ目は「～の間繰り返す」と終了記号で表す
- 反復構造の2つ目は「～ながら繰り返す」と終了記号で表す

反復構造には2通りの方法がある

1～4までの数を足していき合計を表示する

共通テストで使用されるDNCL

```
goukei = 0

nを1から4まで1ずつ増やしながら繰り返す：
    goukei = goukei + n

表示する(goukei)
```

反復処理の終了部分を意味する

反復構造には2通りの方法がある

1～4までの数を足していき合計を表示する

共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
表示する(goukei)
```

- 反復構造の1つ目は「～の間繰り返す」と終了記号で表す

共通テストで使用されるDNCL

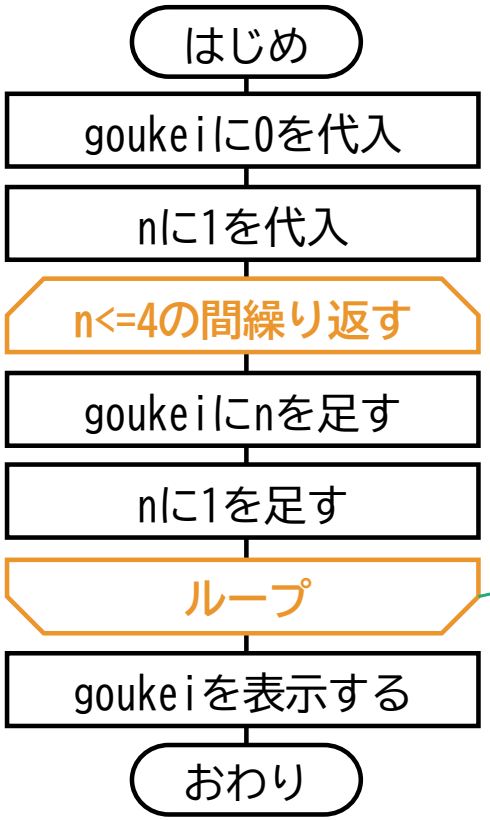
```
goukei = 0

nを1から4まで1ずつ増やしながら繰り返す：
    goukei = goukei + n

表示する(goukei)
```

- 反復構造の2つ目は「～ながら繰り返す」と終了記号で表す

「プログラミング(反復構造)」の要点

「プログラミング(反復構造)」の要点1

反復構造：条件が満たされる間、処理を繰り返し実行する



共通テストで使用されるDNCL

```
goukei = 0
n = 1
n <= 4 の間繰り返す：
    goukei = goukei + n
    n = n + 1
表示する(goukei)
```

反復処理の終了部分を意味する

反復構造の1つ目は「～の間繰り返す」と終了記号で表す

「プログラミング(反復構造)」の要点2

反復構造：条件が満たされる間、処理を繰り返し実行する

共通テストで使用されるDNCL

```
goukei = 0

nを1から4まで1ずつ増やしながら繰り返す：
    goukei = goukei + n

表示する(goukei)
```

反復処理の終了部分を意味する

反復構造の2つ目は「～しながら繰り返す」と終了記号で表す