11章 関数を理解しよう

11章 関数を理解しよう

関数とは何か概要を学び、実際に使ってみます

本章の目標

- 関数とは何か概要をつかむこと
- 関数の作り方・呼び出し方を知ること
- 関数を実際に使ってみること

同じ処理を何度も書くのは大変で、コードが無駄に長くなります

```
print("おはようございます！")
print("昨日はよく眠れましたか？")
print("今日も一日頑張りましょう！")

print("おはようございます！")
print("昨日はよく眠れましたか？")
print("今日も一日頑張りましょう！")
```

処理の一部を変更したいときに、すべてのコードを書き換えなければなりません

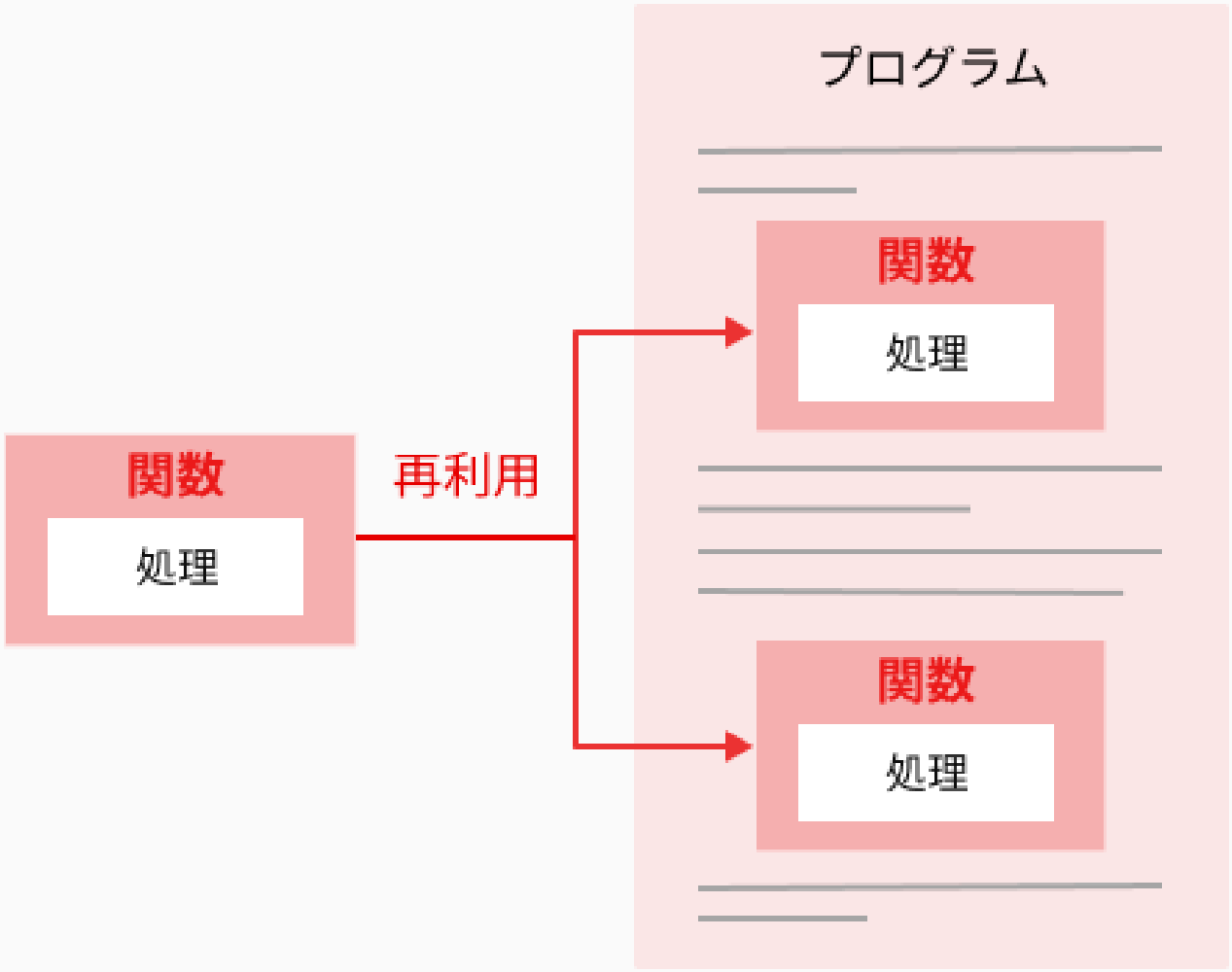
一度関数を作るだけで何度でも再利用できます

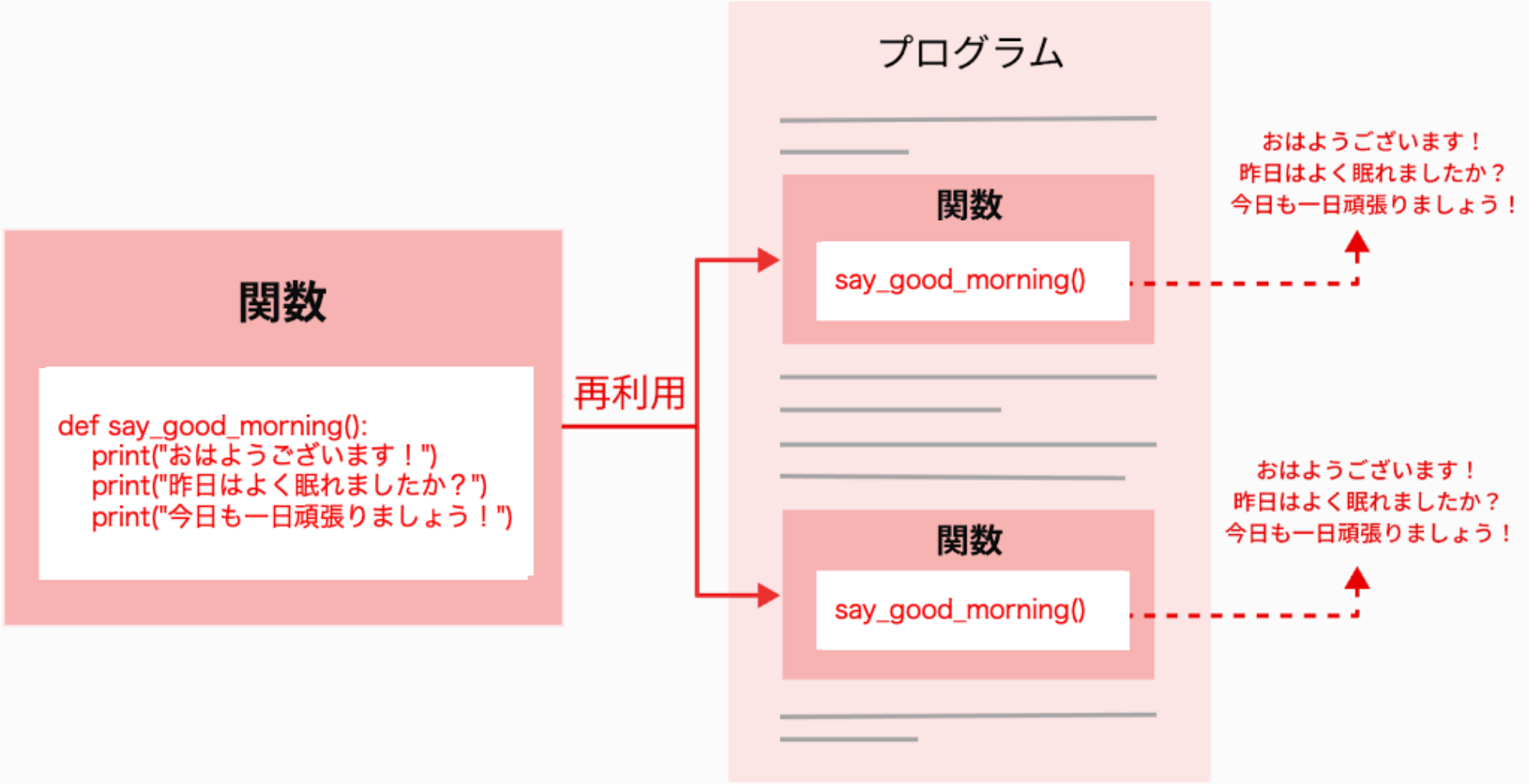
```
# 朝のあいさつを出力する関数を作成する
def say_good_morning():
    print("おはようございます！")
    print("昨日はよく眠れましたか？")
    print("今日も一日頑張りましょう！")

# 関数を呼び出す（1回目）
say_good_morning()

# 関数を呼び出す（2回目）
say_good_morning()
```

出力内容を変えたいときも、一度コードを書き換えれば済みます


プログラム
関数
処理
関数
処理
再利用
関数
処理


関数
def say_good_morning():
print("おはようございます！")
print("昨日はよく眠れましたか？")
print("今日も一日頑張りましょう！")
再利用
プログラム
関数
say_good_morning()
関数
say_good_morning()
おはようございます！
昨日はよく眠れましたか？
今日も一日頑張りましょう！
おはようございます！
昨日はよく眠れましたか？
今日も一日頑張りましょう！

11章 関数のメリット

関数を使うことで得られるメリットをまとめます

- 複雑なコードを1つにまとめられる
- 同じ処理を行うときに何度でも再利用できる
- プログラミングの生産性を高め、素早い開発ができるようになる

defキーワードを使って関数を定義します

```
def 関数名():
    一連の処理
```

- def：「define」「definition」（定義する）の略
- 関数を作成することを「関数を定義する」と呼ぶ

朝のあいさつの例

```
def say_good_morning():
    print("おはようございます！")
    print("昨日はよく眠れましたか？")
    print("今日も一日頑張りましょう！")
```

- 関数名は `say_good_morning`
- インデントされた部分が関数の処理内容

「動詞＋目的語」の形にすると処理内容がわかりやすい

処理の内容	関数名	例
○○を追加する	add_○○	add_product（商品を追加する）
○○を削除する	remove_○○	remove_product（商品を削除する）
○○が存在するか	has_○○	has_product（商品が存在するか）
○○の状態か	is_○○	is_purchased（購入されたか）

関数を呼び出す（実行する）には、関数名を記述するだけ

```
# 関数を定義
def say_good_morning():
    print("おはようございます！")

# 関数を呼び出す
say_good_morning()
```

- 関数名の後に`()`をつけて呼び出す

2つの関数を定義して呼び出してみましょう

```
# 朝のあいさつを出力する関数を作成する
def say_good_morning():
    print("おはようございます！")
    print("昨日はよく眠れましたか？")
    print("今日も一日頑張りましょう！")

# 夜のあいさつを出力する関数を作成する
def say_good_evening():
    print("こんばんは！")
    print("今日も一日お疲れさまでした。")

# 朝のあいさつを出力する関数を呼び出す
say_good_morning()

# 夜のあいさつを出力する関数を呼び出す
say_good_evening()
```

11章 関数を書いてみよう

```
# 朝のあいさつを出力する関数を作成する
def say_good_morning():
  print("おはようございます！")
  print("昨日はよく眠れましたか?")
  print("今日も一日頑張りましょう!")

# 夜のあいさつを出力する関数を作成する
def say_good_evening():
  print("こんばんは!")
  print("今日も一日お疲れさまでした。")

# 朝のあいさつを出力する関数を呼び出す
say_good_morning()

# 夜のあいさつを出力する関数を呼び出す
say_good_evening()
```

```
おはようございます！
昨日はよく眠れましたか?
今日も一日頑張りましょう!
こんばんは!
今日も一日お疲れさまでした。
```

11章 まとめ

本章では以下の内容を学習しました

関数とは

- 一連の処理をひとまとめにして、何度でも再利用できるようにする仕組み
- 複雑なコードを1つにまとめられる
- 同じ処理を行うときに何度でも再利用できる

11章 まとめ（続き）

本章では以下の内容を学習しました

関数の作り方

```
def 関数名():
    一連の処理
```

関数の呼び出し方

- 関数名を記述するだけでよい（例：`say_good_morning()`）
- 関数は呼び出して初めて実行される