# 6.関数の作り方

C言語入門①

# 関数（かんすう）

- 特定の処理をまとめたもの
  - 何度も使う処理
  - 複雑な処理
  - など
- うまく利用することで効率的にプログラムを組むことができる
- 何度も呼び出すことができる

# 関数の要素

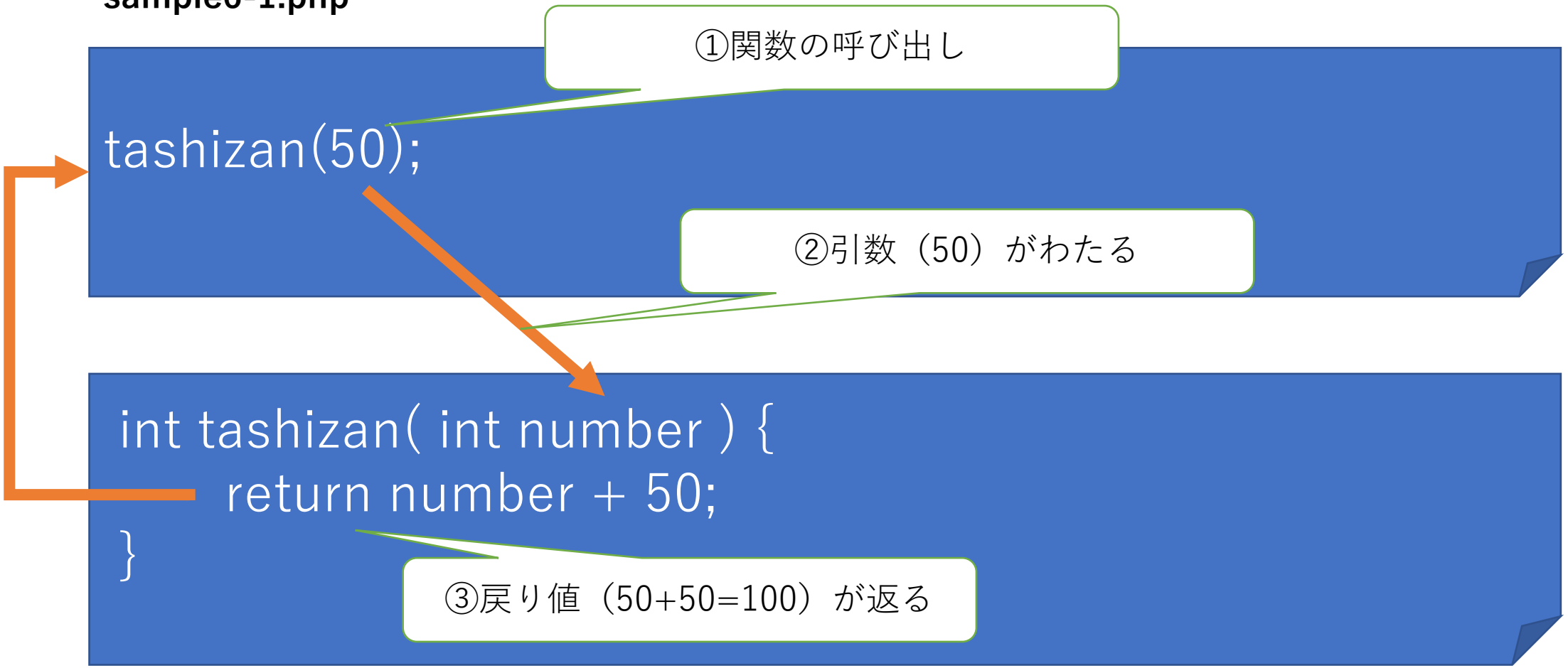
- **関数名**　…　関数の名前。
- **引数（ひきすう）**　…　関数に与えるパラメータ。省略可能
- **戻り値**　…　関数により得られる結果。省略可能
- sample6-1.c参照

# 関数の書式

```
(戻り値の型) (関数名) (引数1,引数2,…){
    処理
    return (戻り値)
}
```

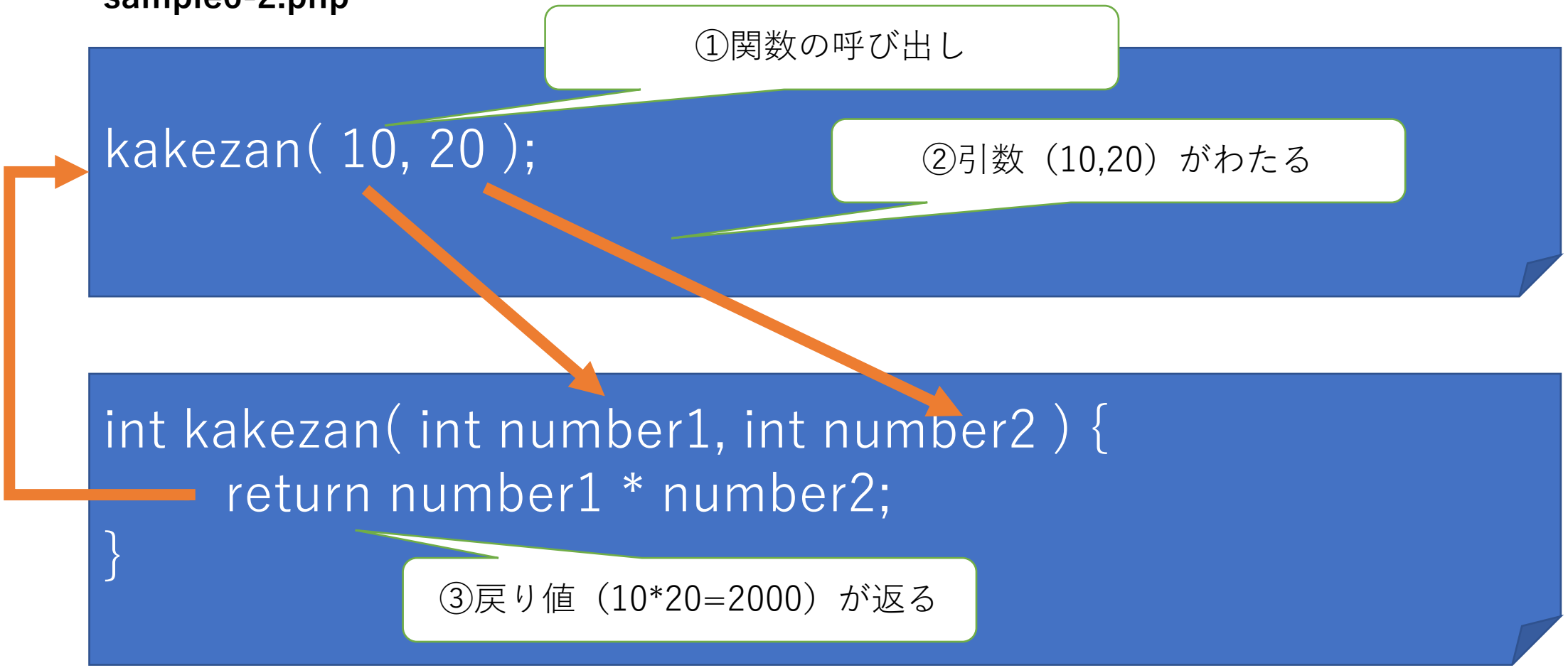
# 関数例①　…　引数１つ、戻り値あり

**sample6-1.php**



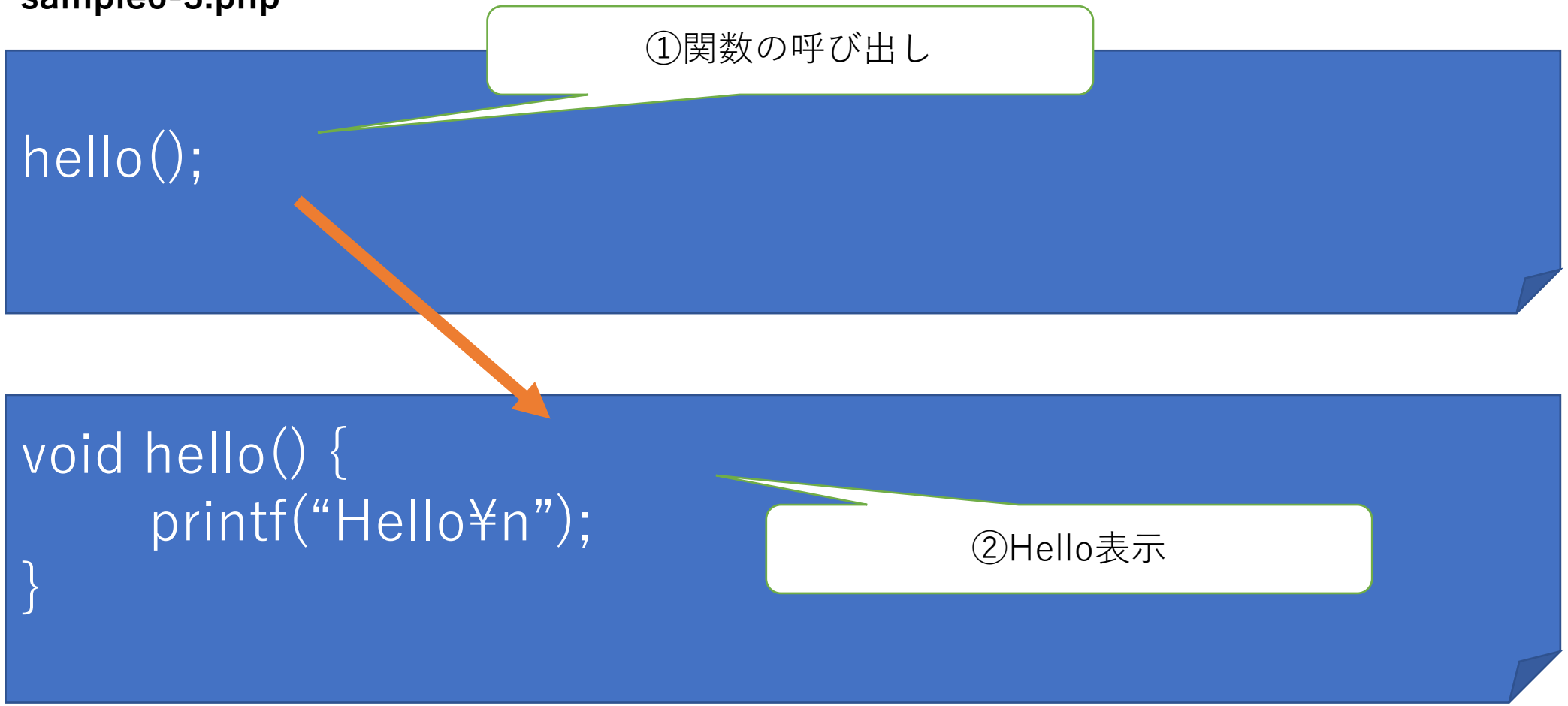
# 関数例②　…　引数２つ、戻り値あり

**sample6-2.php**



# 関数例③　…　引数なし、戻り値なし

**sample6-3.php**



# プロトタイプ宣言

- 呼び出す関数の型をあらかじめ予告しておくもの
- あらかじめ関数の型を決めておけば、そのあとで様々な場所で利用することが可能
- sample6-2.c参照

# void

- 戻り値の部分に書かれている**void(ボイド)**とは、関数の戻り値が無いということを意味しいる
- これらの関数は、戻り値を持たない
- 戻り値がない関数は、returnを省略することができる
- sample6-3.c参照

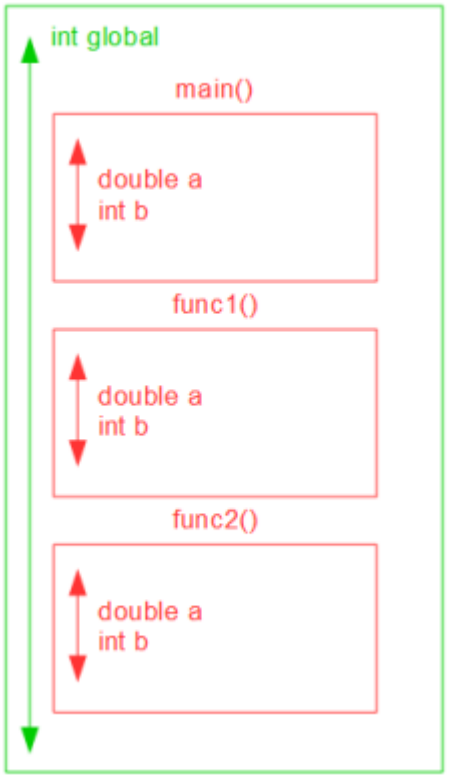
# ローカル変数とグローバル変数

- 関数内で定義されている変数を、**ローカル変数**と言う
- ローカル変数は、その変数が定義されている関数内のみで有効
- 関数の引数もローカル変数扱いとなる
- **グローバル変数とは**プログラムの先頭で定義されており、どこで呼び出しても同じものを指す
- sample6-4.c参照

# 変数のスコープ

- **変数の有効範囲（スコープ）**の違いは、定義されている場所に依存する
- ローカル変数は、定義されている関数が違えば、同名のものを定義しても、それぞれ別のものとして扱われる
- グローバル変数の場合は、プログラムぜんたい　でただ1つしかないので名前の重複は許されない

# 関数と、変数のスコープ


int global
main()
double a
int b
func1()
double a
int b
func2()
double a
int b