STAGE 1-4

基本的な演算

加減乗除と mod 演算

🚩 このSTAGEの目標

- 加減乗除の記法を知る
- モジュロ演算の原理を知る

加減乗除の記法

基本的な計算は以下のように行う

種類	書き方	例	結果
加法	A + B	1 + 3	4
減法	A - B	5 - 9	-4
乗法	A * B	19 * 3	57
除法	A / B	22 / 3	7

モジュロ演算 (割った余りを求める) は

mod	A % B	22 % 3	1

日本語キーボードなら

*	アスタリスク	Shift + け
/	スラッシュ	め
%	パーセント	5・え

加減乗除(式による演算)の注意

種類	書き方	例	結果
加法	A + B	1 + 3	4
減法	A - B	5 - 9	-4
乗法	A * B	19 * 3	57
除法	A / B	22 / 3	7
mod	A % B	22 % 3	1

コード

```c
int main(void) {
  int a = 1;
  int b = 3;
  int answer = a + b;
  printf("%d", answer);
}
```

printf("%d¥n", 変数の名前);
で、変数の値を確認できる

変数を使わなくても、
printf("%d", 計算式);
とすれば、計算式の答えが表示される
(計算式は、定数・変数のように扱ってもよい)

コード

```c
int main(void) {
  // 計算式を変数のかわりに
  printf("%d", 1 + 3);
}
```

加減乗除(式による演算)の注意

種類	書き方	例	結果
加法	`A + B`	`1 + 3`	4
減法	`A - B`	`5 - 9`	-4
乗法	`A * B`	`19 * 3`	57
除法	`A / B`	`22 / 3`	7
mod	`A % B`	`22 % 3`	1

コード

```c
int main(void) {
  // int型の1 + int型の3
  printf("%d", 1 + 3);
}
```

両辺の型が一致する場合、
結果はそれと同じ型になる

`int` + `int` = `int`

そうでない場合、両辺の型をくらべて、

1. 精度の高い小数 (`float` より `double`)
2. 表せる桁数の多い整数 (`int` より `long long int`)

の順番で優先される

小数点以下など、収まらない数字は切り捨てられる

`int` + `double` = `double`

`float` + `double` = `double`

加減乗除(式による演算)の注意

種類	書き方	例	結果
加法	A + B	1 + 3	4
減法	A - B	5 - 9	-4
乗法	A * B	19 * 3	57
除法	A / B	22 / 3	7
mod	A % B	22 % 3	1

コード

```
int main(void) {
  // int型の1 + int型の3
  printf("%d", 1 + 3);
}
```

数値の書き方を変えると、型が変わることがある

22 と整数で書くと int 型になり
printf("%d", 22)

22.0 と小数点以下を一桁でも書くと float 型になる
printf("%f", 22.0)

加減乗除(式による演算)の注意

種類	書き方	例	結果
加法	A + B	1 + 3	4
減法	A - B	5 - 9	-4
乗法	A * B	19 * 3	57
除法	A / B	22 / 3	7
mod	A % B	22 % 3	1

← 22÷3の結果は
7.3333… となるはずなのに 7 になっている

コード

```
int main(void) {
  // int型の22 ÷ int型の3
  printf("%d", 22 / 3);
}
```

加減乗除(式による演算)の注意

種類	書き方	例	結果
加法	A + B	1 + 3	4
減法	A - B	5 - 9	-4
乗法	A * B	19 * 3	57
除法	A / B	22 / 3	7
mod	A % B	22 % 3	1

コード

```c
int main(void) {
  // int型の22 ÷ int型の3
  printf("%d", 22 / 3);
}
```

両辺の型が一致する場合、
結果はそれと同じ型になる

`int` + `int` = `int`

で
22 も 3 も整数でしか書いていないので、
22 / 3 は
int型 ÷ int型 の割り算になっている

だから、結果も int 型になる

`int` 22 / `int` 3 = `int` 7

加減乗除(式による演算)の注意

種類	書き方	例	結果
加法	A + B	1 + 3	4
減法	A - B	5 - 9	-4
乗法	A * B	19 * 3	57
除法	A / B	22 / 3	7
mod	A % B	22 % 3	1

数値の書き方を変えると型が変わる

printf(“%d”, 22 / 3)

printf(“%d”, 22 / 3.0)

コード

```c
int main(void) {
  // int型の1 + int型の3
  printf(“%d”, 1 + 3);
}
```

なので

`int` 22 / `int` 3 = `int` 7

`double` 22.0 / `double` 3.0 = `double` 7.3333…

練習

コードを電卓代わりに使って計算してみよう

(1) 12 - 29 × 3
(2) 22 ÷ 3
(3) 57 を 19 で割ったあまり

例

コード

```
int main(void) {
  int answer = 12-29*3;
  printf( “答えは %d”, answer );
}
```

便利な加減乗除の記法

よくループで使う記法

`A = 5`

種類	書き方	例	表示される値 (式の評価値)	Aに代入される値
インクリメント (前置)	`++A`	`printf("%d", ++A);`	6	6
インクリメント (後置)	`A++`	`printf("%d", A++);`	5	6
デクリメント (前置)	`--A`	`printf("%d", --A);`	4	4
デクリメント (後置)	`A--`	`printf("%d", A--);`	5	4

前置型は、はじめに(計算を行う前に)Aを増減させる
後置型は、最後に(計算を行った後に)Aを増減させる
式の値を使う際は要注意