2回目　関数と参照・アインズ・ウール・ゴウン

✔☒ 第2回　関数と参照・オーバーロード

☒ 授業目標

- ✳ C++の関数定義・宣言・呼び出しを理解する
- 🔒const参照（const &）の使い方を理解する
- ☒ 関数のオーバーロードを使って柔軟な関数設計ができるようにする

☒ 関数の基本

C++の関数は、処理をまとめて呼び出せる仕組みです。
関数には「宣言」と「定義」があり、呼び出すときは名前と引数を使います。

```
#include <iostream>
using namespace std;

int add(int a, int b); // 宣言

int main() {
    int result = add(2, 3);
    cout << result << endl; // 5
}

int add(int a, int b) {
    return a + b;
}
```

☒ ポイント

- 関数宣言（プロトタイプ）は main より前に書く
- main のあとに本体（定義）を書くのが一般的

☒ const参照の基本

大きなデータをコピーせずに扱いたいときに便利！

```
void show(const string& name) {
    cout << "Hello, " << name << endl;
}
```

🔒const参照のメリット

- コピーが発生しない → 高速
- 値を変更できない → 安全
- 一時オブジェクトも受け取れる

種類	内容	特徴
値渡し	コピーして渡す	安全だが遅い場合もある
参照渡し	実体を直接操作	変更される可能性あり
const参照	読み取り専用参照	安全で効率的

⚙ 関数のオーバーロード

同じ名前で引数が違う関数を複数定義できる！

```
int area(int w, int h) { return w * h; }
double area(double r) { return 3.14 * r * r; }
```

⚠ 注意

- 戻り値の型だけ違ってもNG
- 引数の数・型・順序が異なる必要あり

⚠ const参照＋オーバーロードの組み合わせ

```
void print(string& s) { cout << "L-Value: " << s << endl; }
void print(const string& s) { cout << "Const:  " << s << endl; }
```

⚠ 実体（変数）ならL-Value版、一時文字列ならconst参照版！

⚠ 演習問題

1⚠ 2つの整数を足す関数 `add`
2⚠ 名前を受け取って挨拶する `greet(const string&)`
3⚠ 面積を求める `area` を整数／実数でオーバーロード

⚠ まとめ

- 関数は「処理をまとめる箱」
- const参照で安全・高速な引数渡し
- オーバーロードで同じ名前の関数を使い分け

Revision #2
★Created 8 October 2025 04:47:07 by youe2
Updated 15 June 2026 12:16:08 by youe2