•インクルードファイル

C言語: stdio.h

C++: iostream

どちらも標準入出力に関するヘッダファイルだが 名称が変更されている

• 名前空間の概念

C++にはstdという標準名前空間があり、 ほとんどの標準関数がこの名前空間で定義されている

namespace 〇〇というように名前空間を自分で作成して変数や関数を定義できる。名前空間が異なると、同じ変数名や関数名を使っても、異なる処理を書くことができる

### 名前空間の例

• using namespace 名前空間を設定可能

同じ b という変数名でも、名前空間毎に 違う値を格納することができる

#### を用いると、デフォルトの

```
namespace A {
  int b = 0;
namespace B {
  int b = 9;
using namespace A;
std::cout << b; // 0表示
using namespace B;
std::cout << b; // 9表示
```

### 名前空間の例

・名前空間を用いると、同じ変数名や関数名を複数 使用することができる(教科書p.70~73)

```
namespace A {
  int b = 0;
namespace B {
  int b = 9;
cout << A::b;
cout << B::b;
```

```
    ::「スコープ解決演算子」
    ■::△ と書くと
    ■は名前空間やクラス名
    △は変数や関数名
```

```
A::b の内容は 0
B::b の内容は 9
```

•ストリームの概念

C++: 画面表示、キー入力、ファイル入出力すべて ストリーム (プログラムやデータの流れ)と いう概念を使って記述する cout << "test"もストリームを使用

·文字列変数string

C++: 文字列を表現する際、文字型変数でなく string型を使用可能

•型推論

C++: intやdoubleといった明示的な型宣言を使用せず、auto宣言子を使って、変数を宣言したり、関数の戻り値に使用できる

例 auto a = 10; //aは整数型 auto b = 1.5; //bは実数型

・スマートポインタ

C++: Cのポインタから、より厳密なメモリ管理を 行えるようにしたポインタ(2年生で習う)

·bool型

C++: trueで真、falseで偽を表現できる

クラスとオブジェクト【重要!!!】C++: 構造体を発展させたもので、データだけでなく、データを処理する関数も内包できる

クラスは設計図にあたり、そこからオブジェクトを生成して値を設定したり、関数を 実行できる

### その他のC++関連キーワード

- •インスタンス
- •継承/ポリモーフィズム
- •カプセル化
- ゲッター/セッター
- •コンストラクタ/デストラクタ
- コンテナ(vector/list/イテレータ)
- テンプレート
- •オーバーロード

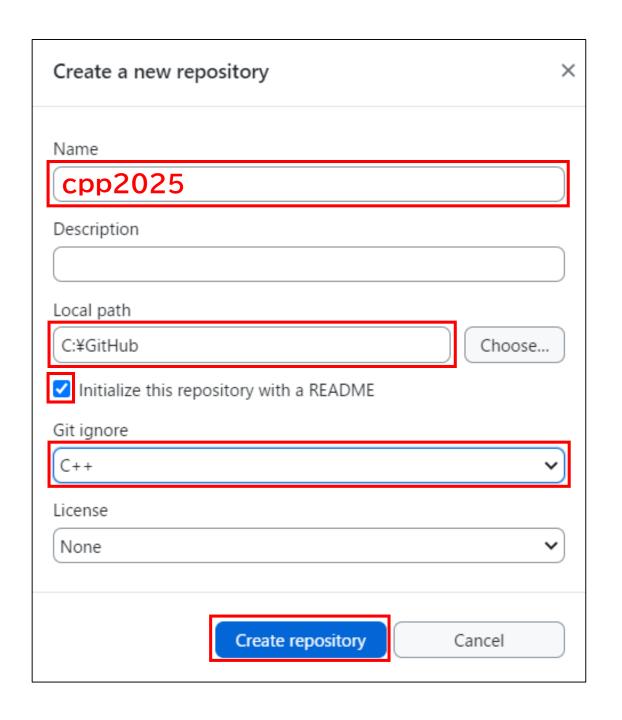
クラス関連

いきなりは覚えられないので、 講義中に出てきたときに解説します

• GitHubDesktopで C++用のリポジトリ作成

リポジトリ名はC言語と かぶらない名前にする ここではcpp2025

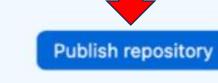
Git ignoreはC++を 選択しておく



• 右ペインに Publish repository ボタンが表示されているので、このボタンをクリック

#### Publish your repository to GitHub

This repository is currently only available on your local machine. By publishing it on GitHub you can share it, and collaborate with others.



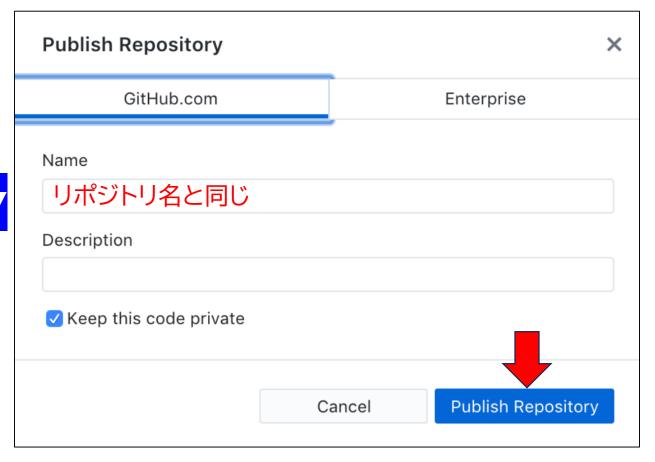
Always available in the toolbar for local repositories or # P

• Nameにリポジトリ名が 入っていることを確認 してから

#### Publish Repository

をクリック

以上で、C++ファイルを 保存する準備が完了



•課題ごとにフォルダを作成する

例えば教科書P.57のプログラムなら...

「Sample 201」というフォルダを作成して、その中に「main.cpp」を作成する

- •C++用フォルダにSample201フォルダを作成するにはコマンドプロンプトで mkdir Sample201 cd Sample201
- •main.cppを新規作成 copy nul main.cpp

フォルダ構成やファイルの場所はこのようになる C:¥GitHub¥cpp2025¥Sample201¥main.cpp

#### •main.cpp

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

int main(int argc, char** argv) {
   cout << "Hello World." << endl;
   return 0;
}</pre>
```

#### •main.cpp

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

```
main関数の引数は当分使用しないので引数は省略する
```

```
int main(int arge, char** argv) {
   cout << "Hello World." << endl;
   return 0;
}</pre>
```

#### main.cpp

```
Input/Output用インクルードファイル
#include <iostream>
                    の読み込み
using namespace std;
int main() {
   cout << "Hello World." << endl;</pre>
   return 0;
```

#### •main.cpp

```
#include <iostream>
using namespace std; 標準名前空間を利用する

int main() {
   cout << "Hello World." << endl;
   return 0;
}
```

•main.cpp(using namespaceを使用しない場合)

```
#include <iostream>

int main() {
    std::cout << "Hello World." << std::endl;
    return 0;
}

標準名前空間で定義されて
```

標準名削空间で定義されている命令等を利用する場合は 【名前空間::】を書く必要あり

#### •main.cpp

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

int main() {
    cout << "Hello World." << endl;
    return 0;
}
```

**cout** はコンソール画面への出力**endl** は改行("¥n"で代用可)

#### main.cpp

```
#include <iostream>
using namespace std;
int main() {
    cout << "Hello World." << endl;</pre>
    return 0;
```

main関数がint型なので戻り値の設定が必要

### main関数の戻り値

·main.cpp(参考)

int main() {

```
FILE* fp;
if (fp = fopen("data.txt", "r")) {
  fclose(fp);
else {
  cout << "File Open Error!" << endl;</pre>
  return -1;
cout << "File Open Success!" << endl;</pre>
return 0;
```

ファイルオープンに 失敗したら-1 成功したら0 を返すプログラム コマンドプロンプトで main.exeを実行後に

#### echo %errorlevel%

で戻り値を確認できる

#### C++でのコンパイル方法

•コンパイルの仕方

コマンドプロンプトで次のコマンドを入力する

C:\footnote{\text{EHsc main.cpp}}

clの後に「/EHsc」オプションを付ける必要があるので注意

コンパイルに成功したら main.exe を実行

•C++用フォルダにSample204フォルダを作成するにはコマンドプロンプトで

cd .. mkdir Sample204 cd Sample204

一つ上のフォルダへ戻るフォルダの作成作成したフォルダ内へ移動

•main.cppを新規作成 copy nul main.cpp

```
#include <iostream>
#include <string>
using namespace std;
               文字列変数stringの宣言
int main() {
   string s, t;
   t = "入力された文字列は「";
   cout << "文字列を入力:";
   cin >> s;
   cout << t + s << "」です。" << endl;
   return 0;
```

```
#include <iostream>
                         string型の変数には
#include <string>
                         代入演算子「=」を使って
using namespace std;
                         直接文字列の代入が可能
int main() {
   string s, t;
                         ※C言語ではstrcpyが必要
   t = "入力された文字列は「";
   cout << "文字列を入力:";
   cin >> s;
   cout << t + s << "」です。" << endl;
   return 0;
```

```
#include <iostream>
                         string型の変数では
#include <string>
                         加算演算子「+」を使って
using namespace std;
                         文字列の連結が可能
int main() {
   string s, t;
                         ※C言語ではstrcatが必要
   t = "入力された文字列は「";
   cout << "文字列を入力:";
   cin >> s;
   cout << t + s << "」です。" << endl;
   return 0;
```

```
#include <iostream>
#include <st 他にもstring型の変数では using namesp 比較演算子
int main() {
    string s = , !=, <, >
    t = "入力
    cout <<
           を文字列同士の比較に使用できる!
    cin >> s
    cout <<
            ※C言語ではstrcmpが必要
```