
最終更新:2019年度

第1回C++輪講

Intro

C++で画像処理ができるようになることが目標（わからないところは先輩に聞こう！！）

・デバッグのやり方を身に着ける

・命名規則を守る

・アルゴリズムをプログラムに落とし込む

Cや難しいと思われる項目はキーワードを示すのみ

Windows 10，VisualStudio2019，OpenCV4.xの環境で説明する

Chrome推奨

第1回の内容

- 第1回課題

C++とは

C++の特徴

- Cとの互換性
- 実行速度重視，可読性重視，再利用性重視など様々な書き方ができる

C++の規格

規格制定日	非公式名称
1998年	C++98
2003年	C++03
2011年	C++11
2014年	C++14
2017年	C++17
2020年予定	C++20

基本的にバージョンアップしても互換性がある
Visual Studioの対応状況は確認すること
コンパイラの実装状況 - cpprefjp C++日本語リファレンス

参考

- C++によるプログラミングの原則と実践 (アスキードワンゴ) Bjarne Stroustrup (著), 江添 亮 (監修), 株式会社クイープ/遠藤美代子 (訳) 単行本: 1248ページ 出版社: KADOKAWA 言語: 日本語

ISBN-10: 4048930516 ISBN-13: 978-4048930512 発売日： 2016/9/2

- cpprefjp - C++日本語リファレンス

Hello World

「Hello, World!」という文字列を出力するプログラムを書く
main.cppファイルを作成し以下を記述

```
#include <iostream>
int main()
{
    std::cout << "Hello, World!\n";
    return 0;
}
```

保存後，コンパイル，実行する

- コンパイルとはコンパイラを通してコードを変換すること
- C++ソースコード → C++コンパイラ → オブジェクトコード

実行結果

```
Hello, World!
```

解説

- `#include <iostream>` std::coutを使うための標準ライブラリ読み込み
- `int main()` int型のmain関数C++ではmain関数から実行される
- `std::cout << "Hello, World!\n";` 文字列の出力 `\n`はエスケープ文字の一種で改行を行う
- `return 0;` main関数の戻り値が0
  0は正常終了を表す

型とは

型	bit	概要
bool	1bit	boolern(true or false)
int	32bit	整数
double	64bit	浮動小数点数
std::string		文字列

その他　　第二回で説明

入力した値，文字列の表示

```
#include <iostream>
#include <string>
int main()
{
    int seisu;
    std::cout << "整数を入力してください" << std::endl;
    std::cin >> seisu;
    std::cout << "入力された整数:\t" << seisu << std::endl;

    double shousu;
    std::cout << "小数を入力してください" << std::endl;
    std::cin >> shousu;
    std::cout << "入力された小数:\t" << shousu << std::endl;

    std::string mojiretu;
    std::cout << "文字列を入力してください" << std::endl;
    std::cin >> mojiretu;
    std::cout << "入力された文字列:\t" << mojiretu << std::endl;

    return 0;
}
```

インデントを確実に！
わかりやすい変数名を！

if文

```
#include <iostream>
int main()
{
    std::cout << "整数を入力してください" << std::endl;
    int num;
    std::cin >> num;
    if (num > 0)
    {
        std::cout << "numは正の数です\n";
    }
    else if(num == 0)
    {
        std::cout << "numは0です\n";
    }
    else
    {
        std::cout << "numは負の数です\n";
    }
    return 0;
}
```

```
```

```
#include <iostream>
int main()
{
    // 入力
    std::cout << "一つ目の整数を入力してください" << std::endl;
    int max;
    std::cin >> max;

    std::cout << "二つ目の整数を入力してください" << std::endl;
    int min;
    std::cin >> min;

    // 条件
    bool swap = max < min;
    if (swap)
    // if (max < min) // 同じ意味
    {
        int temp = min;
        min = max;
        max = temp;
    }

    // 出力
    std::cout << "max:" << max << "\tmin:" << min << std::endl;
    return 0;
}
```

演算子

比較演算子	概要
==	左右の値が等しい
!=	左右の値が等しくない
>=	左の値が右の値以上
>	左の値が右の値より大きい
<=	右の値が左の値以上
<	左の値が右の値より大きい

for文

```
#include <iostream>
int main()
{
    std::cout << "例1" << std::endl;
    for (int i = 0; i < 10; ++i)
    {
        std::cout << i << std::endl;
```

```
    }
    std::cout << "\n例2" << std::endl;
    for (int i = 5; i <= 10; ++i)
    {
        std::cout << i << std::endl;
    }
    std::cout << "\n例3" << std::endl;
    for (int i = 0, j = 0; i < 10; ++i, --j)
    {
        std::cout << i << ",\t" << j << std::endl;
    }
    return 0;
}
```

++をインクリメント演算子という

インクリメント演算子	概要
（++iの）++	前置インクリメント演算子
（i++の）++	後置インクリメント演算子
（--iの）--	前置デクリメント演算子
（i--の）--	後置デクリメント演算子

静的配列（vector）

```
#include <iostream>
#include <vector>
int main()
{
    // 配列の作り方と初期値の確認
    std::vector<int> ary(10, 5); //要素数10,初期値5
    for (int i = 0; i < 10; ++i)
    {
        std::cout << ary[i] << std::endl;
    }

    std::cout << "-----------------------" << std::endl;

    // 配列の作成と確認
    std::vector<int> ary2(10); // 10のところを変えてみましょう
    for (std::size_t i = 0; i < ary2.size(); ++i)
    {
        ary2[i] = i;
    }
    for (std::size_t i = 0; i <  ary2.size(); ++i)
    {
        std::cout << ary2[i] << std::endl;
    }
```

```
    return 0;
}
```

関数化

```
#include <iostream>
#include <vector>
void swap(std::vector<int> &num) { // このnumを引数という
    if (num[0] < num[1])
    {
        int temp = num[1];
        num[1] = num[0];
        num[0] = temp;
    }
}
int main()
{
    std::vector<int> num(2);

    // 入力
    std::cout << "一つ目の整数を入力してください" << std::endl;
    std::cin >> num[0];

    std::cout << "二つ目の整数を入力してください" << std::endl;
    std::cin >> num[1];

    // 入れ替え
    swap(num);

    // 出力
    std::cout << "max:" << num[0] << "\tmin:" << num[1] << std::endl;
    return 0;
}
```

- void型 戻り値が何もないことを表す型
- swap関数の引数にある& キーワード：「参照渡し」

const修飾子

const：変数，引数などに変更不可の制限をつけることができる修飾子
意図しないプログラムのミスを防止することができる¥

例1 ：変数の場合

```
#include <iostream>
int main()
{
    const int x = 3;
    const int y = x + 4;
```

```
    // x, y はconst化されたので値を書き換えられない
    x = 5;      // エラー
    y = 10;     // エラー

    return 0;
}
```

例2 ：関数の引数

```
#include <iostream>
#include <vector>

void expand(const std::vector<int> &vec, const int size);
int main()
{
    std::vector<int> vec{ 0, 1, 2, 3 };
    const int size = 10;

    expand(vec, size);

    return 0;
}
void expand(const std::vector<int> &vec, const int size)
{
    //const化されたのでvecを書き換えられない
    for(int i = 0; i < size; ++i)
    {
        vec.push_back(vec.size() + i);
    }
}
```

- const修飾子は参照渡しと共によく使われる

関数を別のファイルに書きたい

the1stCppLecture.hppファイルを作成

- .hppと.hについて
  .hは純粋なC言語で書かれたファイルであるべきなのでC++で書かれている場合は.hppとする

```
#ifndef THE1STCPPLECTURE_HPP
#define THE1STCPPLECTURE_HPP
#include <vector>
namespace name{
namespace t1cl{
    void swap(std::vector<int> &num);
}
}
#endif
```

`THE1STCPPLECTURE_HPP`は`THE1STCPPLECTURE_HPP__`と書く場合もある
`__THE1STCPPLECTURE_HPP__`のような前後に二重アンダースコアを付ける形式はシステム予約語であるため使用してはならない
`#pragma once`はVisual Studio特有のものであり，VSや他のコンパイラで期待通りに機能しない場合があるため非推奨である

- #ifndef #define #endifとは
  キーワード：「インクルードガード」

the1stCppLecture.cppファイルを作成

```
#include "the1stCppLecture.hpp"
// std::vector<int> &numの部分はswapの引数という
void name::t1cl::swap(std::vector<int> &num)
{
    int temp = num[1];
    num[1] = num[0];
    num[0] = temp;
}
```

main.cppファイル

```
#include "the1stCppLecture.hpp"
#include <iostream>
int main()
{
    std::vector<int> num(2);

    // 入力
    std::cout << "一つ目の整数を入力してください" << std::endl;
    std::cin >> num[0];

    std::cout << "二つ目の整数を入力してください" << std::endl;
    std::cin >> num[1];

    if (num[0] < num[1])
    {
        // 入れ替え
        name::t1cl::swap(num);
    }

    // 出力
    std::cout << "max:" << num[0] << "\tmin:" << num[1] << std::endl;
    return 0;
}
```

ファイル出力とpath

```
#include <iostream>
#include <vector>
#include <fstream>
int main()
{
    std::string outputFilepath = "output.txt";
    // std::string outputFilepath = "%userprofile%/documents/output.txt"; // こちらのパスに変えてみよう
    std::ofstream ofs(outputFilepath);

    for (int i = 0; i < 10; ++i)
    {
        ofs << i << "\n";
    }

    return 0;
}
```

- パス
  キーワード：「相対パス」「絶対パス」
- おまけ
  生文字列リテラル

第1回課題

以下のソースコードをデバッグし，ソースコードに修正内容をコメントせよ

ヘッダには適切にインクルードファイルを指定すること
全ての関数は名前空間name，t1clに設置すること
関数run()で全ての関数が実行できるようにすること
困ったときはエラーの対処法などを参考にすること

ファイル名は the1stCppLecture_name.cpp， the1stCppLecture_name.hpp とせよ

例:yasukawa（安川くんの場合）
the1stCppLecture_yasukawa.hpp

```
namespace name{     // 安川くんの場合 namespace yasukawa
namespace t1cl{     // tanlab the 1nd cpp lecture
    void helloWorld();
    void typeExample();
    void ifExample();
    void forExample();
    void vectorExample();
    void funcExample();
    void swap(std::vector<int> num);
    void foutPathExample();
    void run(int num = 0); //run関数のみ，ヘッダではこのように宣言すること
}
}
```

the1stCppLecture_yasukawa.cpp

```
// 安川くんの場合 nameがyasukawaとなる
// 課題1
void name::t1cl::helloWorld()
{
    cout << Hello, World!\n:
}
// 課題2
void name::t1cl::typeExample()
{
    std::cout << "整数を入力してください" << endl;
    int num;
    std::cin << num;
    if (num > 0)
    {
        std::cout << "numは正の数です\n";
    }
    if(num = 0)
    {
        std::cout << "numは0です\n";
    }
    else
    {
        std::cout << "numは負の数です\n";
    }
}
// 課題3
void name::t1cl::ifExample()
{
    std::cout << "一つ目の整数を入力してください" << std::endl;
    int max;
    std::cin >> max;
    std::cout << "二つ目の整数を入力してください" << std::endl;
    int min;
    std::cin >> min;
    if (max < min)
    {
        min = max;
        max = min;
    }
    std::cout << "max: << max << \tmin:" << min << std::endl;
}
// 課題4
void name::t1cl::forExample()
{
    for (int i, i < 10, ++i);
    {
        std::cout << i << std::endl;
    }
}
// 課題5
```

```
void name::t1cl::vectorExample()
{
    std::vector ary;
    for (int i = 0; i < ary.size(); ++i) {
        ary[i] = i;
    }
    for (int i = 0; i <= ary.size(); ++i) {
        std::cout << ary(i) << std::endl;
    }
}
// 課題6
void name::t1cl::funcExample()
{
    std::vector<int> num(2);

    std::cout << "一つ目の整数を入力してください" << std::endl;
    std::cin >> num[0];

    std::cout << "二つ目の整数を入力してください" << std::endl;
    std::cin >> num[1];

    swap(num[0]);

    std::cout << "max:" << num[0] << "\tmin:" << num[1] << std::endl;
}
void name::t1cl::swap(std::vector<int> num)
{
    int temp = num[1];
    num[1] = num[0];
    num[0] = temp;
}
// 課題7
void name::t1cl::foutPathExample()
    std::string outputFilepath = 'output';
    std::ofstream ofs(outputFilepath);
    for (int i == 0; i < 10; ++i);
    {
        ofs << i << "\n";
    }
}
// void name::t1cl::run(int num = 0); //ヘッダではこのように宣言すること
// run関数は課題対象ではないです
void name::t1cl::run(int num)
{
    switch(num){
    default:
    case 1:
        std::cout <<"run helloWorld"<< std::endl;
        t1cl::helloWorld();
        if(num != 0) break;
    case 2:
        std::cout <<"run typeTest"<< std::endl;
        t1cl::typeExample();
        if(num != 0) break;
    case 3:
```

```
            std::cout <<"run ifTest"<< std::endl;
            t1cl::ifExample();
            if(num != 0) break;
        case 4:
            std::cout <<"run forTest"<< std::endl;
            t1cl::forExample();
            if(num != 0) break;
        case 5:
            std::cout <<"run vectorTest"<< std::endl;
            t1cl::vectorExample();
            if(num != 0) break;
        case 6:
            std::cout <<"run funcTest"<< std::endl;
            t1cl::funcExample();
            if(num != 0) break;
        case 7:
            std::cout <<"run foutPathTest"<< std::endl;
            t1cl::foutPathExample();
    }
```