プログラミング演習第一回　令和4年10月5日

プログラミング演習

- C言語の基礎の習得
- 出来るだけプログラミング言語 の授業とあわせて受講のこと
- 成績は、レポート+小テストで評価を行なう

教科書:
定本 明解C言語 入門編 柴田望洋著
Soft Bank Publishing 2,200円

参考書:
解きながら学ぶC言語 柴田望洋著
Soft Bank Publishing 1,700円

C プログラミングの一般手順

1）ソースファイルの作成（編集）

エディターを用いてソースプログラムを書く。
ソースプログラムをソースファイルとして保存。

C 言語処理系では、ソースファイル名の最後に拡張子 .c をつける。例）my_first_program.c など。

2）ソースプログラムのコンパイル

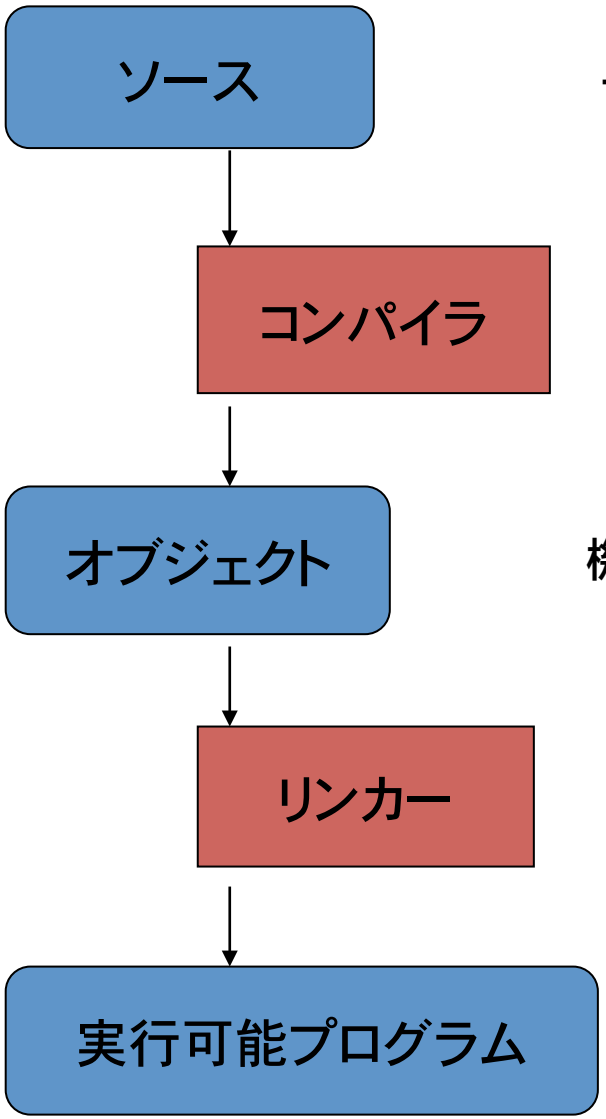
コンパイル途中でソースプログラムに文法の誤りがあれば、コンパイルエラーとなり、コンパイルは中断。

ソースプログラムの欠陥や過ちをバグ bug という。バグを見つけて修正する作業をデバッグ debug という。

バグが無くなるまでデバックを続ける。

文法の過ちによるバグとアルゴリズムのバグは別物であることに注意。

プログラム実行までの流れ



エディタでソースプログラムを作成

コンパイラにソースを入力してコンパイル

機械語に翻訳されたオブジェクトが生成される

標準ライブラリから必要なライブラリをオブジェクトに追加し、実行可能なプログラムを仕立てる作業をリンク(Link)という。
リンクを行うプログラムをリンカー(Linker)と呼ぶ。

開発環境
Microsoft Visual Studio

プログラムを完成させるには、ソースプログラムを作成し、それをコンパイルして実行可能ファイルを作成する。コンパイルとリンクは通常 1 つの過程と見なすことが出来る。通常はコンパイルとリンクを合わせて、単に「コンパイルする」という。

コンパイラとリンカーを合わせて、言語処理系と呼ぶ。

FORTRANでプログラムを書いて実行する為には、FORTRAN 言語処理系が、C 言語のプログラムをするには C 言語処理系が必要になる。

Visual Studioは言語処理系とソースプログラムを編集するためにテキストエディター、作成した実行可能ファイルを動かして期待した通りの動作を行うかのテストやうまく動作しない場合にどこで間違っているかを調べるデバッグ（プログラムの間違いを見つける）を行なうソフトが統合されている。

Visual Studio 2017によるC言語のプログラミング

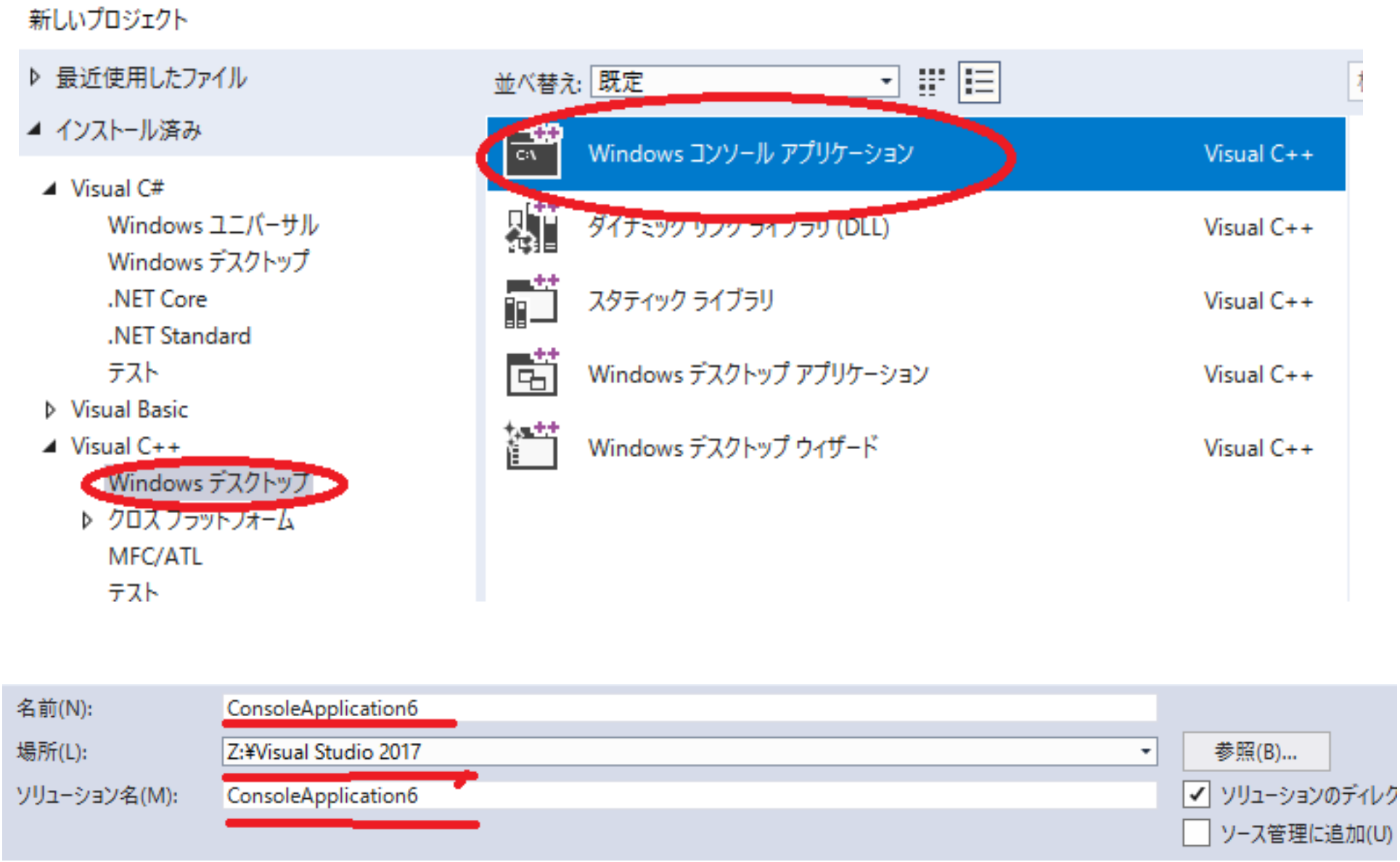
- ソースファイルを作成する前にプロジェクトを作成する。
- メニューバーの「ファイル」→「新規作成」→「プロジェクト」をクリックする。
- 「新しいプロジェクト」というダイアログが表示される
- インストール済みのテンプレートから

  「Visual C++」をクリックする。

  もし「Visual C++」が表示されなければ

  「その他の言語」をクリックし、「Visual C++」を選ぶ。
- 「Windows コンソールアプリケーション」を選択する。


スタート ページ - Microsoft Visual Studio
ファイル(F)
編集(E)
表示(V)
プロジェクト(P)
デバッグ(D)
ツール(T)
テスト(S)
分析(N)
ウィンドウ(W)
ヘルプ(H)
新規作成(N)
スタート ページ(E)
閉じる(C)
ソリューションを閉じる(T)
選択されたファイルを上書き保存(S) Ctrl+S
選択したファイルに名前を付けて保存(A)...
すべて保存(L) Ctrl+Shift+S
ソース管理(R)
ページ設定(U)...
印刷(P)... Ctrl+P
アカウントの設定(I)...
最近使ったプロジェクトとソリューション(J)
終了(X) Alt+F4
プロジェクト(P)... Ctrl+Shift+N
ファイル(F)... Ctrl+N
既存のコードからプロジェクトを作成(E)...
既存のコードからインテル(R) Fortran プロジェクトを作成...
開く
リモート バージョン コントロール システムからコ
か、ローカル ドライブで何かを開きます。
次からチェックアウト:
Azure DevOps
プロジェクト / ソリューションを開く
フォルダーを開く
Web サイトを開く

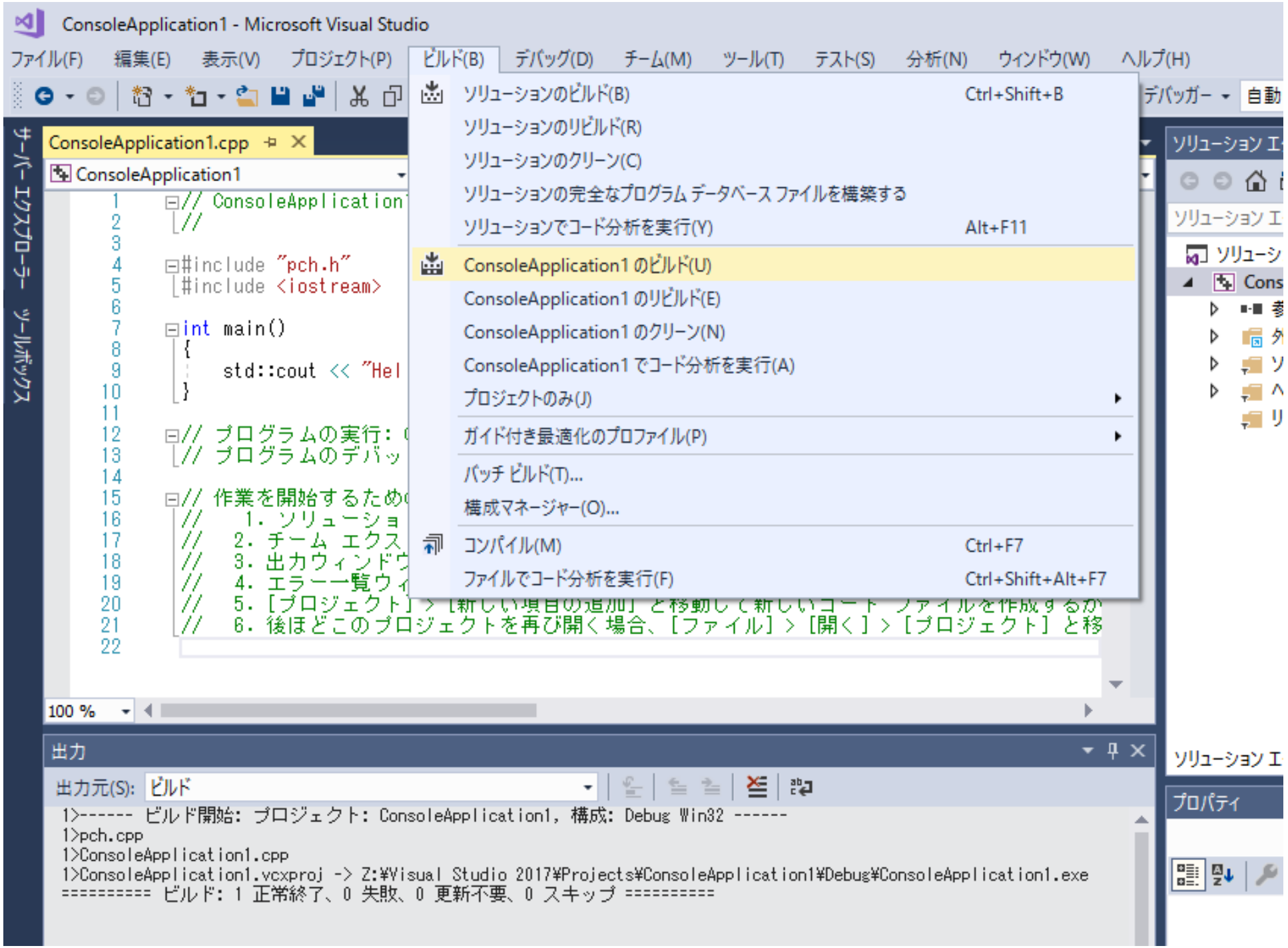

新しいプロジェクト
最近使用したファイル
インストール済み
Visual C#
Windows ユニバーサル
Windows デスクトップ
.NET Core
.NET Standard
テスト
Visual Basic
Visual C++
Windows デスクトップ
クロス プラットフォーム
MFC/ATL
テスト
並べ替え: 既定
Windows コンソール アプリケーション
Visual C++
ダイナミック リンク ライブラリ (DLL)
Visual C++
スタティック ライブラリ
Visual C++
Windows デスクトップ アプリケーション
Visual C++
Windows デスクトップ ウィザード
Visual C++
名前(N):
ConsoleApplication6
場所(L):
Z:¥Visual Studio 2017
参照(B)...
ソリューション名(M):
ConsoleApplication6
ソリューションのディレク
ソース管理に追加(U)

プロジェクト名の入力

- 名前 (N):　＜名前を入力してください＞
- 場所(L):　Z:¥Visual Studio 2017¥Projects
- ソリューション名(M):＜名前を入力してください＞

「新しいソリューションを作成する」が選択されている
場合には、標準でソリューション名はプロジェクト名と
同じに設定される。（違う名前にすることも可能）

1つのソリューションに複数のプログラム（プロジェクト）
を入れることもできるが、この演習では1つのプロジェクト
にする（その方がやりやすい）。


ConsoleApplication1 - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)
ソリューションのビルド(B) Ctrl+Shift+B
ソリューションのリビルド(R)
ソリューションのクリーン(C)
ソリューションの完全なプログラム データベース ファイルを構築する
ソリューションでコード分析を実行(Y) Alt+F11
ConsoleApplication1 のビルド(U)
ConsoleApplication1 のリビルド(E)
ConsoleApplication1 のクリーン(N)
ConsoleApplication1 でコード分析を実行(A)
プロジェクトのみ(J)
ガイド付き最適化のプロファイル(P)
バッチ ビルド(T)...
構成マネージャー(O)...
コンパイル(M) Ctrl+F7
ファイルでコード分析を実行(F) Ctrl+Shift+Alt+F7
ConsoleApplication1.cpp
ConsoleApplication1
出力
出力元(S): ビルド
1>------ ビルド開始: プロジェクト: ConsoleApplication1, 構成: Debug Win32 ------
1>pch.cpp
1>ConsoleApplication1.cpp
1>ConsoleApplication1.vcxproj -> Z:¥Visual Studio 2017¥Projects¥ConsoleApplication1¥Debug¥ConsoleApplication1.exe
========== ビルド: 1 正常終了、0 失敗、0 更新不要、0 スキップ ==========
プロパティ

プログラムの作成

- プログラムの編集画面が現れるのでそこにプログラムを入力。
- 赤字を入力

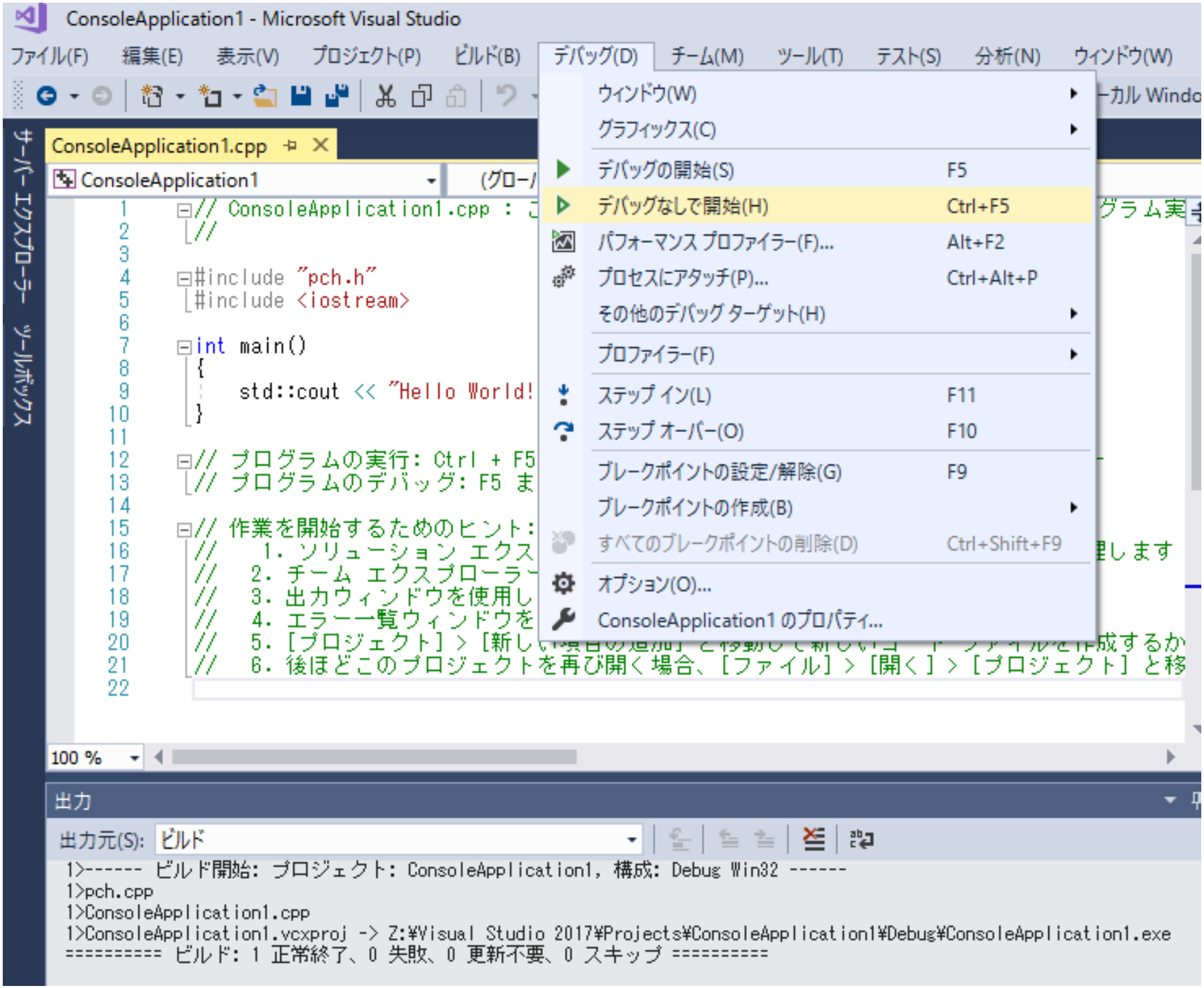
```
//

#include “pch.h”
#include <iostream>
int main()
{
   // std::out << “Hello World!¥n”;
  printf(“Hello World! ¥n”);
  return 0
}
```

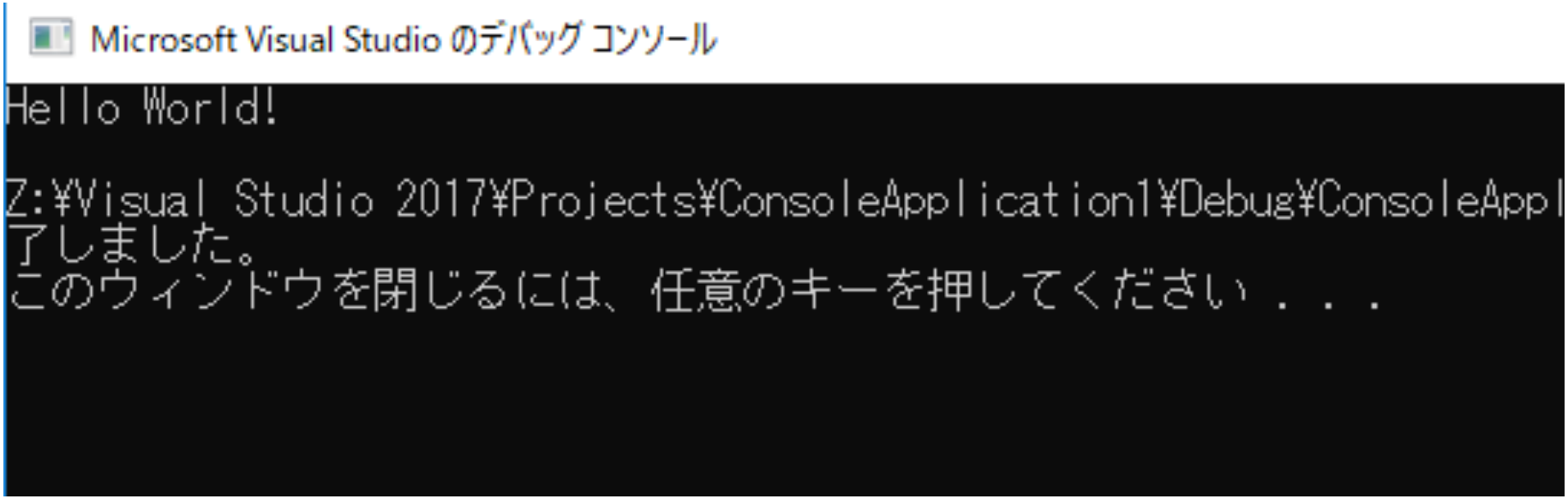
プログラムの実行

- コンパイルするには
  「ビルド」 → 「<名前>のビルド」
  または「ソリューションのビルド」

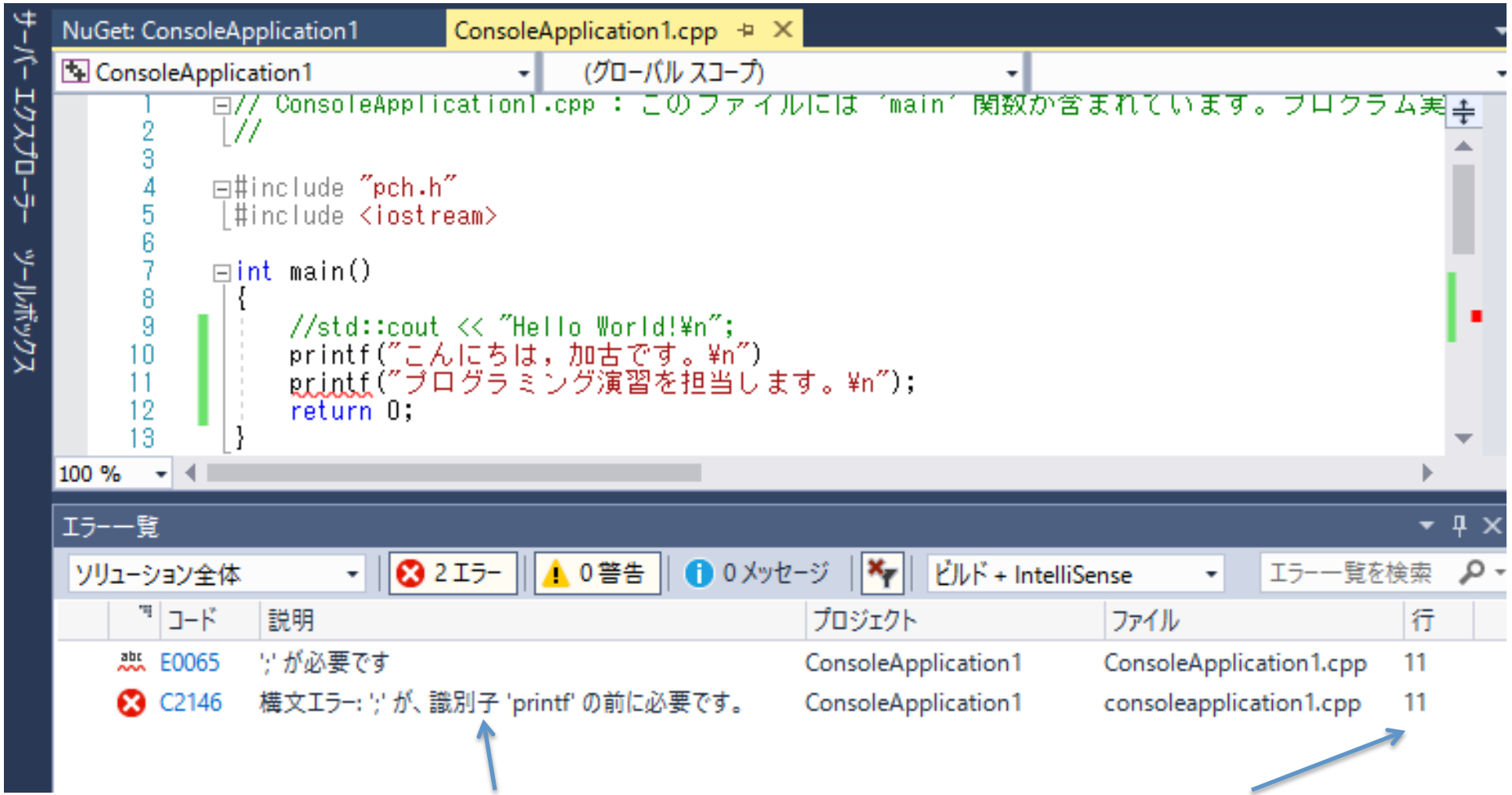
- 実行は
  「デバッグ」 → 「デバッグなしで実行」


ConsoleApplication1 - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W)
ConsoleApplication1.cpp
ConsoleApplication1
// ConsoleApplication1.cpp : こ
//
#include "pch.h"
#include <iostream>
int main()
{
std::cout << "Hello World!
}
// プログラムの実行: Ctrl + F5
// プログラムのデバッグ: F5 ま
// 作業を開始するためのヒント:
// 1. ソリューション エクス
// 2. チーム エクスプローラー
// 3. 出力ウィンドウを使用し
// 4. エラー一覧ウィンドウを
// 5. [プロジェクト] > [新しい項目の追加] と移動して新しいコード ファイルを作成するか
// 6. 後ほどこのプロジェクトを再び開く場合、[ファイル] > [開く] > [プロジェクト] と移
ウィンドウ(W)
グラフィックス(C)
デバッグの開始(S) F5
デバッグなしで開始(H) Ctrl+F5
パフォーマンス プロファイラー(F)... Alt+F2
プロセスにアタッチ(P)... Ctrl+Alt+P
その他のデバッグ ターゲット(H)
プロファイラー(F)
ステップ イン(L) F11
ステップ オーバー(O) F10
ブレークポイントの設定/解除(G) F9
ブレークポイントの作成(B)
すべてのブレークポイントの削除(D) Ctrl+Shift+F9
オプション(O)...
ConsoleApplication1 のプロパティ...
100 %
出力
出力元(S): ビルド
1>------ ビルド開始: プロジェクト: ConsoleApplication1, 構成: Debug Win32 ------
1>pch.cpp
1>ConsoleApplication1.cpp
1>ConsoleApplication1.vcxproj -> Z:¥Visual Studio 2017¥Projects¥ConsoleApplication1¥Debug¥ConsoleApplication1.exe
========== ビルド: 1 正常終了、0 失敗、0 更新不要、0 スキップ ==========

プログラムの実行結果の表示



プログラムに間違いがある場合

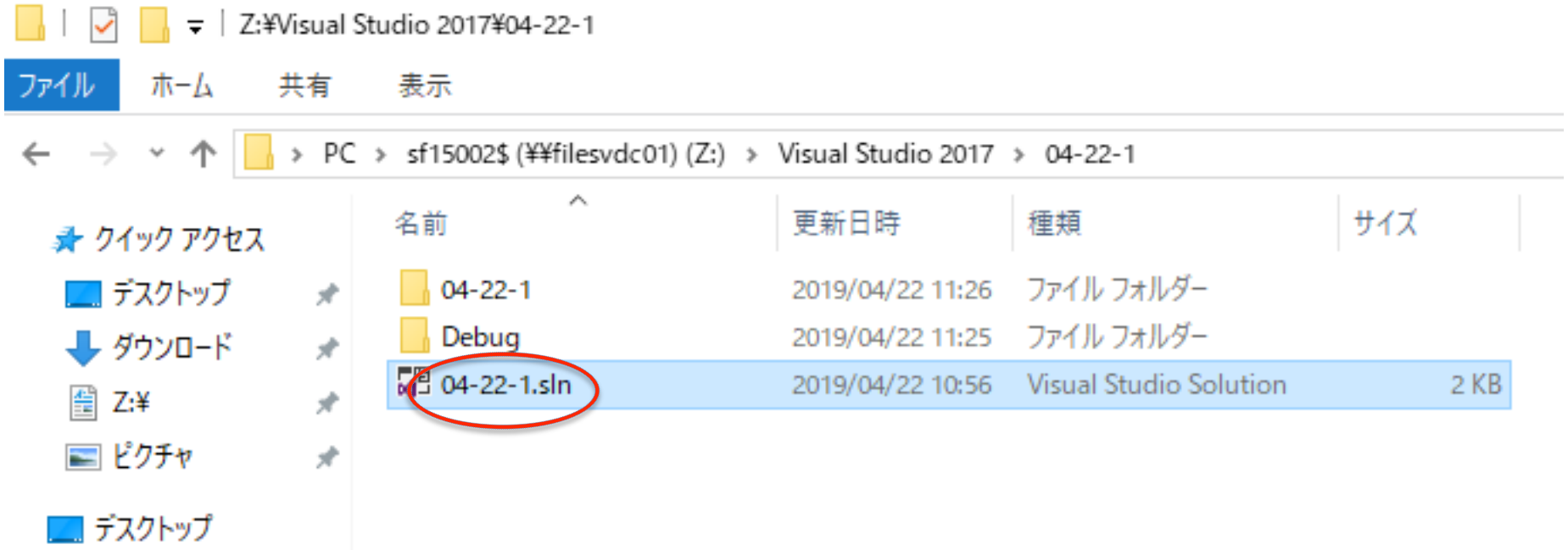


プログラムミスの内容

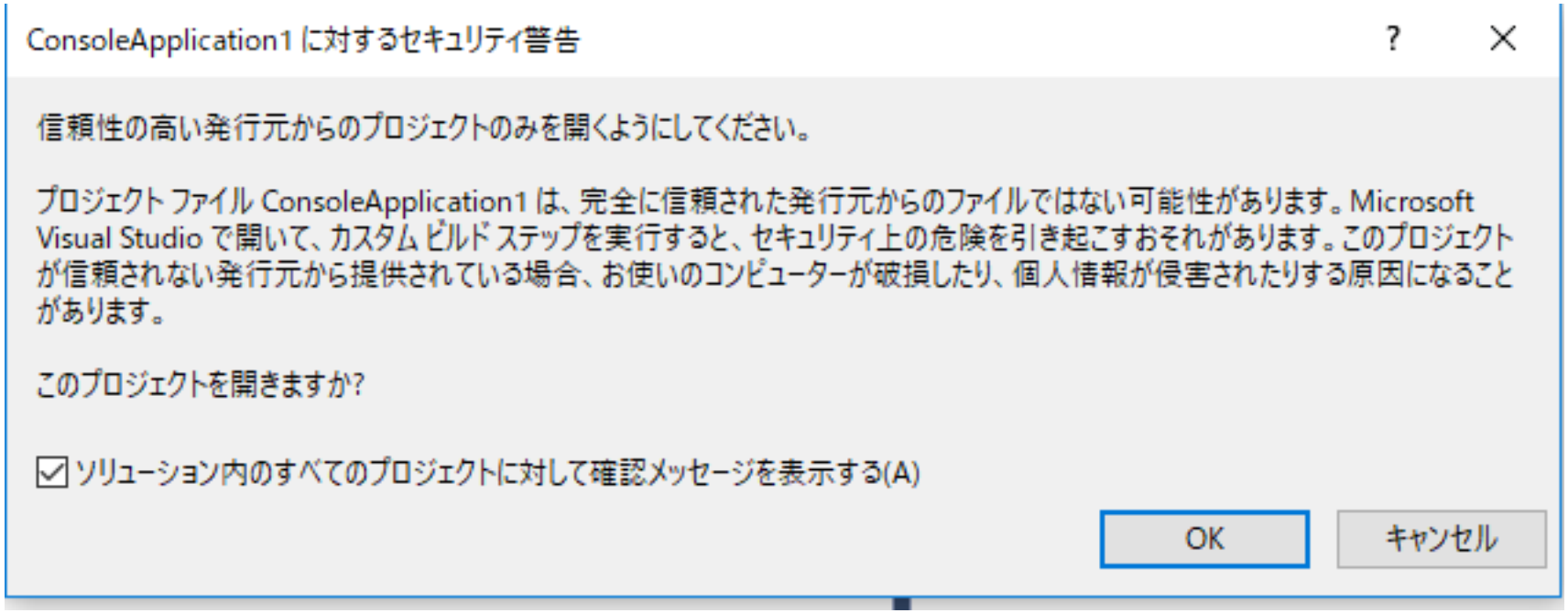
エラーが起こった（検出された）箇所
（実際のミスの箇所はこの前）

以前のプロジェクトを開く

以前に作成したプロジェクトを開くには、
(1)エクスプローラーでファイルを開きます。
(2)自分のホームディレクトリ → ドキュメント → visual studio 2017 → projects
　　のフォルダ内に作成したプロジェクトのフォルダが保存されています。
(3)プロジェクトのフォルダ内でVisual Studio のソリューションファイルをダブルクリック
　して開く。(.sln)



ネットワークドライブからのプロジェクトの読み込み



以前に作成したプロジェクトを開いた場合、上のようなメッセージが表示されることがあります。このまま「OK」を押して続行してください。

初めてのC言語

1)ソースプログラムの作成

エディタでプログラムを書き、sample.c というファイル名で保存

```
/* はじめてのプログラム */
#include “pch.h”
#include <iostream>
int main(int argc, char *argv[])
{
    printf(“Hello!￥n");
    return(0);
}
```

/* と */ で囲まれた部分はコメント文(注釈文)になる。コンパイラに無視される。

この部分は、C 言語のプログラムの(とりあえず)決まり切った型。

記号の読み方:

/ スラッシュ, * アスタリスク, # シャープ, \ バックスラッシュ
" ダブルクォーテーション, ; セミコロン, { } 中カッコ

C言語のプログラムの構成

C では関数を基本単位としてプログラムを構成する

単純なプログラムは main 関数のみから成る。

```
#include “pch.h”
#include <iostream>
int main(int ac, char *av[])
{
    文１
    文２
    …
}
```

pch.h というヘッダファイルをこの場所に挿入する

pch.h には標準的に使われる関数の定義が書かれている

main 関数内に書かれた文 (statement) は上から下へと順番に実行される

文はセミコロン ; で終わる

C言語のプログラムの構成

Visual Studio以外の開発環境やLinuxでのCのプログラミングでは、
プログラムは次のようになる。

```
#include <stdio.h>
int main(int ac, char  *av[])
{
    文１
    文２
    …
}
```

stdio.h という標準ヘッダファイルを挿入する。

stdio.h には標準的に使われる入出力関数等の定義が書かれている

<stdio.h>とヘッダファイル名を小なり記号＜と大なり記号＞でくくっているのは標準のヘッダファイルの挿入を意味する。
自分で作成したような標準でないヘッダファイルを挿入するには、文字列と同じように “pch.h”のように二重引用符 “ でくくる。

標準出力

```
#include “pch.h”
#include <iostream>
int
main(int ac, char    *av[])
{
    printf("Hello!¥n");
  return(0);
}
```

文字の列を文字列という

文字列をプログラム中で表現するには " " で囲む。これを文字列リテラルという

左の場合、"Hello!¥n" が文字列リテラル

¥n は特殊文字の一つで、改行を表す

文字列を表示するには printf 関数を用いる

printf 関数に文字列リテラルを引き渡すと、文字列が標準出力（画面）に表示される。printf 関数は標準ライブラリ関数の一つ。ヘッダファイル iostream (stdio.h) で定義されている

printf を用いた例

```
#include “pch.h”
#include <iostream>
int main(int argc, char *argv[])
{
    printf("Hello!¥n");
    printf("How are you?¥n");
  printf("I am fine. And you?¥n");
  return(0);
}
```

2 つのプログラムを実行すると、
まったく同じ結果を得る。

```
#include “pch.h”
#include <iostream>
int main(int argc, char *argv[])
{
    printf("Hello!¥nHow are you?¥nI am fine. Are you?¥n");
  return(0);
}
```

プログラムの記述形式

C 言語では原則としてソースファイルの自由な位置にプログラムを記述することが出来る（自由形式という）

```
#include “pch.h”
#include <iostream>
int main(int ac, char *av[])
{printf(“こういう書き方もあり¥n");return(0);}
```

```
#include “pch.h”
#include <iostream>
int
main(int ac, char *av[])
{
printf(
“こんなのもOKよ¥n"
);
return(0);
}
```

通常、プログラムを見易くするために段付け（インデント）をつける（タブキー）。

```
#include “pch.h”
#incude <iostream>
int main(int ac, char *av[])
{
  printf(“読み易くない？¥n");
  return(0);
}
```

コメント（注釈）

プログラムの説明等、実際に実行されない任意の文章をプログラムの中に入れる事が出来る。これをコメントという。プログラムの一部分を /*　*/ で囲ってコメント文にすることをコメントアウトという。また、//（スラッシュを二つ）を付けるとそれ以後、行の終わりまでコメントとして処理される。
/* … */で囲ってコメント文を入力する場合に、… の文章中に */ や /*の文字列が含まれないように注意する必要がある。

```
// これはコメント（行の終わりまで）

/* これもコメント
      加古富志雄　９９９９９９９ */

#include “pch.h”
#include <iostream>
int main(int argc, char *argv[])
{
  printf(“読み易くない？¥n");
  return(0);
}
```

特殊文字について

¥n：改行
¥r：先頭にカーソルを移動
¥t：タブ（タブ位置まで空白を出力）
¥b：一文字前に戻る（バックスペース）
¥'：シングルクォーテーション ’
¥"：ダブルクォーテーション ”
¥?：疑問符 ？
¥a：警報文字（alert）
¥¥：円記号

printfの文字列中では%記号は特別な意味を持ちます。
文字列中で％を使いたい時は％を二つ重ねて％％と
入力する。
%%：パーセント記号 %　(printfの文字列中に%を書く場合)

Microsoft Windows（日本語）では特殊文字を表すのに円記号¥を付ける。
LinuxやMacOSでは¥記号では無く、バックスラッシュ（逆斜線）＼ が使われる。

課題

1）標準出力に自分の氏名と学籍番号および簡単な自己紹介を表示するプログラムを作れ。

課題の提出はメールで行う。
宛先：kako@ics.nara-wu.ac.jp
件名には必ず「プログラミング演習」と付けてください。

注意1：メールの件名は間違わないようにしてください。
注意2：プログラムと実行結果の両方ををメールにコピーして送ってください。

メール作成 --- テンプレート選択 --- --- プロフィール選択 ---

Bcc表示 署名欄非表示 アドレス帳

宛先 kako@ics.nara-wu.ac.jp,

Cc

件名 プログラミング演習

本文

課題を提出します。

課題1

プログラム

```
//
// 99999999 加古富志雄

#include "pch.h"
#include <iostream>

int main()
{
    //std::cout << "Hello World!\n";
        printf("こんにちは，加古です。\n")
        printf("プログラミング演習を担当します。\n");
        return 0;
}
```

実行結果

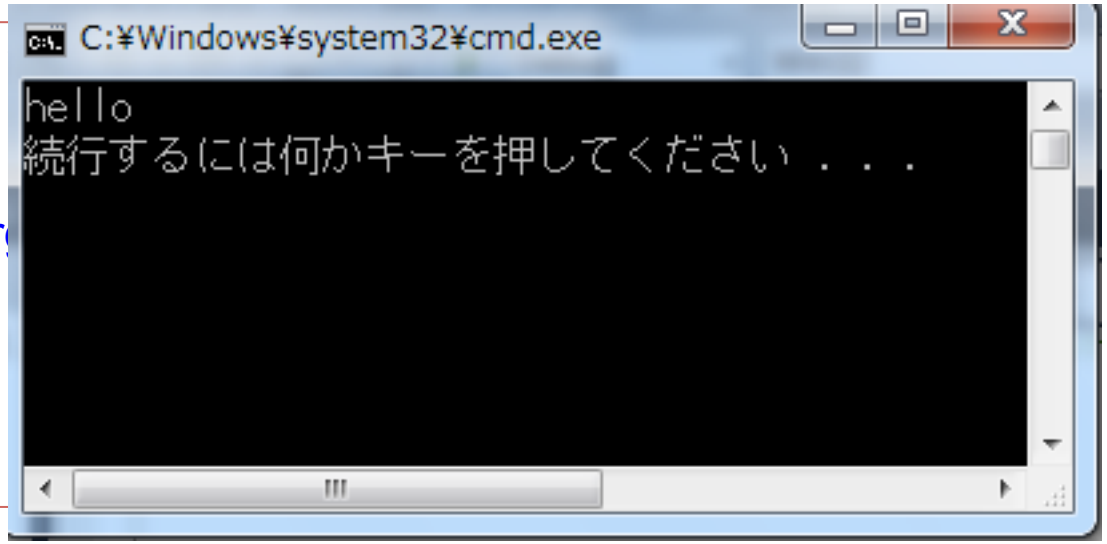
```
こんにちは，加古です。
プログラミング演習を担当します。

Z:\Visual Studio 2017\Projects\ConsoleApplication1\Debug\ConsoleApplication1.exe (プロセス
1168) は，コード 0 を伴って終了しました。
```

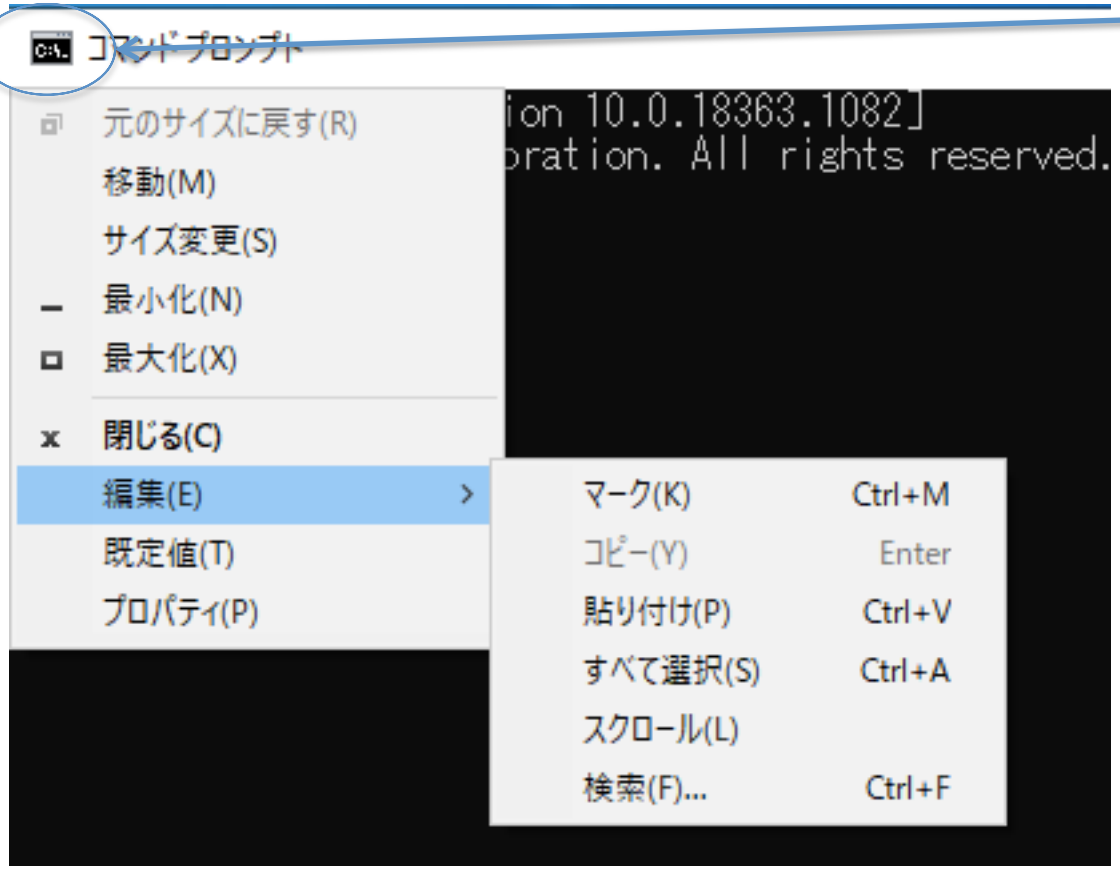
注意

0) Visual Studioのファイルメニューから「新規作成」→「プロジェクト」
→「Windowsコンソールアプリケーション」でプロジェクトを作成。
1)　Windowsでは ＼が ￥になる。
2)　プログラムの実行は、Visual Studioの「デバッグ」メニューから、
「デバッグなしで実行」をクリックすると、コマンドプロンプトの
画面が現れ、そこにprintfで出力した文字が表示されたあとに、
「続行するには何かキーを押してください」
と表示される。キーを押すとプログラムの実行は終了する.

```
/* はじめてのプログラム
#include “pch.h”
#include <iostream>
int main(int argc, char *ar
{
    printf("Hello! ¥n");
        return 0;
}
```



コマンドプロンプトの内容をコピーする方法



1）ここをクリック

2）編集（E)をクリック
サブメニューが現れるので、
(a) すべて選択
(b) コピー
を順にクリックすると内容がクリップボードにコピーされる。

3）メモ帳あるいはメールソフトで、
編集 => 貼り付け
を行えば、コピーされる。

Visual Studio Community 2022

マイクロソフトから無料で使用出来るVisual Studioが提供されている。
https://visualstudio.microsoft.com/ja/vs/community/
からダウンロード可能。

インストールする時にコンポーネントを選択する。
「C++によるディスクトップ開発」をインストールする。

C言語ではなく、C++言語がインストールされるが、これはC言語の拡張なので、これでC言語のプログラム開発が出来ます。

```
#/ ConsoleApplication1.cpp : このファイルには 'main' 関数が....
//
#include "pch.h"
#include <iostream>
int main()
{
        //std::cout << "Hello World!\n”;
        printf("hello\n");
}
```