内容

[**前書き　Visual Studioの使い方** 2](#_Toc510321047)

[**プログラムの作成手順** 2](#_Toc510321048)

[**・章末テスト** 7](#_Toc510321049)

[**Lesson 1 はじめの一歩** 8](#_Toc510321050)

[**1.1 C++のプログラム** 8](#_Toc510321051)

[**1.2 コードの入力** 8](#_Toc510321052)

[**1.3プログラムの作成** 8](#_Toc510321053)

[**1.Ex C言語の標準出力関数を知ろう(補足)** 9](#_Toc510321054)

[**Lesson 1 章末テスト** 9](#_Toc510321055)

[**Lesson 2 C++の基本** 10](#_Toc510321056)

[**2.1 画面への出力** 10](#_Toc510321057)

[**2.2 コードの内容** 10](#_Toc510321058)

[**2.3 文字と数値** 11](#_Toc510321059)

[**Lesson 2 章末テスト** 13](#_Toc510321060)

[**Lesson 3 変数** 14](#_Toc510321061)

[**3.3 型** 14](#_Toc510321062)

[**3.5 変数の利用** 14](#_Toc510321063)

[**3.5ex(補足) printf関数を使用して、変数の値を出力する。** 15](#_Toc510321064)

[**Lesson 3 中間テスト1** 15](#_Toc510321065)

[**3.6 キーボードからの入力** 16](#_Toc510321066)

[**3.6.ex(補足) C言語の標準入力関数scanf\_s** 17](#_Toc510321067)

[**3.7 定数** 17](#_Toc510321068)

[**Lesson 3 章末テスト** 17](#_Toc510321069)

[**Lesson 4 式と演算子** 18](#_Toc510321070)

[**4.1 式と演算子** 18](#_Toc510321071)

[**Lesson4 中間テスト１** 19](#_Toc510321072)

[**4.2 演算子の種類** 19](#_Toc510321073)

[**Lesson 4 中間テスト3** 23](#_Toc510321074)

# **前書き　Visual Studioの使い方**

## **プログラムの作成手順**

プロジェクトLesson0を作ってみましょう。大文字の入力はshiftキーを押しながらで行えます。

１. プロジェクトの作成

StartメニューからVisualStudioを起動する。



ファイル→新規作成→プロジェクトを選択



ドキュメントの下にc\_purapura\_1というフォルダを作って、そこにプロジェクトを作りましょう。\_(アンダースコア)はshift + /で入力できます。図１の赤枠を参照。

図１



2. プロジェクトのプロパティを変更する。

ソリューションエクスプローラーでプロジェクトを選択→右クリック→プロパティ



3.ソースファイルの追加

ソリューションエクスプローラーでプロジェクトを選択→右クリック→追加→新しい項目を選択。



4. ソースコードの入力

　追加したソースファイルを選択。



選択が出来たら、下記のようにコードを入力する。



次の４点に注意する。

①　スペースは半角。入力を忘れないように。図２，３の青枠。

　②　大文字、小文字に注意。

　③　#　、<　、（　、)、{ 、}、”(ダブルクォーテーション)などはShiftキーを押しながら入力する。図２，３の赤枠。

　④　;(セミコロン)を忘れない。上記コード、図２，３の緑枠。

⑤ 日本語入力→英字入力は半角/全角ボタンで切り替えることができる。図２の黄色枠。

**図２**



**図３**



5. ソースコードのビルド→実行

　Ctrl+F5でデバッガなしで実行できます。下記図の赤枠。



６. コマンドプロンプト(黒いウィンドウ)がすぐに消える

　　この現象が発生した場合は、2をやり直す。

## **・章末テスト**

下記のURLのテストを行いなさい。

<https://docs.google.com/forms/d/e/1FAIpQLSfWXuo1SqrWla4WEOIGEVUOJlqHg_vBP6p-5FZtuyqzRLNs5A/viewform?usp=sf_link>

# **Lesson 1 はじめの一歩**

## **1.1 C++のプログラム**

機械語(p.3)

　　→こんなの。



## **1.2 コードの入力**

統合環境を使ってみる(p.5)

本校の授業ではVisual Studio(統合環境)を使います。

テキストエディタにコードを入力する。(p.6)

　新しいプロジェクト、Lesson1を作成して、ソースファイルを追加して、p.7のSample1.cppの内容を入力する。

コンパイルエラーを起こしてみる

コードを書くことができて、実行出来たら、わざとコンパイルエラーを起こしてどうなるか確認してみましょう。

## **1.3プログラムの作成**

コンパイラを実行する(P9)

オブジェクトファイルを確認してみる。

オブジェクトファイルをリンクする(P9)

実行ファイルができていることを確認する。

## **1.Ex C言語の標準出力関数を知ろう(補足)**

printf関数

　coutはC++の標準出力関数ですが、printfはC言語の標準出力関数となります。C++はC言語のスーパセット言語となっており、C言語の機能はすべて使えます。では、下記のコードを入力してみてください。

|  |
| --- |
| #include <stdio.h>  int main()  {  printf(“ようこそ C++へ！”);  return 0; } |

C言語検定では標準出力関数として、printf関数が使用されていることと、ほかの言語でも似たようなprint関数が用意されてるので、こちらを使えるようになることも重要です。

## **Lesson 1 章末テスト**

下記のURLのテストを行いなさい。

<https://docs.google.com/forms/d/1NPbcFTvH37P5kJ3pHvOIB0GWXZoNYUG93aCOaEYAXYA/edit>

# **Lesson 2 C++の基本**

## **2.1 画面への出力**

新しいコードを入力する(p.16)

新しいプロジェクトLesson2を作成して、Sample.1.cppの内容を入力して実行してください。

色々な出力方法を知る(p.18)

Sample2.cppの内容を入力して実行してください。

改行には\nを使う

　テキストにも書かれていますが、重要です。しっかりと覚えましょう。

## **2.2 コードの内容**

main()関数

　main関数はエントリーポイントと呼ばれる特殊な関数です。プログラムのスタート地点となる関数です。

#include <iostream>をコメントアウトしてみる。

#include <iostream>をコメントアウトして、コンパイルしてみましょう。下記のようなエラーが表示されたと思います。



エラーをダブルクリックすると、エラーが起きている箇所にジャンプできます。試してみてください。

using namespace std;をコメントアウトしてみる。

こちらもコメントアウトしてコンパイルしてみてください。先ほどと同様のエラーが表示されたと思います。



重要

　エラーをダブルクリックするとコンパイルエラーが起きている箇所にジャンプできることをしっかりと覚えましょう。

## **2.3 文字と数値**

数値リテラル

　123、579、30.0など

文字リテラル

　‘A’、’B’、’c’など。**シングルクォーテーションで囲まれたもの。**一文字を表す。文字列とは違う！！！

文字列リテラル

　“ABC”、”ようこそC++”など。**ダブルクォーテーションで囲まれたもの。**

エスケープシーケンス

　全部を覚える必要はない。重要なのは下記の３点です。

　\n 改行コード

　\0 文字列の終わり(詳細はLesson9.7)

\マークをつけることで、\や’や”などの特殊文字を文字列に組み込むことができる。

　　→どういうこと？

例えば、下記のような文字列を表示する場合を考えてみましょう。



ダブルクォーテーションは文字列リテラルを囲むものという、特殊文字となっているため、下記のようなプログラムではコンパイルエラーになります。



文字としてダブルクォーテーションを出力したい場合は、下記のように記述する必要があります。



その他のエスケープシーケンスが必要になったら、ネットで検索をすればＯＫです。ググりましょう。

実習1 (時間 5分)

　一度のcoutの実行で下記のような表示ができるようにしてみよう。



printf関数でも同じ

　エスケープシーケンスのルールはprintf関数でも同じです。



実習２(時間 5分)

一度のprintf関数の実行で下記のような表示ができるようにしてみよう。



８進数

10を8とする表記法。

16進数

10を16とする表記法

## **Lesson 2 章末テスト**

下記のURLのテストを行いなさい。

<https://docs.google.com/forms/d/e/1FAIpQLSdNjkkwZ4M0L5_ujSSc-GULWBWXiA_dRPSHZ8ct2XpUrocsjg/viewform?usp=sf_link>

# **Lesson 3 変数**

## **3.3 型**

数が多くて、いきなり覚えるのは難しいと思いますが、今は下記の4点を覚えてください。

　整数型　 　　　　int

浮動小数点型 　　　　float

文字型 　　　　char

unsignedをつけると　 符号なしになる。(頭の隅の方にでも置いておいてください。)

|  |
| --- |
| int hoge = －10; //hogeは符号付き整数型。負数も記憶できる。  unsigned int hoge2 = 20; //hoge2は符号なし整数型。負数は記憶できない。 |

## **3.5 変数の利用**

変数に値を代入する

|  |
| --- |
| int num;  num = 3; |

＝が代入であることに注意してください。数学であればイコールですが、プログラムでは代入です。右辺の値を左辺に代入します。簡単に思えるかもしれませんが、実は、ここの理解で躓く人が多いです。

p.50のコードを入力して結果を確認しよう

　新しいプロジェクトLesson3を作成して、p.50のコードを入力して確認しましょう。

変数を初期化する

　変数は宣言することで、数値を記録するための領域がメモリ上に確保されます。では下記のようなコードの場合、どのような値が表示されるのでしょうか？

|  |
| --- |
| int num; //numという変数を用意する。  cout << num << “\n”; 何が表示される？ |

変数の値を変更する

　p.53のコードを入力して、動作を確認しましょう。

他の変数の値を代入する

　p.55のコードを入力して、動作を確認しましょう。

値の代入についての注意

　p.56のコードを入力して、動作を確認しましょう。

　doubleはfloatでもＯＫです。

## **3.5ex(補足) printf関数を使用して、変数の値を出力する。**

coutを使用して変数の中身を出力する方法は見てきましたが、printf関数も変数の値を出力することができます。int型の変数の値を出力する場合は下記のように記述します。



また、下記のように記述することで、複数の変数の値を出力できます。



## **Lesson 3 中間テスト1**

下記のURLのテストを行いなさい。

<https://docs.google.com/forms/d/e/1FAIpQLScDqY-iHav90m71Sr1uTCXsHrTW1t5TJLkW1oNA0xNZUAtfmg/viewform?usp=sf_link>

## **3.6 キーボードからの入力**

キーボードから入力する

最近使ったプロジェクトからLesson3を起動して、P.60のSample5.cppの内容を打ち込んで下さい。



C++標準入力関数 cin

cinを実行するとキーボードからの入力待ちになります。

二つ以上の数値を入力する

　p.61のSample6の内容を打ち込んでください。

## **3.6.ex(補足) C言語の標準入力関数scanf\_s**

C++の標準入力関数はcinですが、C言語にも同様の関数のscanf\_sがあります。下記のようなコードを入力することで、cinと同じ動作になります。入力して確認してください。



(注意)正確には、C言語の標準入力関数はscanf関数なのですが、この関数はセキュリティホールが存在しているため、マイクロソフトがより安全なscanf\_s関数を用意しています。使い方はscanfと同じなので、混乱しないようにしてください。

## **3.7 定数**

P.63のSample7.cppのコードを入力してみてください。

変更することのできない変数にconstをつける。

プログラマのミスで値を変更してしまうヒューマンエラーをなくす。

今、定数の利点を理解するのは難しいと思うので、これは後期に詳しくやります。

## **Lesson 3 章末テスト**

下記のURLのテストを行いなさい。

<https://docs.google.com/forms/d/e/1FAIpQLSfgUfWrA1P6vhlj-UqzQr3CBhHYmOxpzsFYOo0mdsmuQpSU9A/viewform?usp=sf_link>

# **Lesson 4 式と演算子**

## **4.1 式と演算子**

式の値を出力する(p.71)

新しいプロジェクトLesson4を作成して、Sample1.cppを入力して実行してください。

プログラムの世界では＊が掛け算になります。＊はshift＋：、と+はshift＋；で入力できます。下記の図の赤枠を参照。



色々な演算をする(p.71)

下記の２点をしっかりと意識しながら、Sample2.cppのコードを入力してください。

・ 変数と変数の演算を行える。

* 変数と数値リテラルとの演算を行える。

変数num1の値に１を足し、その値を再度num1に代入する(p.73)

　下記のコードに注目してください。

|  |
| --- |
| num1 = num1 + 1; |

数学的にはおかしな式です。 = の記号が等しいではなく、代入であったことを思い出してください。右辺の結果(num1+1)を左辺に代入しているため、このような記述が可能になります。

キーボードから入力した値を足し算する(p.74)

Sample3.cppの内容を入力して、実行してください。

実習 1(10分)

Lesson4の内容を改造して、下記のような表示をできるようにしなさい。

また、除算の演算子は / です。



## **Lesson4 中間テスト１**

下記のURLのテストを行いなさい。

<https://docs.google.com/forms/d/e/1FAIpQLSfyMQIL4RSN9mjOLdaeRnAJv2mDBi4U-WjkY4mywBdxOnw-uQ/viewform?usp=sf_link>

## **4.2 演算子の種類**

いろいろな演算子(p.76)

非常に多くの演算子があります。残念なことにここに記載されている演算子はすべて重要です。世のプログラマはすべてを覚えていると思います。でも、安心してください。覚えようとしなくてもプログラムを書いていると、自然と覚えてしまいます。

　まずは四則演算(＋－×÷)+αをマスターしましょう。

では、Lesson4を最近使ったプロジェクトから開いて、p.77のSample4.cppのコードを入力して、実行してください。

実習2(10分)

Lesson4を改造して、次の動画のような挙動になるプログラムを作成しなさい。

出席番号で０～４のグループに振り分けるプログラム。

<https://www.youtube.com/watch?v=lRwpa-nbEuI&feature=youtu.be>

インクリメント・デクリメント演算子(p.79)

これも四則演算子なのですが、少し変わっています。よく使う演算子なのでしっかりと覚えましょう。

インクリメント・デクリメントの前置と後置(p.80)

前置と後置で実行結果が変わることがあります。普段プログラムを書くときは、この規則を意識しなくてもいいように書く方が優れている場合がほとんどですが、資格・検定の試験でこれを問う問題がでることがあるので、覚えましょう。

Lesson4にp.81のSample.cppのコードを入力して、実行してください。

下記が前置と後置の挙動の違いの覚え方です。

**・前置なので、代入する前にインクリメント・デクリメントする。**

**・後置なので、代入した後でインクリメント・デクリメントする。**

ただし、これに依存するようなコードを書くことは可読性を下げることになるので、下記のようなコードを書くことを推奨します。

代入する前にインクリメントしたい場合

|  |
| --- |
| a++; 　//インクリメントしてから  b = a; //代入する。 |

代入した後でインクリメントしたい場合

|  |
| --- |
| b = a; //代入してから、  a++; //インクリメントする。 |

Lesson 4 中間テスト２

下記のURLのテストを行いなさい。

<https://docs.google.com/forms/d/e/1FAIpQLSdAUAv0FP5S5zUoNbznYQUhUVoGBl82IVA6wdh3LZrc0I-oRA/viewform?usp=sf_link>

代入演算子(p.83)

Lesson4にp.85のSample6.cppを入力して実行してください。

sizeof演算子

　この演算子は、配列(Lesson9)の要素数を調べるときなどに使用されることがあります。国家試験でもよく出てくる演算子です。例えば下記のように使います。

|  |
| --- |
| int num = sizeof(int); |

この演算子は読んで字のごとく、「size of int」int型 の(of) サイズ(size)を求めてくれます。

では、int型のサイズはいくつだったでしょうか？教科書のp.43に戻って確認してみましょう。

確認テスト

　下記のプログラムの実行結果を答えなさい。

|  |
| --- |
| int main()  {  int hoge = 0;  int size = sizeof(hoge);  cout << size << “\n”;  return 0;  } |

答え

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |
| --- |
| int main()  {  char hoge = ‘w’;  int size = sizeof(hoge);  cout << size << “\n”;  return 0;  } |

答え

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Hands-On 2

Lesson 4に下記のコードを入力して動作を確認しなさい。



シフト演算子(p.88)

下記の２点を覚える。

　・１ビット左にシフトすると値は倍になる。

　・１ビット右にシフトすると値は半分になる。

Hands-On 3

Lesson4に下記のコードを入力して動作を確認しなさい。



## **Lesson 4 中間テスト3**

下記のURLのテストを行いなさい。

<https://docs.google.com/forms/d/e/1FAIpQLScoCxdw4wk-kCYEglROeYF9pFD-tRO5MOZj9S-LSrc4pMY2Jw/viewform?usp=sf_link>