

プログラミング基礎 #07

基礎演習

担当：向井 智彦

先週のおさらい

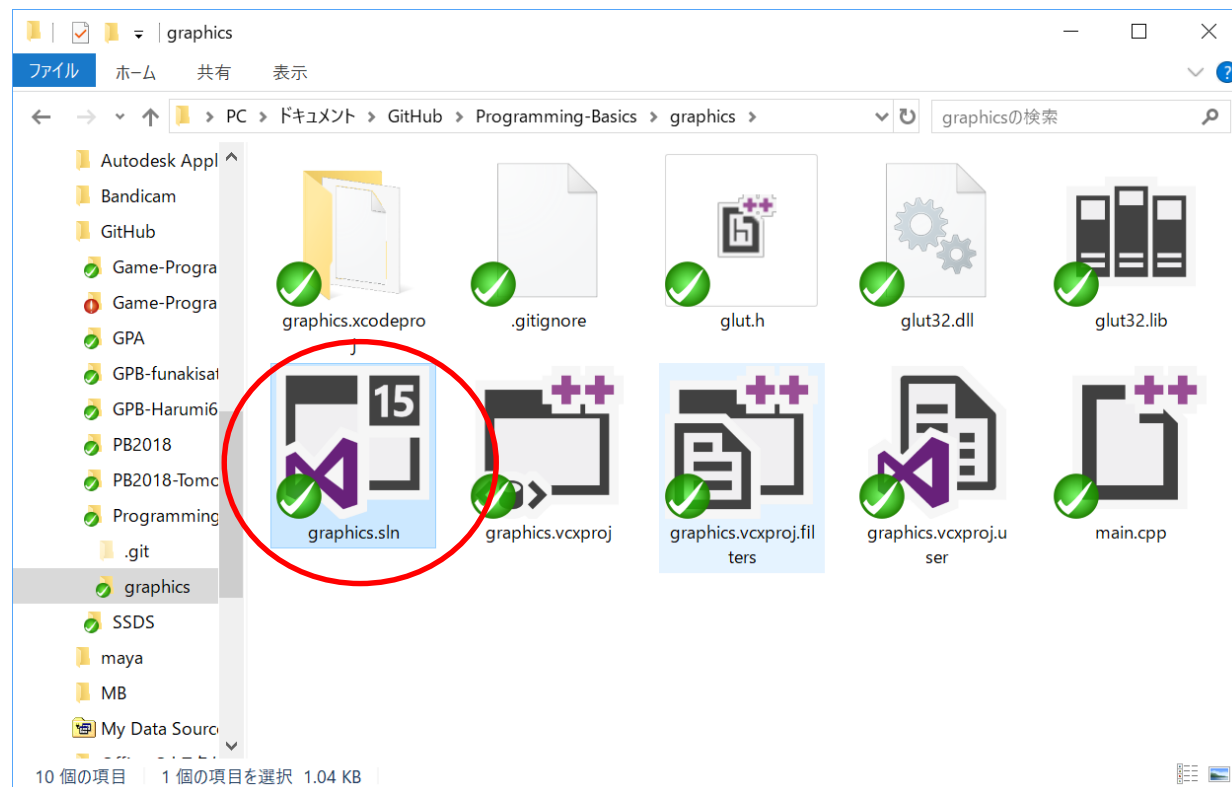
- 値渡しと参照渡し
 - 値渡し: 変数が持つ値を別の変数に代入
 - 代入先で変更が生じててもコピー元はそのまま
 - 参照渡し: 変数に別名を与える
 - 参照代入先で変更が生じるとコピー元にも影響
- ポインタ, 配列とポインタ
 - ポインタ: 変数に別名を与える方法2
 - ポインタが指す変数は変更可能(参照は変更不可)
 - 配列名 = ポインタ

本日の内容

- Github Desktopの利用
- Visual Studio 2017の利用
- グラフィックスアプリのサンプルプログラム
- 本日は課題なし
 - 未提出の課題がある人は、この機に復習を

Visual Studio 2017の起動

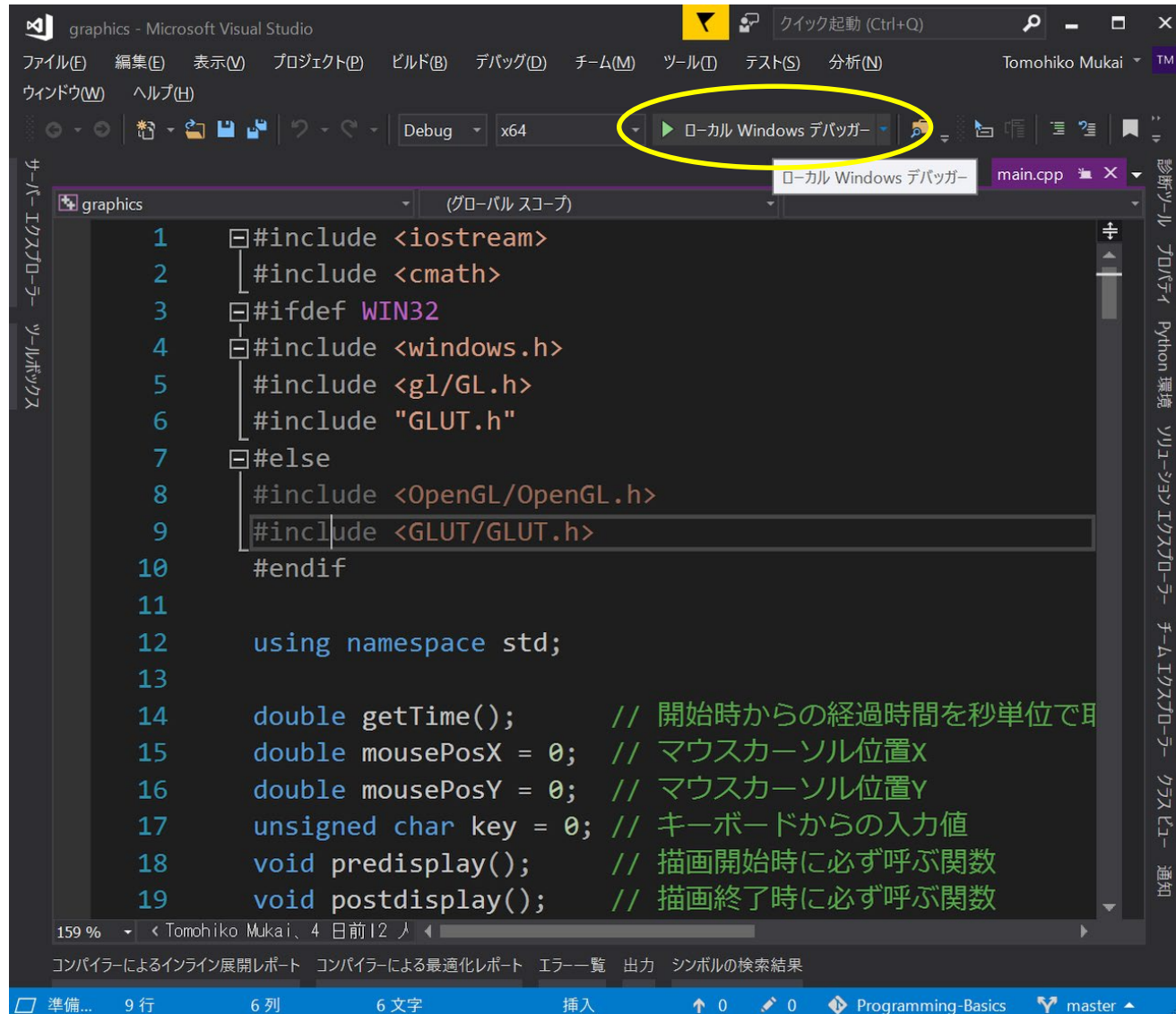
- 「graphics.sln」というファイルをダブルクリック
– 右肩に「15」と描いてあるアイコン



[表示]→[ソリューションエクスプローラー]



プログラムの実行



出力ウィンドウの読み方

The screenshot shows the Microsoft Visual Studio interface. The 'View' menu is open, and the 'Output' option is highlighted with a yellow circle. The 'Output' window is visible at the bottom, showing the output of a build process. The output text is as follows:

```
ビルド: 0 正常終了、0 失敗、1 更新不要、0 スキップ
```

Annotations in the image include:

- Green text: 始時からの経過時間を秒単位で取得する関数
- Green text: ウスカーソル位置x
- Green text: ウスカーソル位置y
- Green text: ーボードからの入力値
- Green text: 画開始時に必ず呼ぶ関数
- Green text: 画終了時に必ず呼ぶ関数
- Yellow text: エラーがある時には「1 失敗」になる
- Yellow text: 0個 正常終了、0個 失敗、1個 更新不要

The status bar at the bottom indicates '準備完了' (Ready) and '1 行 1 列 1 文字' (1 line, 1 column, 1 character).

エラー一覧の読み方

The screenshot shows the Microsoft Visual Studio interface. The 'View' menu is open, and 'エラー一覧 (I)' (Error List) is highlighted with a yellow circle. The code editor shows a C++ file named 'main.cpp' with the following code:

```
exit(0);  
}  
  
double time = getTime(); // 経過時間  
  
///// 2Dパート開始  
glDisable(GL_LIGHTING); // 消滅  
  
// 画面を横切る線  
//  
// 引数に線の太さを指定 (浮動小数点)  
glLineWidth(10.0);  
// 線の色: 引数は先頭から Red, Green, Blue  
glColor3d(1.0, 1.0, 1.0);  
// 線の描画  
glBegin(GL_LINES); // 線  
glVertex3d(-100.0, 0, 0); // 始点  
glVertex3d(100.0, 0, 0); // 終点  
glEnd(); // 線
```

The 'Error List' window at the bottom shows a single error:

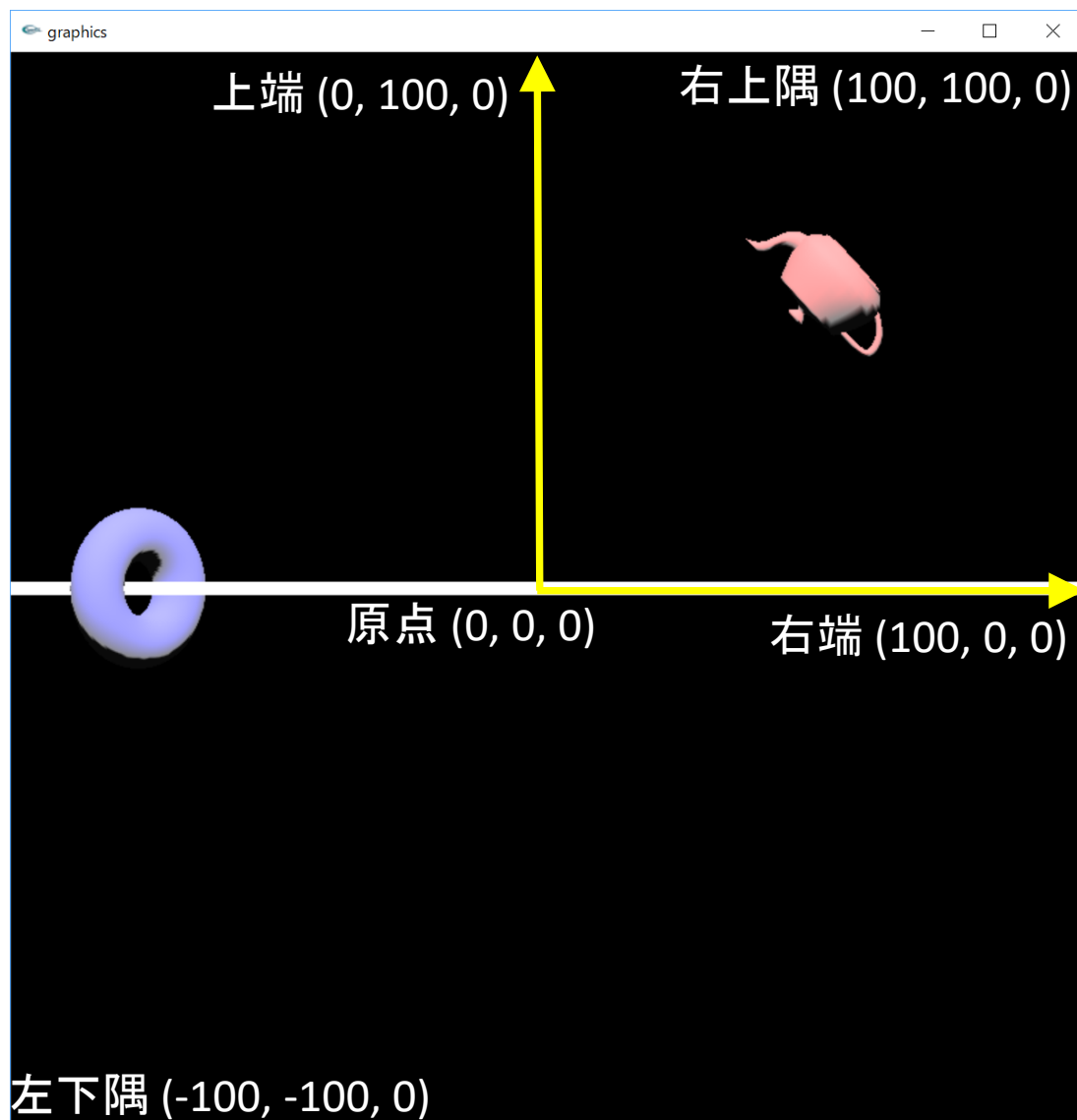
プロジェクト	ファイル	行	抑止
graphics	main.cpp	47	

エラー: C2146 構文エラー: ';' が、識別子 'glBegin' の前に必要です。

例: セミコロン忘れ

赤い×印がエラー（黄色▲は無視して良い警告）
ダブルクリックすると、エラーの箇所にジャンプする

グラフィックスアプリの座標系



編集するのはdisplay関数の中のみ

- キーボード入力処理
 - キーボード入力に応じて処理を条件分岐
- 2Dパート： 陰影表現を使わない部分
 - `glDisable(GL_LIGHTING);` ではじまる部分
 - 主に線や面を描く目的を想定
- 3Dパート： 陰影表現を用いる部分
 - 主に立体物を描く目的を想定

2Dパートでの線描例

```
// 引数に線の太さを指定(浮動小数)
```

```
glLineWidth(10.0);
```

```
// 線の色: 引数は先頭から[Red, Green, Blue]
```

```
glColor3d(1.0, 1.0, 1.0);
```

```
// 線の描画
```

```
glBegin(GL_LINES);
```

```
// 線描はじめ
```

```
glVertex3d(-100.0, 0, 0); // 始端 [x, y, z]
```

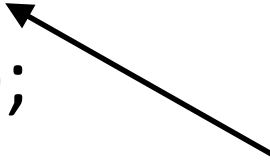
```
glVertex3d( 100.0, 0, 0); // 終端 [x, y, z]
```

```
glEnd();
```

```
// 線描おわり
```

次のコードを実行すると？

```
glBegin(GL_LINE_LOOP);  
glVertex3d( 80.0, -80, 0);  
glVertex3d( 80.0,  80, 0);  
glVertex3d(-80.0,  80, 0);  
glVertex3d(-80.0, -80, 0);  
glEnd();
```



指定した点の間に順番に線を描く。最後に指定した点と最初の点もつなぐ命令

次のコードを実行すると？

```
glBegin(GL_POLYGON);  
glVertex3d( 80.0, -80, 0);  
glVertex3d( 80.0, 80, 0);  
glVertex3d(-80.0, 80, 0);  
glVertex3d(-80.0, -80, 0);  
glEnd();
```

面を描く命令



次のコードを実行すると？

```
glBegin(GL_POLYGON);  
glVertex3d( 80.0, -80, -50);  
glVertex3d( 80.0, 80, -50);  
glVertex3d(-80.0, 80, -50);  
glVertex3d(-80.0, -80, -50);  
glEnd();
```

奥(z=-50)に移動させる

次のコードを実行すると？

```
glBegin(GL_POLYGON);  
glColor3d(1.0, 1.0, 1.0);    // 以降、白で描く  
glVertex3d( 80.0, -80, -50);  
glColor3d(1.0, 1.0, 1.0);    // 以降、白で描く  
glVertex3d( 80.0, 80, -50);  
glColor3d(1.0, 0.0, 0.0);    // 以降、赤で描く  
glVertex3d(-80.0, 80, -50);  
glColor3d(1.0, 0.0, 0.0);    // 以降、赤で描く  
glVertex3d(-80.0, -80, -50);  
glEnd();
```

基本プリミティブ(一部)

- ティーポット
 - `void glutSolidTeapot(double size);`
- 立方体
 - `void glutSolidCube(double size);`
- 四面体
 - `void glutSolidTetrahedron(void);`
- 球
 - `void glutSolidSphere(double radius, int slices, int stacks);`
- ドーナッツ
 - `void glutSolidTorus(double innerRadius, double outerRadius, int nsides, int rings)`