Class

- クラスとはオブジェクト内で使用されるデータやメソッド自体を、データ構造とプログラムで定義したものである。
- 基本的には構造体を拡張したものだが、主な特徴としては情報隠蔽機能がある。これはアクセス制御指定子としてpublicとprivateがあり、このアクセス権によってデータメンバを公開したり非公開にしたりする。
- オブジェクト生成と消滅時には特別な関数を使用する。
  - コンストラクタ(constructor):生成子： データメンバを初期化するときに使用
    1. コンストラクタはclass名と同一
    2. 関数の型は指定しない
    3. 戻り値はもたない
  - デストラクタ(destructor):消滅子： 終了時にメモリの開放等を行う。
    1. デストラクタはclass名と同じで、名前の前に~(チルダ)を付ける
    2. 関数の型は指定しない
    3. 戻り値はもたない
    4. 引数は無い。ただしvoidとはせずに、単に()とする

Classの定義

- クラスの定義: (クラス名:MyClass)

```
class  MyClass {
   private:
      int        data1;
      char       str[64];
   public:
      MyClass(void) { int = 0;}    //コンストラクタのプロトタイプ宣言
      int        func1(void){ return data1*2;}
      ~ MyClass( ) { };            //デストラクタのプロトタイプ宣言
};
```

メンバ関数をクラス(class)の内で定義する方法

- クラスの宣言:

```
MyClass       myObject;
```

（クラス名）　（オブジェクト名）

Classの定義

- クラスの定義: (クラス名:MyClass)

```
class  MyClass {
   private:
      int        data1;
      char       str[64];
   public:
      MyClass(void);    //コンストラクタのプロトタイプ宣言
      int        func1(void){return data1*2;}
      int        func2(void);
      ~ MyClass( );    //デストラクタのプロトタイプ宣言
};
 MyClass :: MyClass(void){ int = 0;}        //コンストラクタ
int MyClass ::func2(void){return data1*2;}
 MyClass ::~ MyClass( ){ }                  //デストラクタ
```

メンバ関数をクラス(class)の外で定義する方法

クラスの宣言:

```
MyClass       myObject;
```

（クラス名）　（オブジェクト名）

Classの設計の基本

クラスを利用してデータを隠蔽化する。
データを変更するには決められたメソッドを通じてのみ変更できる。

```
#include "stdafx.h"
#include <iostream>
class DataClass{
  private :
    int data;
  public :
    void setData( int x){ data = x; }
    int getData( void ){ return (data); }
    void printData(void){ cout << "data=" << data << endl; }
};
```

変数はprivate指定で隠蔽化

メソッドはpublic指定で公開（公開することでmainメソッドから、利用できるようにする。この他にprotectedがある）

```
int _tmain (int argc, _TCHAR* argv[ ]){
   DataClass  a;
   a.setData(1000);
   a.printData();
}
```

Classの使用例

```
//電卓(Calculator)クラスを利用してデータをセットし、和と差を計算する
#include "stdafx.h"
#include <iostream>
 class Calculator
{
   private :
      float  data1, data2;
   public :
      void setData(float x, float y){ data1 = x; data2 = y; }
      float Wa(void){ return( data1 + data2 ); }
      float Sa(void){ return( data1 - data2 ); }
};
 int _tmain (int argc, _TCHAR* argv[ ])
 {
   Calculator  calc;
   calc.setData(10,20);
   cout << " wa= " << calc.Wa( ) << endl;
   cout << " sa= " << calc.Sa( ) << endl;
   return 0;
 }
```

課題:
一度プログラムを実行して、動作を確認したら、掛け算と割り算を追加して表示させて下さい。

Classの例題

```
#include "stdafx.h"
#include <iostream>
#include   <stdlib.h>
#include   <time.h>
class Dice
{
   private :
         int i;
   public :
      int button(void){
         srand((unsigned)time(NULL) % RAND_MAX);
         i = rand() % 6 + 1;
         return i;
      }
};
int _tmain (int argc, _TCHAR* argv[ ]){
{
   Dice  a;
   cout << " サイコロの結果= " << a.button() << endl;
  return 0;
}
```

課題:
このサイコロクラスを流用して、3列のルーレットを作成しましょう。
（数字が3つ並べばOKです。）
結果例:
221