5.1-5.6

1.作用域是标识符的有效范围。有函数原型作用域、文件作用域，局部作用域、类作用域、命名空间作用域。

2.可见性是标识符是否可以引用的问题。标识符声明在前，引用在后。同一作用域中不能有同名标识符，不同作用域中，内层作用域的标识符会覆盖（hide）掉外层同名标识符。

3.

```
...
5 7
5 10
5 7
```

函数内y hide了全局变量y。

4.使用友元

```
class fuel;
class engine
{
    friend class fuel;
    //...
}
```

5.类的静态数据成员是带有static关键字声明的类数据成员，一个类只会保存一个该静态数据成员，所有类对象公用该数据。

6.类的静态函数成员是带有static关键字声明的类函数成员，静态函数成员只能直接访问类的静态数据成员，也可以直接由其他函数（比如main）调用（不需要依赖于对象）。一个类只会维护一个静态函数成员的拷贝，节省了性能。

5.8

友元函数是指某些虽然不是类成员却能够访问类的所有成员的函数。友元类的所有成员函数都是另一个类的友元函数，都可以访问另一个类中的隐藏信息（包括私有成员和保护成员）。他们都使用friend关键字来声明。

5.10

可以

```
class C
{
    private:
        static int a;
}
```

5.14

```cpp
#include <iostream>
using namespace std;

class Boat;
class Car;

class Boat
{
private:
  int weight;

public:
  Boat(int w) : weight(w){};
  friend int getTotalWeight(Boat, Car);
};

class Car
{
private:
  int weight;

public:
  Car(int w) : weight(w){};
  friend int getTotalWeight(Boat, Car);
};

int getTotalWeight(Boat b, Car c)
{
  return b.weight + c.weight;
}

int main()
{
  Boat b(100);
  Car c(200);
  cout << "Total weight is " << getTotalWeight(b, c) << endl;
  return 0;
}

```

5.15

普通局部变量在块结束时候被释放，静态局部变量会一直存在。每次调用函数时会创建一个普通局部变量，块结束时该变量被释放。而静态变量声明之后不会被释放，每次调用函数使用的均是同一个变量。

底层上：普通局部变量存储在栈空间，静态局部变量存储在静态数据存储区，因为栈的特性，程序运行需要不断进栈出栈，只要对应的函数（或块）运行结束，对应的局部变量就会出栈（被释放），而静态数据存储区不会存在进栈出栈这种情况，所以变量一直存在。