C++程序设计入门（上）

第四周编程作业

1.题目内容：（难度：易）
考虑创建一个绘图程序。需要有一个类 Screen 用来表示绘图时所用的屏幕
这个屏幕有一些基本属性，比如宽和高；有一些基本操作，比如获取屏幕的宽和高

1）. Screen 类有两个私有的 int 型数据成员，分别代表屏幕的宽和高
2）. Screen 类有两个构造函数
a.有两个整型参数，分别是屏幕的宽和高（以像素为单位）
b.有参构造函数将屏幕的宽和高存储在类的私有数据域中
Screen 类的有参构造函数
Screen 类的默认构造函数将屏幕宽和高分别设置为 640 和 480
c.Screen 类的所有构造函数均应输出字符串“screen”并换行
d.代码中的换行需使用 cout::endl
3）. 为私有数据成员提供 getter 和 setter 函数，如有必要，则增加其他数据成员及函数成员。函数原型如下

```
int getWidth();
int getHeight();
int setWidth(int width);    //return width
int setHeight(int height);  //return height
```

4）. 代码所用的主函数如下（不得做任何更改）：

```
int main() {
  int width, height;
  std::cin >> width >> height;
  Screen screen1 (width, height);
  Screen screen2;

  screen2.setWidth(800);
  screen2.setHeight(600);

  std::cout << screen1.getWidth() << ' ' << screen1.getHeight() << std::endl;
  std::cout << screen2.getWidth() << ' ' << screen2.getHeight();

#ifdef DEBUG
```

```
  std::cin.get();
#endif
  return 0;
}
```

输入格式:

两个由空格分隔的整数，代表屏幕的宽和高

输出格式：

两次调用构造函数所输出字符串，字符串后换行
两个不同屏幕对象的宽和高，由空格分隔，第一个屏幕对象的宽和高输出后换行

输入样例：

320 240

输出样例：

screen
screen
320 240
800 600

注意 ：上述输出一共 *4* 行，最后一行后面 **没有** 换行

2.题目内容：（难度：中）
为 Screen 类增加一个私有函数，用于检测屏幕的宽与高是否符合逻辑

基于本单元作业【1】，在 Screen 类中添加一个私有函数 exitWhenInvalidScreen 用于检测屏幕的宽与高是否 符合逻辑

1）. 函数 exitWhenInvalidScreen 的返回值类型、参数的个数和类型请你自行指定。

2）. 函数 exitWhenInvalidScreen 的判断逻辑如下：
a.宽度和高度均不得大于 1000 像素(可以等于 1000 像素)
b.宽度和高度必须大于 0 像素（不能等于 0 像素）
c.如果宽或者高不满足上述任一条件，则整个程序仅仅输出字符串"invalid screen size"，然后退出程序
3）. 在 Screen 类的有参构造函数及 Setter 函数中，要调用 exitWhenInvalidScreen 函数检测屏幕的宽和高

4）. 程序中的主函数如下（与作业【1】完全相同）

```
int main() {
  int width, height;
  std::cin >> width >> height;
  Screen screen1 (width, height);
  Screen screen2;

  screen2.setWidth(800);
  screen2.setHeight(600);

  std::cout << screen1.getWidth() << ' ' << screen1.getHeight() << std::endl;
  std::cout << screen2.getWidth() << ' ' << screen2.getHeight();

#ifdef DEBUG
  std::cin.get();
#endif
  return 0;
}
```

5. 提示

提示 1： exit() 函数可以强行退出程序，该函数在头文件 <cstdlib>中
提示 2: 函数 exitWhenInvalidScreen 可以设计为拥有两个参数，分别为宽和高。当仅需 判断宽或者高二者之一是否符合逻辑时，可以给另一个参数随便赋一个符合逻辑的数
提示 3: 当屏幕宽和高不符合逻辑时，仅输出规定的字符串，不要输出任何多余信息

输入格式:

空格分隔的两个整数，代表屏幕的宽和高

输出格式：

由输入的数据决定输出的内容。
有两种可能输出：

输出字符串"invalid screen size"。输出该字符串后不可以换行

或者

类似作业【1】的输出格式

输入样例 1：
320 2400

输出样例 1：
invalid screen size

输入样例 2：
320 240

输出样例 2：
screen
screen
320 240
800 600

3.题目内容：（难度：难）
在本单元作业【1】和作业【2】的基础上，创建一个 MyRectangle 类，并在 main 函数中创建类的实例。

Screen 类：
与作业【2】要求完全相同。
如果你的作业【2】顺利通过，那么你可以直接使用作业【2】中 Screen 类的代码

MyRectangle 类：
MyRectangle 代表的是一个矩形。我们用矩形左上角和右下角两个顶点的(x,y)坐标来表示它

1）. MyRectangle 类中的数据域有一个唯一与 Screen 类有关的成员，其类型为 Screen* 类型

```
Screen* screen_;
```

2）. MyRectangle 类的带参构造函数接受 5 个参数，其中前 4 个是整型参数
a.按照顺序，整型参数分别为矩形的左上顶点的 x1、y1 坐标，以及右下顶点的 x2、y2 坐标。（在构造函数中，不检查坐标有效性，也就是说，哪怕坐标出现负值，也不理会它。而是在后面的 Draw 函数中再做有效性检查）
b.按照顺序，第 5 个参数为 Screen 类的对象指针

```
//带参构造函数原型声明
MyRectangle::MyRectangle(int x1, int y1, int x2, int y2, Screen* screen);
```

3）. MyRectangle 类的默认构造函数将【左上----右下】对角线两个点的坐标均设置为原点坐标(0,0)

```
//默认构造函数原型声明
MyRectangle::MyRectangle();
```

4）. MyRectangle 类的所有构造函数均应使用 cout 输出字符串“myrectangle”并换行（使用 cout::endl）

5）. MyRectangle 类中应提供 setCoordinations()用于设置对角线的左侧及右侧顶点坐标；该函数共有 4 个形式参数。这些参数的含义及类型与“带参构造函数”的前 4 个参数相同。该函数将形式参数的值拷贝到类的私有数据域中。

6）. MyRectangle 类中应提供 setScreen(Screen& screen)用于设置该类的实例所对应的 Screen 对象；

a. 也就是说，setScreen 函数会将引用参数 screen 这个对象的地址赋给 MyRectangle 类中的私有成员 screen_。

b. 要注意：私有成员 screen_ 是对象指针类型，而 setScreen 的函数参数 screen 是对象引用类型

c. 所以，必须要取 screen 的地址，然后将该地址赋值给私有成员 screen_

d. 函数返回值类型由你自己决定

注： 如果你学有余力，可以尝试在这一步中，将函数原型变为 setScreen(const Screen& screen)，尝试解决编译错误（提示：需要修改 Screen 类）

7）. MyRectangle 类的 Draw()函数应检查坐标的有效性，确保矩形的顶点坐标是合理的、在前述屏幕的宽和高范围内是可见的（矩形框与屏幕框重合算是不可见、不合理）；

a. 如果上述坐标不合理，则在 Draw()中用 cout 输出“invalid myrectangle”然后换行(用 std::endl)；

b. 如果上述坐标合理，则在 Draw()中用 cout 输出矩形的左上顶点的 x、y 坐标以及矩形的宽度和高度（一共 4 个数值，任意两个相邻数值间以 1 个空格分隔；第 4 个数值后面没有空格），然后换行(用 std::endl)

8）. 如有必要，则增加其他数据成员及函数成员

main() 函数：

需使用如下 main()函数（不得更改）

```
int main() {
  int width, height;
  cin >> width >> height;
  Screen  screen (width, height);

  int leftX, leftY, rightX, rightY;
  cin >> leftX >> leftY >> rightX >> rightY;
  MyRectangle myRectangle1 (leftX, leftY, rightX, rightY, &screen);
  MyRectangle* myRectangles = new MyRectangle[2];
```

```
  myRectangles[1].setCoordinations(10, 300, 700, 500);
  myRectangles[1].setScreen(screen);

  myRectangle1.Draw();
  for (int i = 0; i < 2; i++) {
    myRectangles[i].setScreen(screen);
    (myRectangles+i) -> Draw();
  }

  delete [] myRectangles;

#ifdef DEBUG
  std::cin.get();
#endif
  return 0;
}
```

输入格式:
空格分隔的整数

输出格式：
字符串
或者
空格分隔的整数

输入样例：
800 600
30 20 300 200

输出样例：
screen
myrectangle
myrectangle
myrectangle
30 20 270 180
invalid myrectangle
10 300 690 200

4.题目内容：（难度：难）

1）本次作业在第 4 单元作业【1】的基础之上，修改而来；

2）在 Screen 类的构造函数中调用图形库的 initgraph()；

3）在 Screen 类的析构函数中调用图形库的 closegraph() （本项要求为选作）；

4）在 MyRectangle 类的 Draw()函数中调用图形库的 rectangle()函数绘图；

5)必要的地方，将原程序中的 cout 语句以图形库中的 outtextxy()或者 xyprintf()等函数代替；

6）必要的地方，将原程序中的 cin 语句以图形库中的 getInteger()、getCoords()等函数代替 （注意：这两个函数只有使用 ege-13.04.02-full.zip 这个 ege 发行版才有！ 你从其它任何地方下载的图形库中都没有这些函数！）

7）对原来的代码进行其他修改，使得程序可以正确运行，并根据输入的坐标等信息绘图。