

A faint, light gray world map is centered in the background of the slide, showing the outlines of continents and major landmasses.

GIS 软件工程

A solid blue trapezoidal shape is positioned at the bottom of the slide, pointing upwards towards the center, creating a base for the title.

- 
- **软件与软件工程**
 - GIS系统分析
 - GIS用户界面设计
 - GIS程序编写

1

软件定义

- “软件”（software）这一名词是在1960年代初从国外传过来。对于它的一种公认的解释为：软件是计算机系统中与硬件相互依存的另一部分，它是包括程序、数据及其相关文档的完整集合。

2

软件的特点

- 软件是一种逻辑实体，而不是具体的物理实体。
- 软件的生产与硬件不同，一旦某一软件项目研制成功，以后就可以大量地复制同一内容的副本。
- 软件的运行和开发常常受到计算机系统的限制。

3

软件分类

- 按功能：系统软件、平台软件、应用软件。
- 按软件规模：超大型软件、大型软件、中型软件、小型软件。
- 按软件服务对象的范围：项目软件、产品软件。

4

软件工程

- 早期的软件开发都是“作坊”式开发方式，软件开发就是写程序并设法使之运行，忽视软件分析的重要性，轻视软件维护。随着软件规模的不断增大，以致1960年代出现软件危机。

- 1968年北大西洋公约组织的计算机科学家在联邦德国召开国际会议，讨论软件危机问题，在这次会议上正式提出并使用了“软件工程”这个名词，在此以后，软件开发逐步采用软件工程方法。

- 软件工程的定义：
 - 关于软件工程目前有很多定义。1983年IEEE（国际电子电气工程师协会）给出的定义为“软件工程是开发、运行、维护和修复软件的系统方法”。

- 软件工程的目标：
 - 在给定成本、进度的前提下，开发出具有可修改性、有效性、可靠性、可理解性、可维护性、可重用性、可适应性、可移植性、可追踪性和可互操作性并满足用户需求的软件产品。

- 软件工程是从管理和技术两方面研究如何更好地开发和维护计算机软件的一门新兴学科。

- B. W. Boehm在1983年提出了软件工程的如下准则：

- 用分阶段的生命周期计划严格管理
- 坚持进行阶段评审
- 实行严格的产品控制
- 采用现代程序设计技术
- 结果应能清楚地审查
- 开发小组的人员应该少而精
- 不断改进软件工程的实践

- 一般情况下，软件的生命周期通常划分为五个阶段，即：软件分析、软件设计、程序编码、软件测试及运行维护。

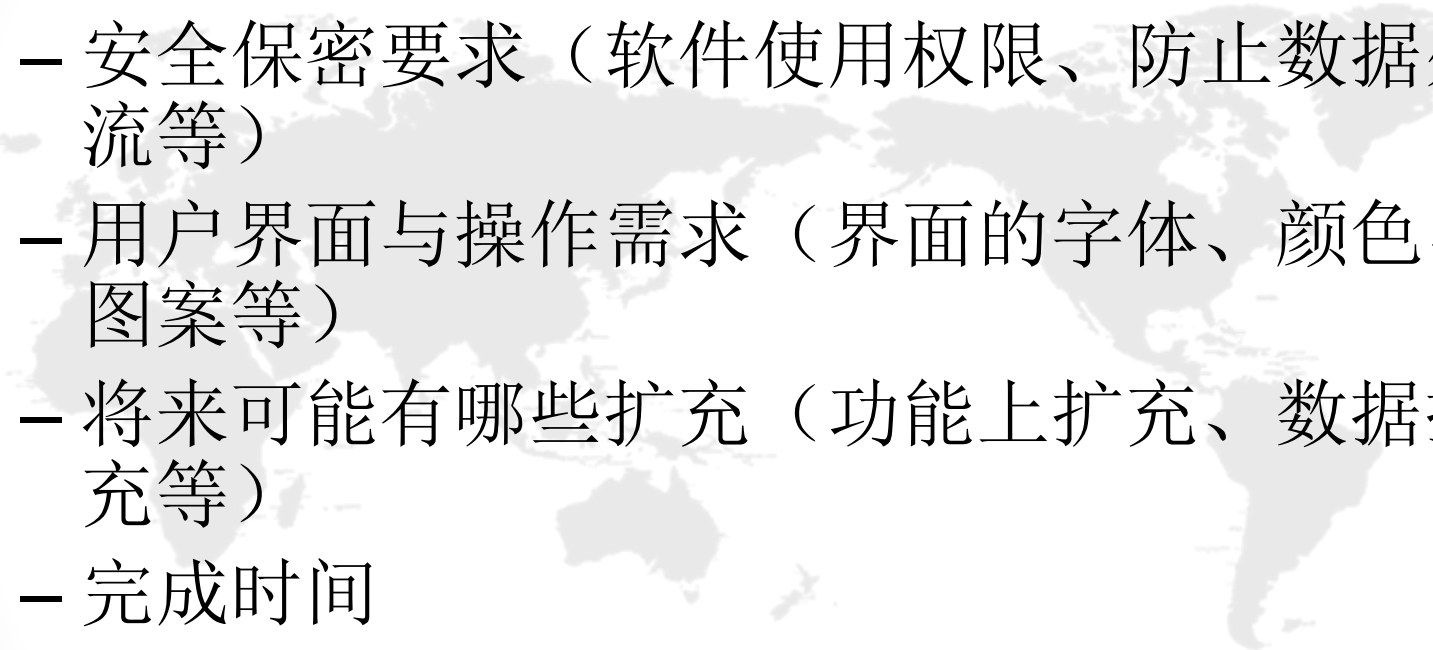
- 
- 软件与软件工程
 - **GIS软件分析**
 - GIS用户界面设计
 - GIS程序编写

- 这里的**GIS**软件主要是指为特定用户开发的项目软件（应用系统）。
- 软件工程强调在系统开发前，首先要进行系统分析，充分了解用户的要求，并把双方的理解用系统方案书表达出来。
- 系统分析的任务包括：
 - 用户需求分析
 - 用户基础分析
 - 可行性分析
 - 系统方案书编写。

1

用户需求分析

- 用户需求分析就是要明确用户的需要及要求，通常包括以下几个方面内容：
 - 系统总体目标（系统的用途）
 - 所期望的功能（有哪些功能）
 - 所要求的系统性能（支持的数据量、处理速度等）
 - 可靠性和质量的要求（软件运行的稳定性、数据处理精度等）

- 
- 环境要求（软件系统运行时对硬软件环境的要求）
 - 安全保密要求（软件使用权限、防止数据外流等）
 - 用户界面与操作需求（界面的字体、颜色、图案等）
 - 将来可能有哪些扩充（功能上扩充、数据扩充等）
 - 完成时间
 - 其它

- 用户需求分析的一个重要过程是用户访谈。访谈有两种基本形式：正式的和非正式的。
 - 在正式的访谈中，系统分析员将提出一些事先准备好的具体问题。
 - 在非正式的访谈中，将提出一些可以自由回答的开放性问题，以鼓励被访问的人员表达自己的想法。

- 在用户分析阶段，最好能快速建立一个原型系统（或利用其它类似系统），使用户对系统的功能有一个基本认识，这样，有利于相互沟通。

2

用户基础分析

- 在明确用户需求以后，需要了解用户已有基础，以避免资源浪费。
- 用户基础分析的内容：
 - 软硬件设备
 - 数据积累
 - 已有研究工作

3

可行性分析

- 软件项目的可行性分析主要是从技术、经济、法律等方面分析项目是否可以实施。


- 技术可行性分析是根据客户提出的系统功能\性能及实现系统的各项约束条件，从技术的角度研究实现系统的可行性，包括当前的科学技术是否支持系统开发的全过程、是否具备系统开发所需的人员和资源、在给定的约束条件下能否实现系统所需功能和性能。

- 经济可行性分析是评估项目的开发成本及项目预期利润，分析系统开发对其他产品或利润的影响。
- 法律可行性分析是研究在系统开发过程中可能涉及到的各种合同、侵权、责任以及各种与法律相抵触的问题。

4

系统方案书

- 系统方案书通常包括以下几方面内容：
 - 系统建设背景
 - 系统目标
 - 系统总体结构
 - 系统开发方案（二次开发、组件开发、底层开发等）
 - 系统软硬件配置方案
 - 人员组织
 - 进度
 - 经费预算
 - 附件（如可行性研究）

- 
- 软件与软件工程
 - GIS系统分析
 - **GIS用户界面设计**
 - GIS程序编写

- 在系统分析之后，需要对系统进行详细设计，其中用户界面设计是一项重要内容，包括界面显示形式设计和界面操作设计。
- 用户界面设计的好坏，影响到用户对系统的态度，决定了系统能否被用户接受，进而影响到系统的应用和推广。友好的用户界面，是GIS成功的条件之一。

用户界面设计原则：

- 在同一系统中，界面应始终保持同一种形式和风格，如菜单选择、命令输入、数据显示和其他功能。

城市人口	州人口
城市名: Boise	州: Nevada
人口: 125738	人口: 1201833

- 操作简单、自动化程度高，尽可能减少用户的操作，如提供列表框选择代替数据输入。

Clip

Input Features

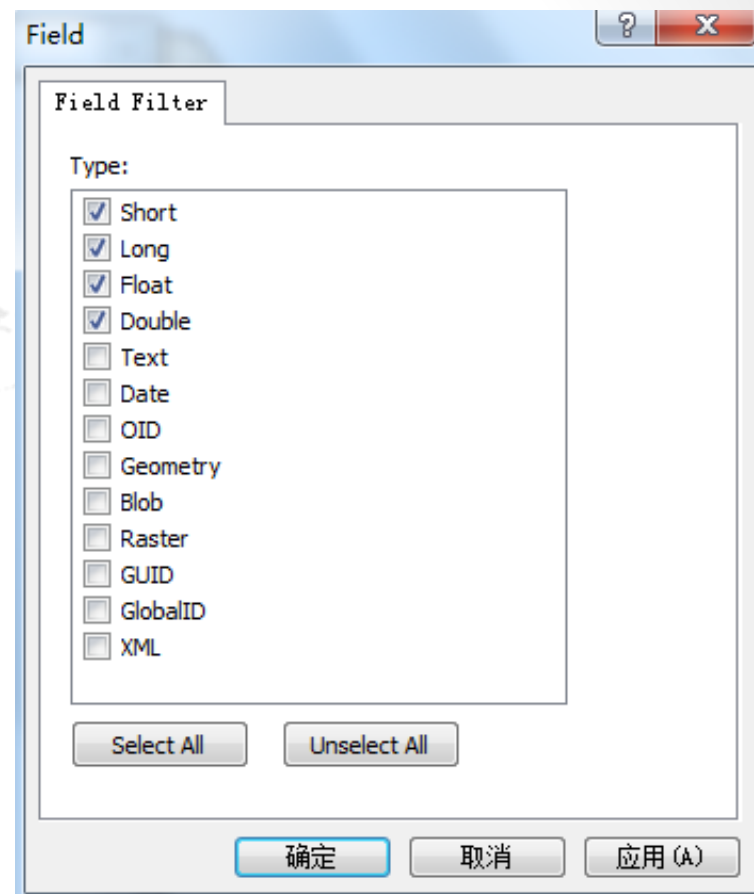
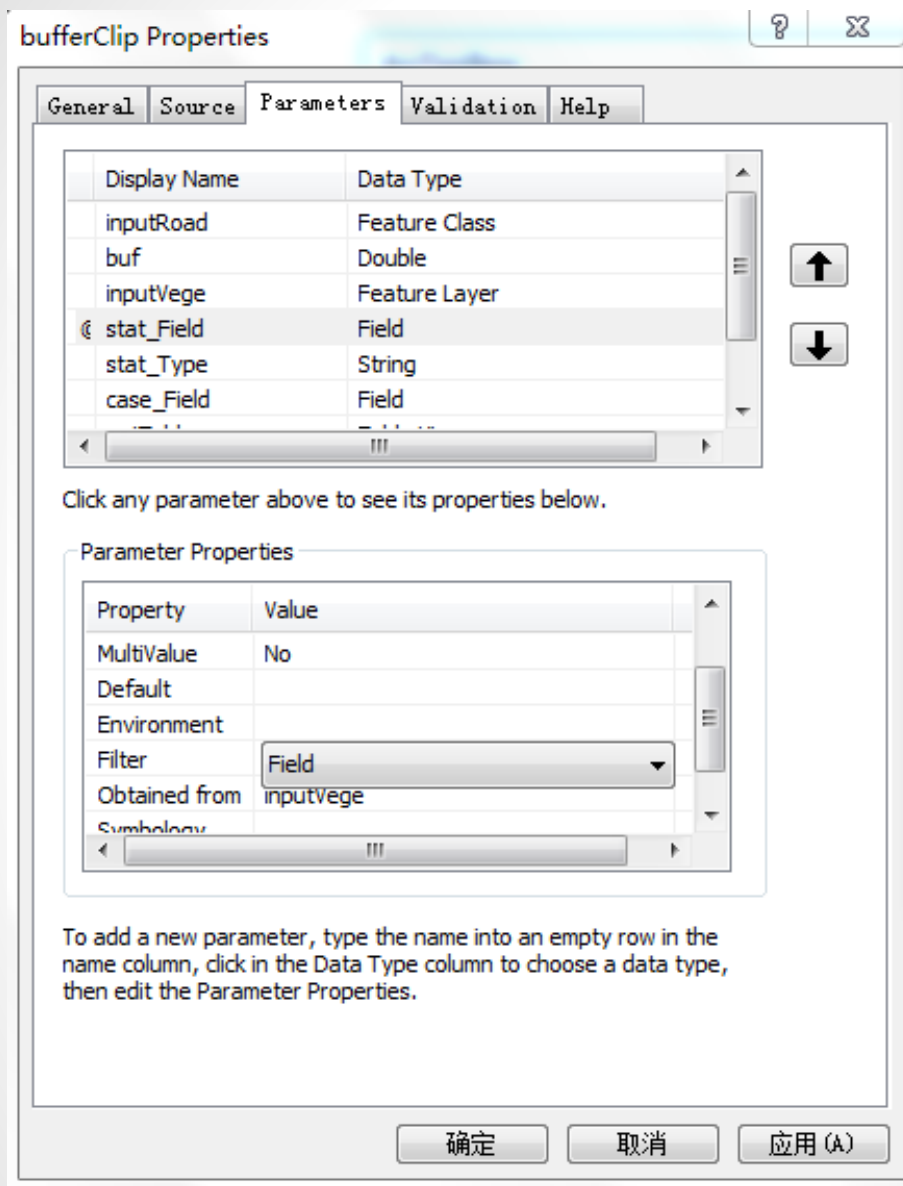
- 道路
- 河流 (多边形)
- 行政区域 (乡镇编码)

Output Feature Class

Cluster Tolerance (optional)

Unknown

- 利用列表框提供用户选择时，只显示有用的选项，无用的选项不显示。如选择字段用于统计，则只显示数字型字段；加载数据时，通过文件过滤在打开文件对话框中只显示可加载的数据。



选择字段用于统计时，则只显示数字型字段。

打开



查找范围 (I):

USA



我最近的文档



桌面



我的文档



我的电脑



网上邻居

- | | | |
|--------------|------------|-------------|
| Capitals | States | Uslakes |
| CAPITALS.DBF | STATES.DBF | USLAKES.DBF |
| CAPITALS.sbn | STATES.sbn | USLAKES.SBN |
| CAPITALS.sbx | STATES.sbx | USLAKES.SBX |
| CAPITALS.SHP | STATES.SHP | USLAKES.SHP |
| CAPITALS.SHX | STATES.SHX | USLAKES.SHX |
| codemog.dbf | untitled | |
| Counties | Ushigh | |
| Counties.dbf | USHIGH.DBF | |
| Counties.sbn | USHIGH.SBN | |
| Counties.sbx | USHIGH.SBX | |
| Counties.shp | USHIGH.SHP | |
| Counties.shx | USHIGH.SHX | |

文件名 (N):

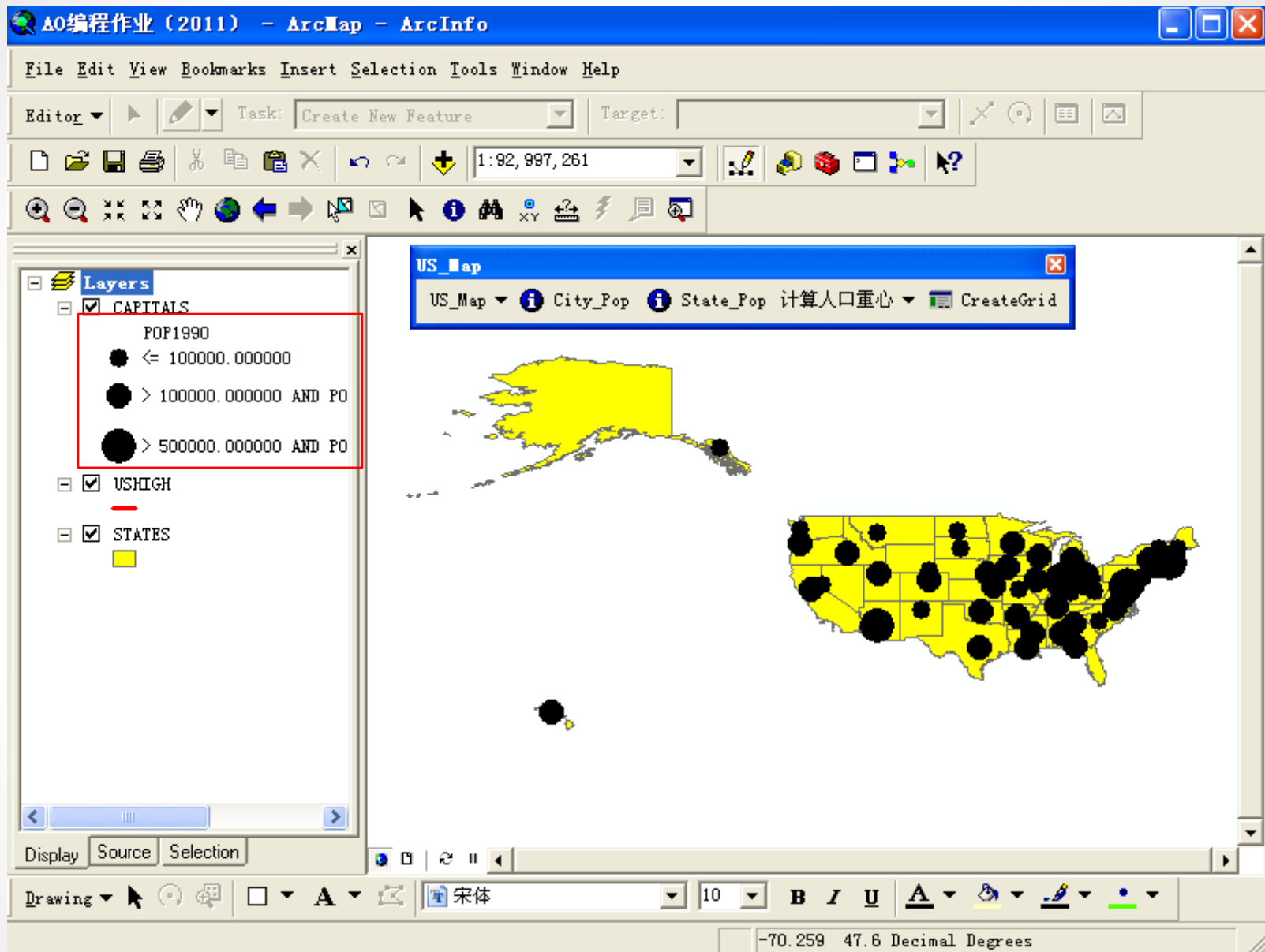
文件类型 (T):

打开 (O)

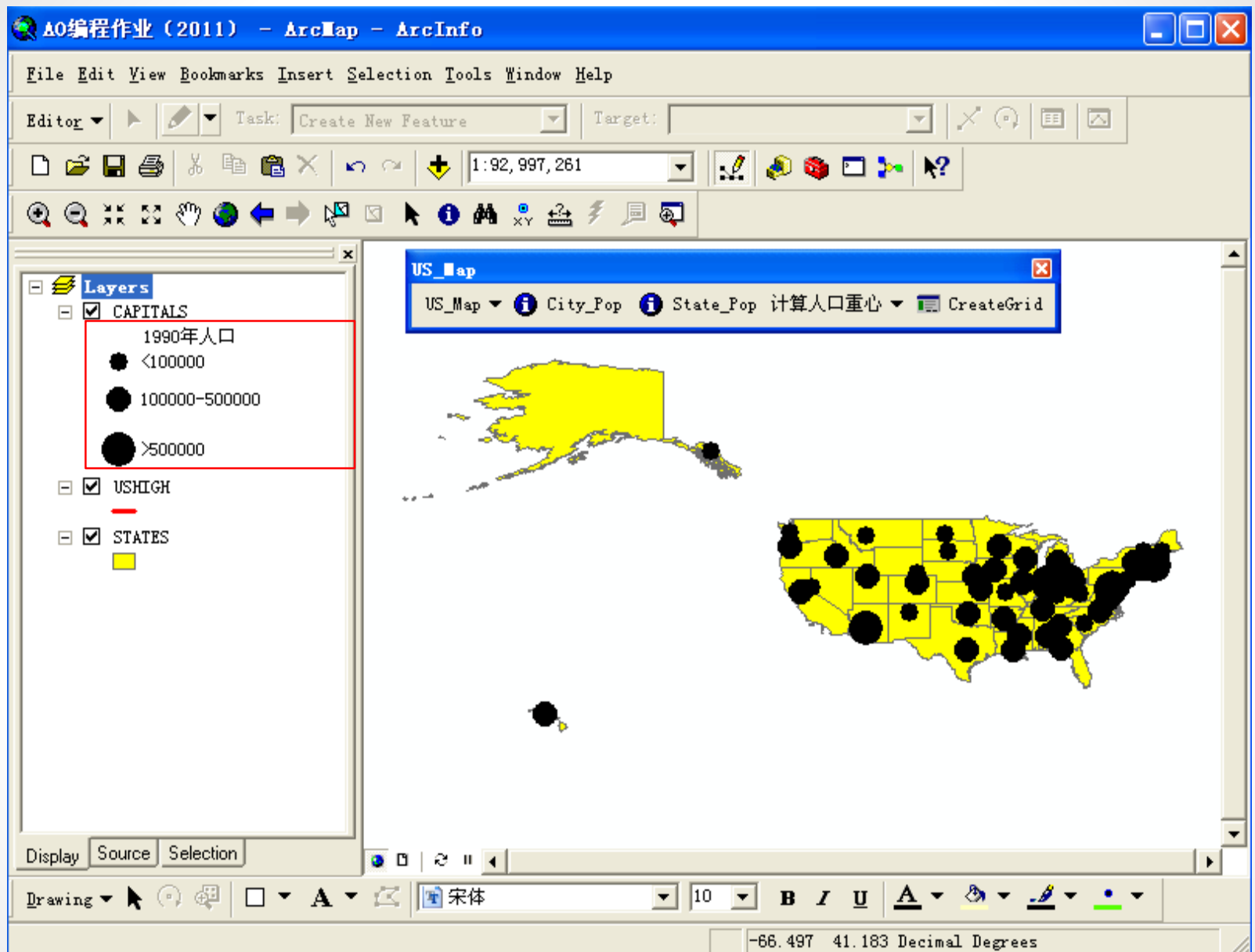
取消

☐ 以只读方式打开 (R)

- 界面上的图层名、字段名、选项名等应按用户熟悉的名字显示，如数据表中的字段可能是以A1、A2、B1、B2...形式显示，但在界面显示时应按实际的名字显示，如海拔高度、地貌类型、土壤类型、土壤厚度等。

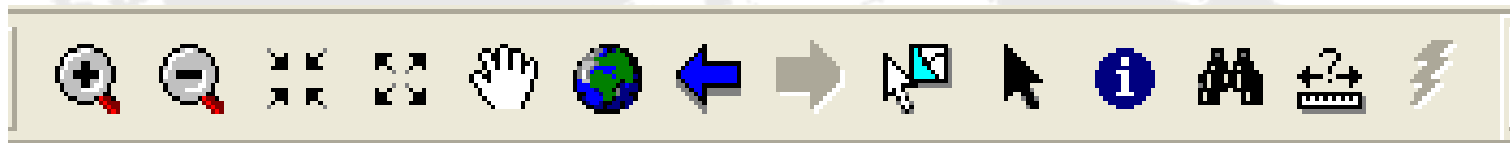


缺省情况下显示的图例组和图例标注

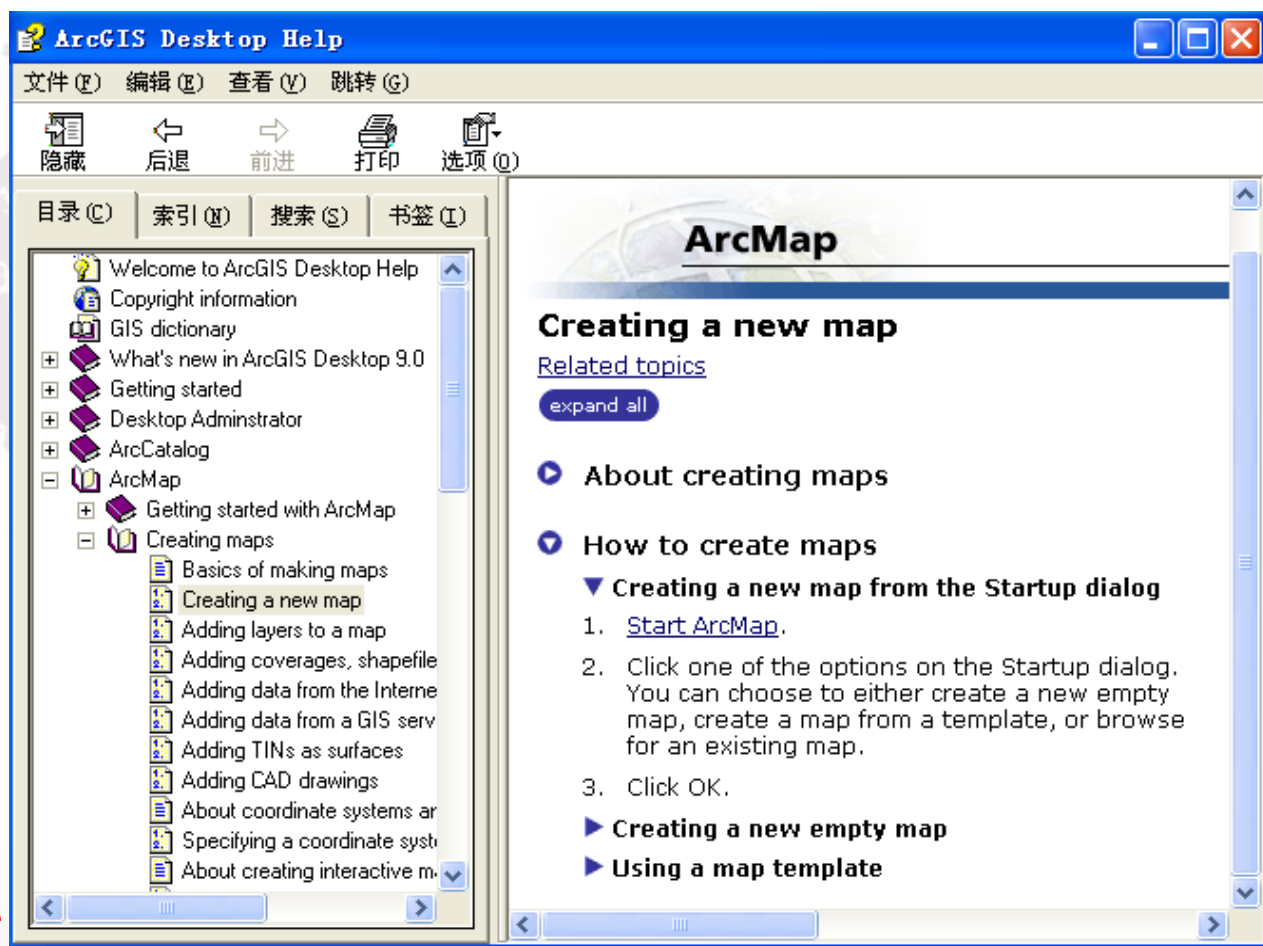


针对用户要求显示的图例组和图例标注

- 术语、符号等标准化，如GIS中的放大、缩小、漫游等按钮都有固定的符号，尽可能采用标准的或通用的符号。



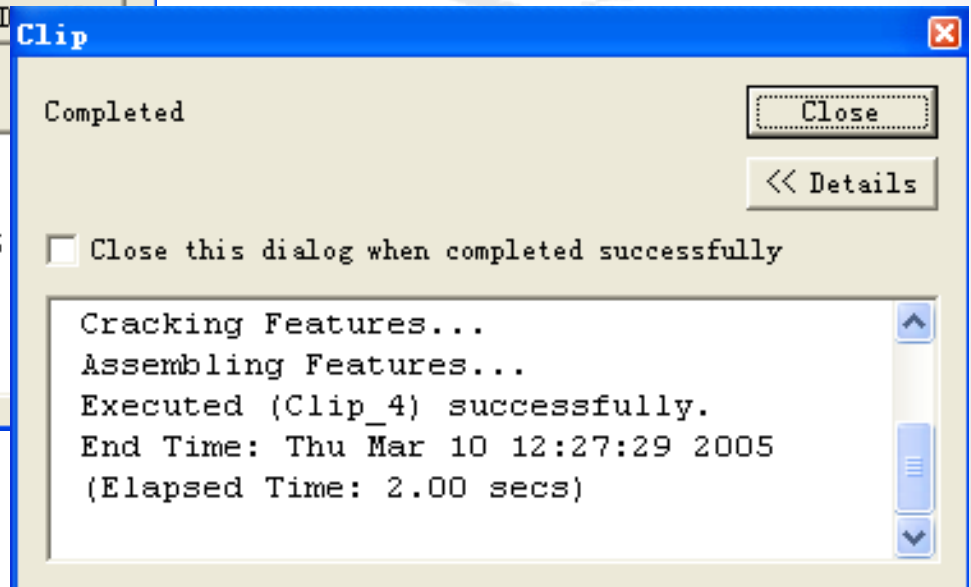
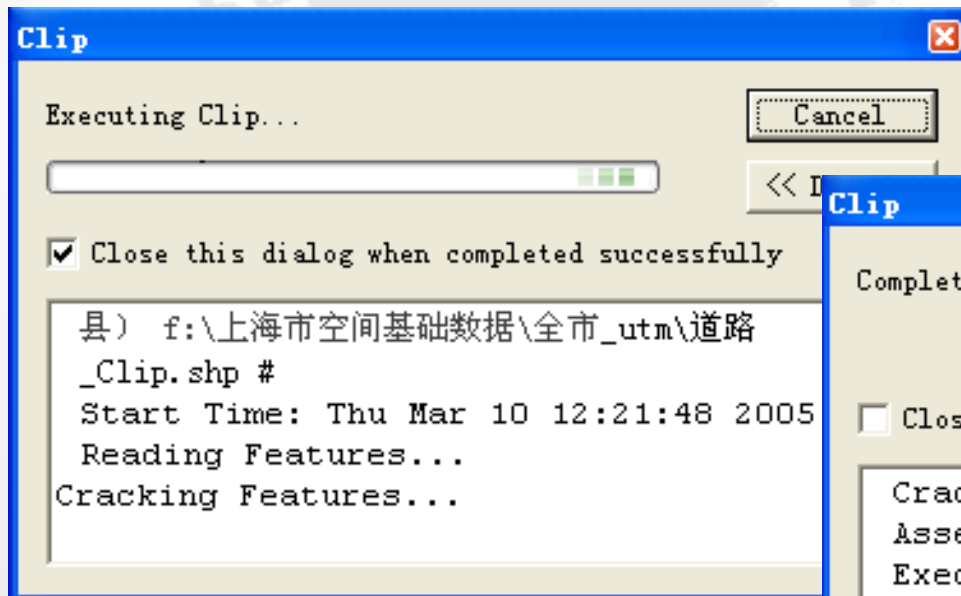
- 提供操作提示和联机帮助功能，在对话框中提供操作提示，系统有帮助菜单并有详细内容。



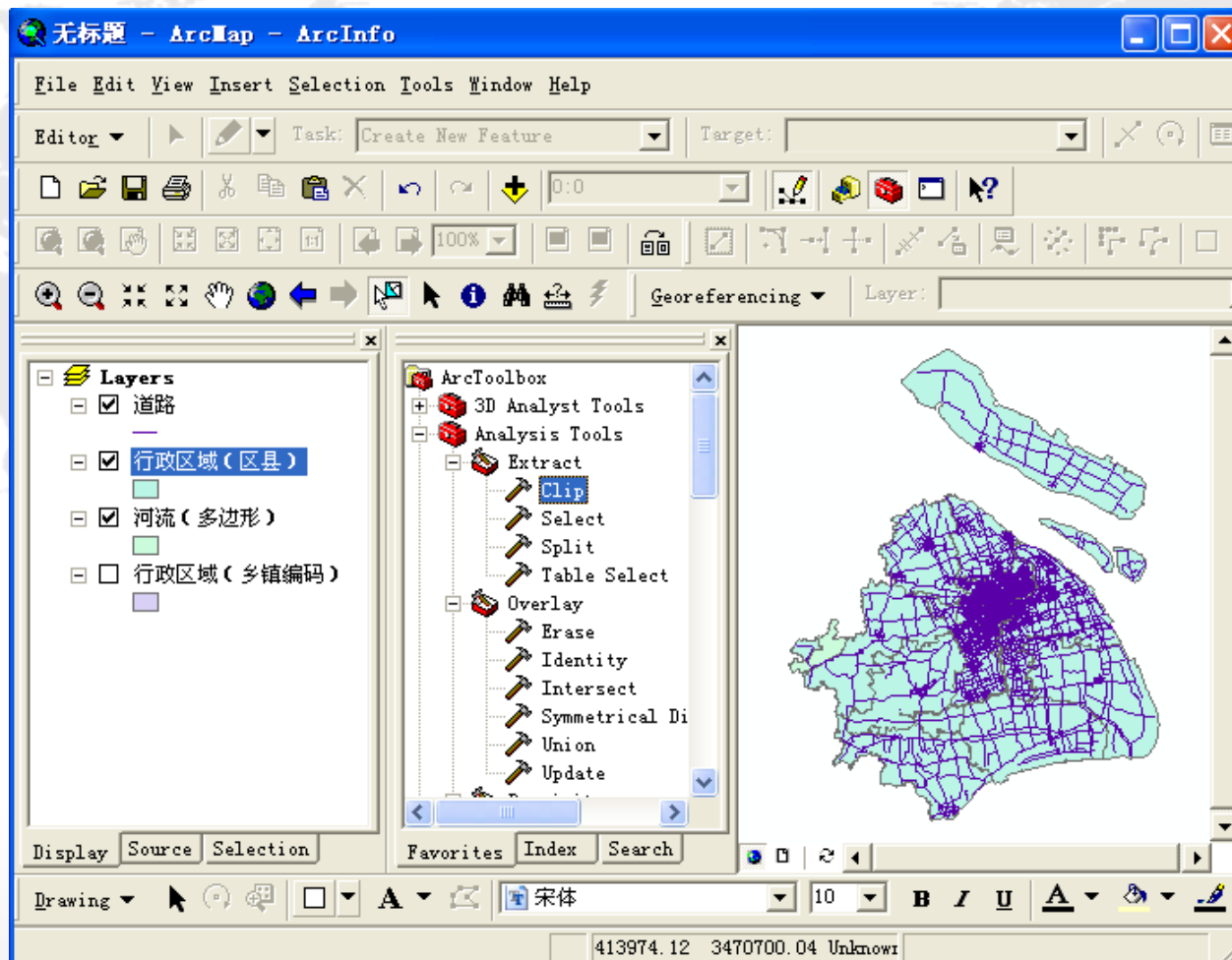
操作提示

联机帮助

- 提供系统运行信息和响应信息，系统在处理运行过程中，应改变鼠标的显示形式，如处理时间超过10秒，应显示进程条，运行结束后应有响应信息，如屏幕显示发生变化，或显示处理结束窗口。



- 用户可以根据需要制定和修改界面方式，允许用户对界面的显示形式进行修改，如放大、缩小窗口。



- 关键操作要有强调和警告，能保证有关程序和数据的安全性。

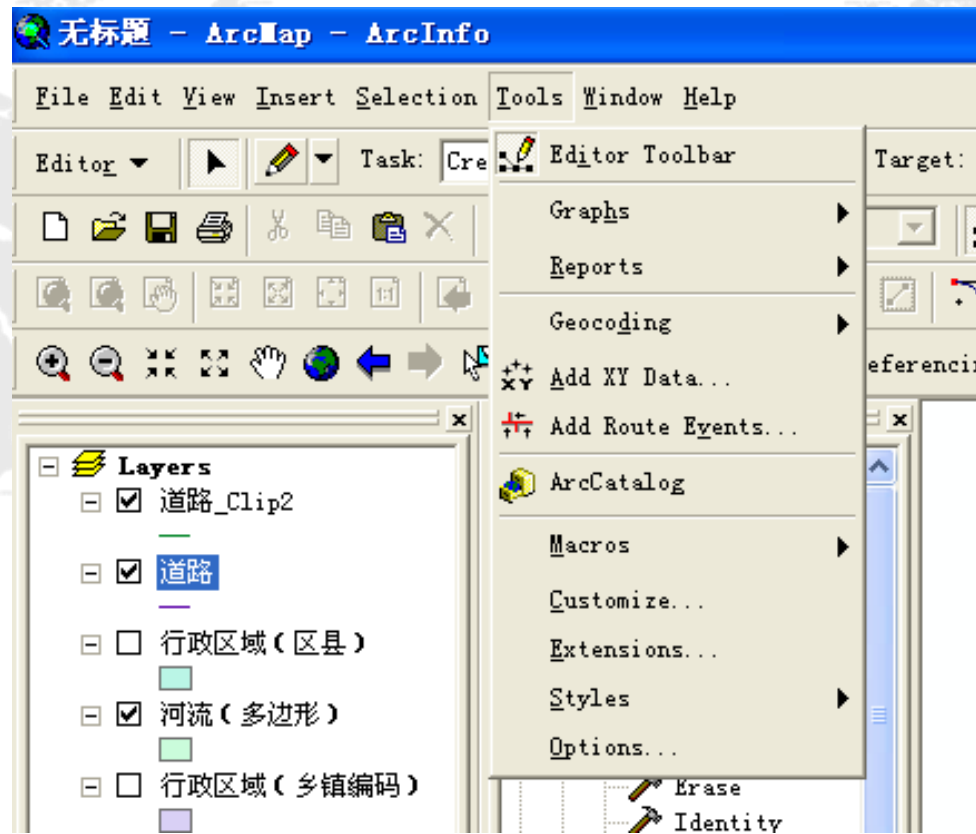
Object Already Exists

The object named "道路_Clip2.shp" already exists. Do you wish to replace it?

是(Y)


否(N)

- 按功能分类组织界面上的活动，对菜单项、按钮等按照功能进行组织分割。



- 提供缺省选择，需要用户进行选择时，以最有可能的选择作为缺省选择。



- 
- 软件与软件工程
 - GIS系统分析
 - GIS用户界面设计
 - **GIS程序编写**

- 程序编写的过程就是利用某种程序设计语言把详细设计编码成计算机可接受的形式，也是人借助编程语言与计算机通信的过程。
- 应该说，在系统开发的各个阶段中，编程是最容易，也是人们已掌握得较好的一项工作。但编写一个好的程序需要高水平的编程人员。

- 对于程序好坏的评介，早期与现在的标准有很大不同。
 - 早期的计算机内存小、速度慢，人们往往把程序的长度和执行速度放在很重要的位置。
 - 现在情况有了很大的不同，一般认为好程序应具备的最重要的条件是可读性，此外，还应具有可靠性、可重用性、可适应性、可移植性、可追踪性和可互操作性等。

1

提高程序可读性

- 程序员在写程序时应该记住：程序不仅是给计算机执行的，也是供人阅读的。
- 提高程序可读性的方法
 - 模块化编程。
 - 程序中包括说明文档。
 - 良好的编程风格。

模块化编程

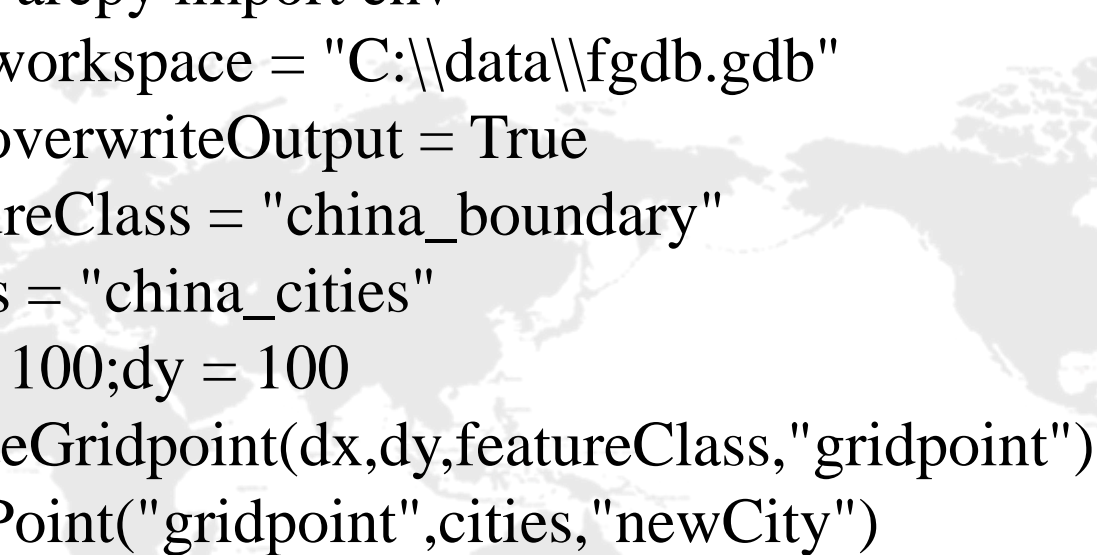
- 模块化编程是对复杂的程序按功能进行分解，编写相应的功能函数，然后通过函数的调用实现最终目标。如要在一个区域内找出离已有城市距离和为最小的点，可以先编写两个功能函数（`createGridpoint`和`findPoint`），然后通过定义输入数据及调用函数得到最终结果。

```
def createGridpoint(dx,dy,featureClass,output):
    import arcpy
    from arcpy import env
    desc = arcpy.Describe(featureClass)
    ext = desc.extent
    XMin = int(ext.XMin);YMin = int(ext.YMin)
    XMax = int(ext.XMax);YMax = int(ext.YMax)
    point = arcpy.Point()
    pointGeometryList = []
    for X in range(XMin, XMax, dx):
        for Y in range(YMin, YMax, dy):
            point.X = X;point.Y = Y
            pointGeometry = arcpy.PointGeometry(point)
            pointGeometryList.append(pointGeometry)
    arcpy.Clip_analysis (pointGeometryList, featureClass, output)
```

createGridpoint函数

```
def findPoint(inputFeature,cities,output):
    import arcpy
    arcpy.PointDistance_analysis (inputFeature, cities, "dist")
    arcpy.Statistics_analysis ("dist", "summary", [["DISTANCE", "SUM"]],
"INPUT_FID")
    dist_List = []
    FID_List = []
    cur = arcpy.SearchCursor("summary")
    for row in cur:
        dist_List.append(row.SUM_DISTANCE)
        FID_List.append(row.INPUT_FID)
    min_dist = min(dist_List)
    min_dist_FID = FID_List[dist_List.index(min_dist)]
    cur = arcpy.SearchCursor(inputFeature,"OBJECTID = " + str(min_dist_FID))
    for row in cur:
        geometry = row.shape
    arcpy.CopyFeatures_management (geometry, output)
```

findPoint函数

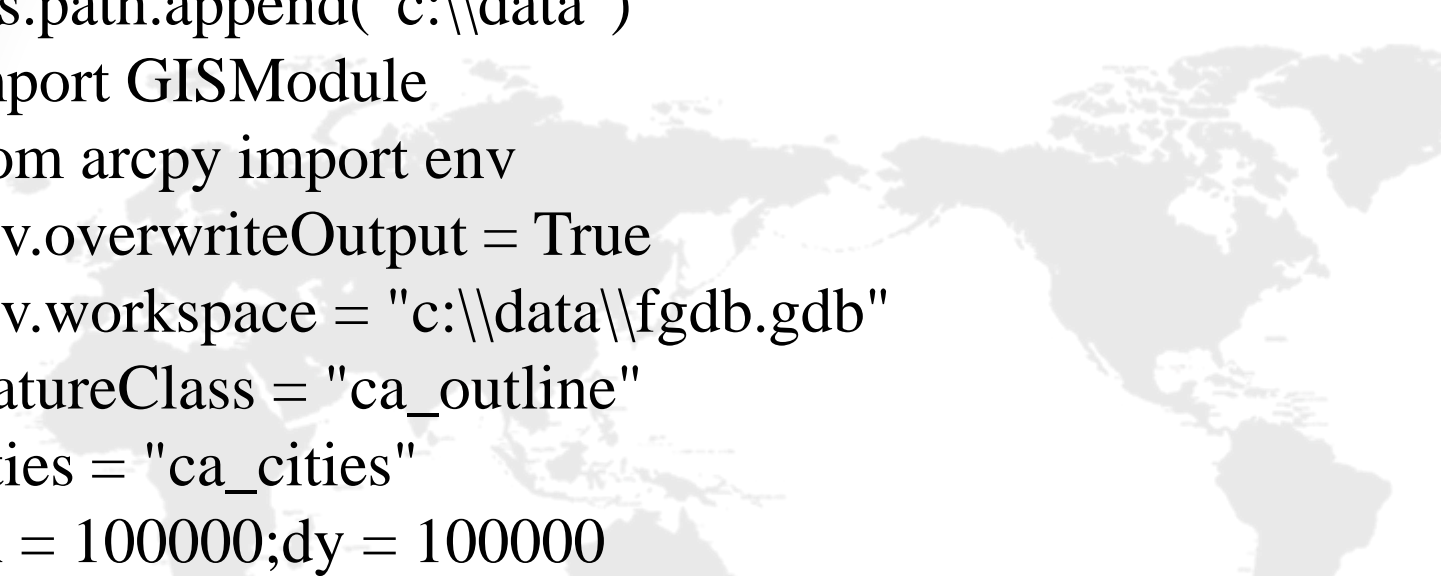
A faint, light gray world map is visible in the background, centered behind the text. It shows the outlines of continents and major landmasses.

```
from arcpy import env
env.workspace = "C:\\data\\fgdb.gdb"
env.overwriteOutput = True
featureClass = "china_boundary"
cities = "china_cities"
dx = 100;dy = 100
createGridpoint(dx,dy,featureClass,"gridpoint")
findPoint("gridpoint",cities,"newCity")
```

定义输入数据及调用函数

- 编写的函数或类可以保存在模块中，供其它python程序调用。





```
import sys
sys.path.append("c:\\data")
import GISModule
from arcpy import env
env.overwriteOutput = True
env.workspace = "c:\\data\\fgdb.gdb"
featureClass = "ca_outline"
cities = "ca_cities"
dx = 100000;dy = 100000
GISModule.createGridpoint(dx,dy,featureClass,"gridpoint")
GISModule.findPoint("gridpoint",cities,"newCity")
```

把函数放在独立的模块（GISModule.py）中，供其它程序调用（这里假定模块在c:\\data 文件夹中）。

程序中增加必要的说明文档

- 程序中增加必要的说明文档可以提高程序可阅读性。说明文档用注释语句书写，包括文件头注释、函数注释、程序段注释和语句注释。注释不是重复程序语句，而应提供从程序本身难以得到的信息。在修改程序时，要注意对注释进行相应的修改。

"""

工具名：确定与已有城市距离和为最小的点

文件名：NearestPoint.py

版本：1.0

作者：华东师范大学

日期：2014年6月16日

说明：

这个工具用于找出与已有城市距离和为最小的点，包括两个步骤：

(1) 在指定区域中产生均匀分布的格点。

(2) 计算每个格点与已有城市的距离和，找出距离和为最小的格点，保存到一个新的点要素类中。

"""

文件开头的注释样例

```
def createGridpoint(dx,dy,featureClass,output):  
    """在指定区域中产生均匀分布的格点。
```

输入：

dx (Num) :x方向间距。

dy (Num) :y方向间距。

featureClass(str)：定义区域的要素类（多边形）。

output (str)：输出的要素类（点要素类）

```
"""
```

函数开头的注释样例

以要素类的最小x、y为原点，以dx、dy为间隔，一直到最大x、y产生点几何对象

```
for X in range(XMin, XMax, dx):
```

```
    for Y in range(YMin, YMax, dy):
```

```
        point.X = X;point.Y = Y
```

```
        pointGeometry = arcpy.PointGeometry(point)
```

```
        pointGeometryList.append(pointGeometry)
```

对点几何对象进行切割，切割出的点几何对象保存到输出要素类中

```
arcpy.Clip_analysis (pointGeometryList, featureClass, output)
```

程序段和语句注释样例

- 编程的风格在很大程度上影响着程序的可读性。良好的编程风格是在不影响性能的前提下，有效地编排和组织程序，以提高可读性。

规范化命名

- 变量、函数、类等命名尽管没有严格要求，但应遵循一些基本规范，如：
 - 名称最好采用英文单词或其组合，便于记忆和阅读。切忌使用汉语拼音来命名。
 - 变量的名字应当使用“名词”或“名词+名词”或“形容词+名词”，第一个单词的第一个字母小写，后面单词的第一个字母大写，如 `outFeatureClass`、`clusterTolerance`。
 - 程序中不要出现仅靠大小写区分的相似的名称，如 `x` 和 `X`。

直接反映代码的意图

- 代码的意图是直接的，而不是隐含的。

```
L = []  
for i in range(1,4):  
    for j in range(1,4):  
        Value = (i/j) * (j/i)  
        L.append(Value)  
print L
```

不是直接反映意图的程序代码

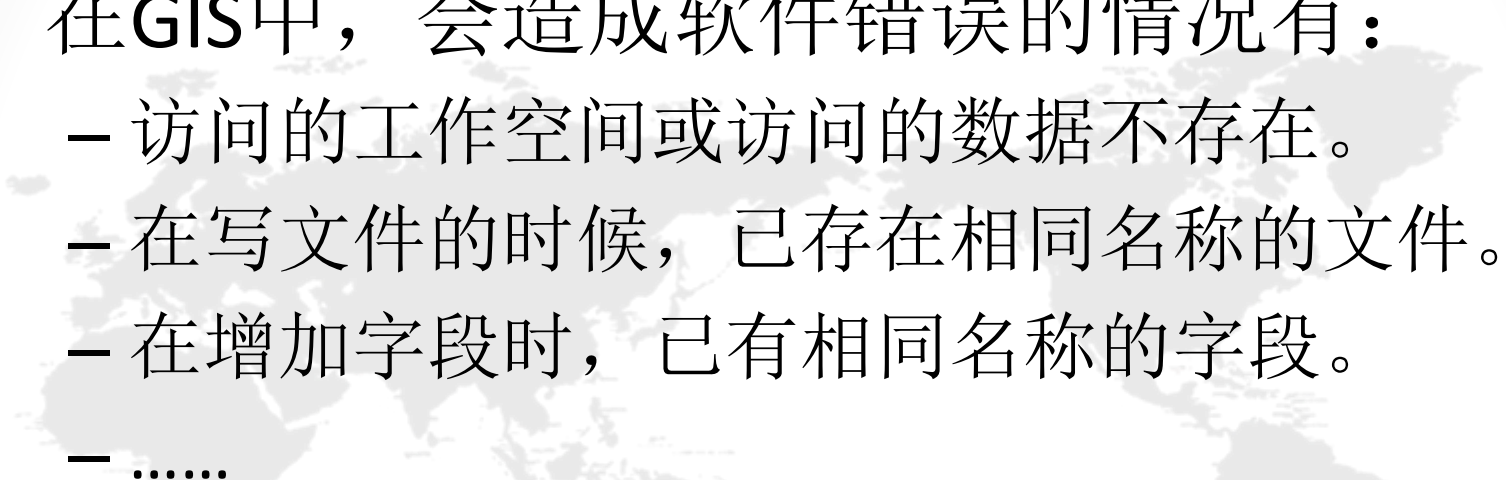
```
L = []  
for i in range(1,4):  
    for j in range(1,4):  
        if i == j:  
            Value = 1  
        else:  
            Value = 0  
        L.append(Value)  
print L
```

直接反映意图的程序代码

2

提高软件的容错性

- 提高软件的容错性是针对由于软件运行环境的变化、用户的错误操作等情况造成的软件错误进行处理，避免出现死机或退出系统的情况。
- 提高软件的容错性关键是要分析出哪些语句会造成软件错误，然后进行相应的容错处理。

- 
- 在GIS中，会造成软件错误的情况有：
 - 访问的工作空间或访问的数据不存在。
 - 在写文件的时候，已存在相同名称的文件。
 - 在增加字段时，已有相同名称的字段。
 -

- 对会造成软件错误的语句要通过错误捕获语句或判断语句进行处理。

- **实例1：** 如访问的工作空间不存在，则通过错误捕获语句显示该工作空间不存在的信息。

```
import arcpy
Workspace = "c:\data1"
FileName = "rivers.shp"
try:
    arcpy.CreateFeatureclass_management(Workspace,FileName,"POLYLINE")
except:
    print "c:\data1 does not exist"
```

- 实例2：在增加字段时，如已有相同名称的字段，则通过增加数字改变字段名，一直到和已有字段不重名（即产生唯一字段名）。

```
import arcpy  
fc = "c:\\data\\ca_pm10_pts.shp"  
desc = arcpy.Describe(fc)  
fields = desc.fields  
fn = "x"  
iterate = True  
n = 0
```

```
while iterate:
    fn_change = False
    for field in fields:
        if field.name == fn:
            n = n + 1
            fn = "x" + str(n)
            fn_change = True
            break
        else:
            continue
    if fn_change == False:
        iterate = False
    else:
        continue
arcpy.AddField_management(fc, fn, "TEXT")
del fc
```

产生唯一字段名