EPANETH程序员工具箱

原著：Lewis A. Rossman
（俄亥俄州辛辛那提市美国环境保护局研究和开发办公室
国家风险管理研究实验室供水和水资源分部，45268）
翻译：李树平
（上海市同济大学环境科学与工程学院，200092）
2011年9月21日

EPANETH程序员工具箱

EPANETH是一个分析配水系统水力和水质特性的程序。EPANETH程序员工具箱是一个函数动态链接库（DLL），允许开发人员为了特定需求，定制EPANETH计算引擎。函数可以包含在C/C++, Delphi Pascal, Visual Basic，或者任何其它语言编写的32位Windows应用程序中，可以在Windows DLL中调用函数。工具箱DLL文件命名为EPANET2H.DLL，与EPANETH一起发布。工具箱包含了头文件，函数定义文件和.lib文件，以简化将它与C/C++, Delphi和Visual Basic代码衔接的任务。

EPANET及其程序员工具箱由美国环境保护署国家风险管理研究实验室供水和水资源分部开发；汉化后命名为EPANETH及其程序员工具箱，该汉化工作由同济大学环境科学与工程学院完成。

1．工具箱概览

程序员工具箱是EPANETH模拟软件包的扩展。EPANETH执行有压管网水力和水质特性的延时模拟。管网可以包含管道、节点（管道连接节点）、水泵、阀门、贮水池或者水箱。EPANETH跟踪每一管道的流量、每一节点的压力、每一水池的水位，以及多时段模拟过程中整个管网的化学成分浓度。除了化学成分，也可以模拟水龄和源头跟踪。

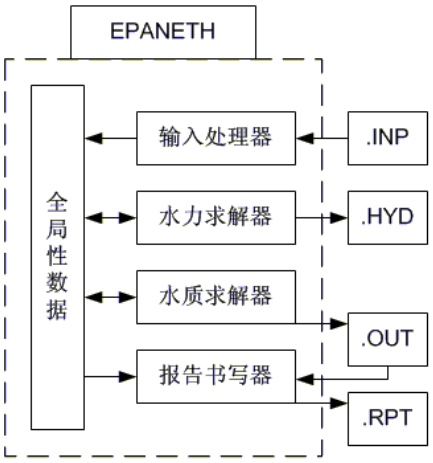
工具箱提供了一系列函数，允许程序员定制应用EPANETH的水力和水质计算引擎。利用工具箱之前，应熟悉EPANETH表示管网的方式，以及执行模拟需要的设计和运行信息。该信息可以从EPANETH的在线帮助文件或者EPANETH用户手册获得。

为了分析配水系统的工具箱函数，其典型应用如下：

1. 使用ENopen函数打开工具箱系统，同时打开EPANETH输入文件。
2. 使用ENsetxxx系列函数，改变和选择系统特性。
3. 利用ENsolveH函数执行完整的水力模拟(它自动将结果保存到水力文件)；或者利用ENopen - ENinitH - ENrunH - ENnextH - ENcloseH系列函数，逐步进行水力模拟；利用ENgetxxx系列函数访问计算结果。
4. 利用ENsolveQ执行完整的水质分析(它自动将水力和水质结果保存到输出文件)；或者使用ENopenQ - ENinitQ - ENrunQ - ENnextQ (或者ENstepQ) - ENcloseQ系列函数，逐步进行水质模拟；利用ENgetxxx系列函数访问结果。
5. 为了执行额外的分析，返回步骤2，或者利用ENreport函数，将格式化报告写入到报告文件。
6. 调用ENclose函数，关闭所有文件，并释放系统内存。

利用这些函数的特定例子见示例应用主题。

2. 数据流程图



EPANETH工具箱利用ANSI标准C语言书写，对于输入过程、水力分析、水质分析、稀疏矩阵/线性方程分析和报告生成，具有单独的代码模块。分析管网的数据流程图见左图。本流程图描述的处理步骤总结如下：

- 输入处理器模块接受被模拟管网的描述，来自外部输入文件（.INP)。文件的内容被分析、解释和保存在共享内存区域。
- 水力求解器模块执行延时水力模拟。每一时间步长获得的结果可以写入外部非格式化（二进制）水力文件(.HYD)。这些时间步长可以表示系统改变中间点的时间，因为水池充满或者放空，或者水泵启闭，由于水位控制或者时间调度运行。
- 如果需要水质模拟，水质求解器模块访问来自水力文件的流量数据，计算整个管网在每一水力时间步长中物质的迁移和反应。该过程可以书写前面计算的水力结果，以及每一预先设定报告间隔的水质结果，写入到非格式化（二进制）输出文件（.OUT)。如果没有调用水质分析，那么保存在.HYD文件中的水力结果可以均匀报告间隔，简单写入到二进制输出文件。
- 如果需要，报告书写器模块从二进制输出文件(.OUT)读回计算的模拟结果，对于每一报告时段，可将选择的数值写入格式化报告文件（.RPT)。运行过程产生的任何错误或者警告信息也将写入该文件。

工具箱函数将在程序员控制下执行所有这些步骤，包括阅读或者更改多数系统全局性数据。

3. 怎样使用工具箱

以下主题简要描述怎样利用工具箱完成一些基本任务。工具箱完整应用的代码列表见示例应用主题。

3.1 打开和关闭工具箱

调用其他任何函数之前，工具箱必须打开EPANETH的输入文件o，获取分析管网的描述。(例外是ENepanet函数，它执行了完整的水力/水质模拟，类似于EPANETH命令行执行情况)。一旦完成所有分析，必须关闭它，释放所有分配内存。完成该任务的函数有ENopen和ENclose。使用这些函数的例子如下。

```
char *f1, *f2, *f3;
int  errcode;
errcode = ENopen(f1, f2, f3);
if (errcode > 0)
{
   ENclose();
   return;
}
/*
   调用执行期望分析的函数
*/
ENclose();
```

3.2 检索和设置管网参数

工具箱具有各种用于检索和设置参数的函数，定义了被分析管网的设计和运行状况。所有检索函数名称均以*ENget*开始(例如ENgetnodevalue, ENgetoption等)，而用于设置参数值的函数以*ENset*开始(例如ENsetnodevalue, ENsetoption等)。

其中多数函数使用索引号参考特定管网组件（例如节点、管段或者时间模式）。该数字是所有类似类型组件列表中的位置 (例如节点10在管网中是从1开始的第10个节点)，在被处理的输入文件中没有相同的ID标签赋给组件。具有一系列参数确定给定ID标签的组件索引(见ENgetlinkindex, ENgetnodeindex和ENgetpatternindex)。类似的，具有函数检索给定索引号的组件ID标签 (参见ENgetlinkid, ENgetnodeid和ENgetpatternid)。函数ENgetcount可用于确定管网中不同组件的数量。

以下编码是使用参数索引和设置函数的例子。它将所有直径为250 mm的管道改为300 mm。

```
int   i, Nlinks;
float D;
ENgetcount(EN_LINKCOUNT, &Nlinks);
for (i = 1; i <= Nlinks; i++)
{
  ENgetlinkvalue(i, EN_DIAMETER, &D);
  if (D == 250)
    ENsetlinkvalue(i, EN_DIAMETER, 300);
}
```

3.3 执行水力分析

利用工具箱执行水力分析具有两种方式：

1. 使用ENsolveH函数，运行完整的延时分析，不需要访问中间结果；
2. 使用ENopenH - ENinitH - ENrunH- ENnextH - ENcloseH系列函数，步进形式模拟水力时间步长。

方法1对于仅仅希望执行单一的水力分析是有用的，可能将输入提供给水质分析。利用该方法，每一时间步长的水力分析结果总是保存到水力文件。

如果需要访问时间步长之间的结果；或者希望有效地执行多次分析，必须使用方法2。为了完成后者，可以仅调用一次ENopenH，开始该过程，然后连续调用ENinitH - ENrunH – EnnextH，执行每一次分析，最后调用ENcloseH，关闭水力系统。这样的例子见下（调用ENnextH不是需要的，因为在该例中仅仅执行单一时段分析)。

```
int  i, Nruns;
long t;
ENopenH()
for (i = 1; i <= Nruns; i++)
{
/* 设置当前执行的参数 */
  setparams(i);
```

```
/* 初始化水力特性 */
  ENinitH(0);
/* 单一时段运行 */
  ENrunH(&t);
/* 检索结果 */
  getresults(i);
}
ENcloseH();
```

3.4 执行水质分析

水质分析之前，必须产生水力结果，来自执行水力分析或者来自以前执行而导入保存的水力文件。与水力分析类似，水质分析也具有两种方式：

1. 利用ENsolveQ函数，执行完整的延时分析，不需要访问中间结果；
2. 利用ENopenQENopenH>info - ENinitQ - ENrunQ- ENnextQENnextH>info - ENcloseQ函数序列，步进通过每一水力时间步长模拟。(ENnextQ替换为ENstepQ，将步进通过每一水质时间步长。)

使用方法2的例子如下。

```
int  err;
long t, tstep;
err = ENsolveH();
if (err > 100) return(err);
ENopenQ();
ENinitQ(1);
do
{
  ENrunQ(&t);
  ENnextQ(&tstep);
} while (tstep > 0);
ENcloseQ();
ENreport();
```

3.5 检索计算结果

ENgetnodevalue和ENgetlinkvalue函数，用于检索水力和水质模拟结果。计算的参数（及其工具箱代码）可以按照以下检索：

对于节点:	**对于管段:**
EN_DEMAND (需水量)	EN_FLOW (流量)
EN_HEAD (水力水头)	EN_VELOCITY (流速)
EN_PRESSURE (压强)	EN_HEADLOSS (水头损失)
EN_QUALITY (水质)	EN_STATUS (管段状态)
EN_SOURCEMASS (水质源头质量进流)	EN_SETTING (水泵转速或者阀门设置)

以下代码说明怎样在每一时间步长水力分析之后，检索管网每一节点的压力(writetofile是由用户定义的将记录写入文件的函数)：

```
int   i, NumNodes;
long  t, tstep;
float p;
char  id[16];
ENgetcount(EN_NODECOUNT, &NumNodes);
ENopenH();
ENinitH(0);
do
{
  ENrunH(&t);
  for (i = 1; i <= NumNodes; i++)
  {
     ENgetnodevalue(i, EN_PRESSURE, &p);
     ENgetnodeid(i, id);
     writetofile(t, id, p);
  }
  ENnextH(&tstep);
} while (tstep > 0);
ENcloseH();
```

3.6 书写报告

工具箱具有一些内置功能，用于产生报告到文件的格式化输出结果。更特殊的报告需要通过书写特定编码处理。

ENsetreport函数用于定义报告的格式，同时ENreport函数实际上书写报告。或者仅仅在水力或者水质分析之后调用。创建列出所有节点的报告例子，其中压力超过20 psi随模拟时段变化，见下：

```
/* 计算范围 (max - min) */
  ENsettimeparam(EN_STATISTIC, EN_RANGE);
/* 求解水力特性 */
  ENsolveH();
/* 从水力文件向输出文件转换结果 */
  ENsaveH();
/* 定义报告的内容 */
  ENresetreport();
  ENsetreport("FILE myfile.rpt");
  ENsetreport("NODES ALL");
  ENsetreport("PRESSURE PRECISION 1");
  ENsetreport("PRESSURE ABOVE 20");
/* 将报告写入文件 */
  ENreport();
```

4. 示例应用

例1 - 为其他应用程序提供内嵌引擎

本例说明工具箱为其他应用程序提供管网分析的简单性。应用程序将需要三个步骤：

1. 将配水系统数据写入到EPANETH格式的输入文件(参见输入文件格式)。

2. 调用ENepanet函数，提供EPANETH输入文件名，状态和错误信息需要写入的报告文件名，以及将包含分析结果的二进制输出文件名。

3. 通过访问输出文件，以在应用程序中显示期望的分析结果(参见输出文件格式)。

例2 - 为消防流量研究建立消火栓性能曲线

本例说明了工具箱怎样用于开发消防流量研究中使用的消火栓性能曲线。该曲线说明可用作系统内压力函数的的节点流量。曲线通过稳态水力分析的次数产生，每一次对受到不同需水量影响的感兴趣节点进行分析。对于该例，假设感兴趣节点ID标签为MyNode，N次不同的需水量水平保存在需要检验的数列D中。相应的压力保存在P中。为了保证代码更好的可读性，没有进行工具箱函数调用返回结果的错误检查。

例2 - C

```
#include "epanet2h.h"

void HydrantRating(char *MyNode, int N, float D[], float P[])
{
  int   i, nodeindex;
  long  t;
  float pressure;

  /* 打开EPANETH工具箱和水力求解器 */
  ENopen("example2.inp", "example2.rpt", "");
  ENopenH();

  /* 获取感兴趣节点的索引 */
  ENgetnodeindex(MyNode, &nodeindex);

  /* 迭代计算所有需水量 */
  for (i=1; i<N; i++)
  {

  /* 设置节点需水量、初始水力特性， */
  /* 执行单一时段运行，并检索压强   */
     ENsetnodevalue(nodeindex, EN_BASEDEMAND, D[i]);
     ENinitH(0);
     ENrunH(&t);
     ENgetnodevalue(nodeindex, EN_PRESSURE, &pressure);
     P[i] = pressure;
  }
```

```
  /* 关闭水力求解器和工具箱 */
  ENcloseH();
  ENclose();
}
```

例2 - Pascal

```
uses epanet2h;  { 利用工具箱提供的导入单元 }

procedure HydrantRating(MyNode: PChar; N: Integer;
   D: array of Single; var P: array of Single);

var
  i, nodeindex: Integer;
  t: LongInt;
  pressure: Single;

begin
{ 打开EPANETH工具箱和水力求解器 }
  ENopen('example2.inp', 'example2.rpt', '');
  ENopenH();

{ 获取感兴趣节点的索引 }
  ENgetnodeindex(MyNode, nodeindex);

{ 迭代计算所有需水量 }
  for i := 1 to N do
  begin

  { 设置节点需水量，初始化水力特性， }
  { 执行单一阶段分析，并检索压强     }
     ENsetnodevalue(nodeindex, EN_BASEDEMAND, D[i]);
     ENinitH(0);
     ENrunH(t);
     ENgetnodevalue(nodeindex, EN_PRESSURE, pressure);
     P[i] := pressure;
  end;

{ 关闭水力求解器和工具箱 }
  ENcloseH();
  ENclose();
end;
```

例2 - Visual Basic

```
'将EPANET2H.BAS 作为代码模块加入到工程

Sub HydrantRating(ByVal MyNode as String, N as Long,_
  D() as Single, P() as Single)

Dim i as Long
Dim nodeindex as Long
```

```
Dim t as Long
Dim pressure as Single

'打开EPANET工具箱和水力求解器
ENopen "example2.inp", "example2.rpt", ""
ENopenH

'获取感兴趣节点的索引
ENgetnodeindex MyNode, nodeindex

'迭代计算所有需水量
For i = 1 to N

  '设置节点需水量，初始化水力特性，
  '执行单个阶段运行，并检索压强
  ENsetnodevalue nodeindex, EN_BASEDEMAND, D(i)
  ENinitH 0
  ENrunH t
  ENgetnodevalue nodeindex, EN_PRESSURE, pressure
  P(i) = pressure
Next i

'关闭水力求解器和工具箱
ENcloseH
ENclose
End Sub
```

例3 – 满足最小余氯目标

本例说明工具箱怎样确定配水系统进口处的最低氯剂量，为了保证整个系统满足最小余氯。假设EPANETH输入文件包含了合适集合的动力学系数，它们描述了系统中氯衰减速率。在示例代码中，源头节点的ID标签包含在SourceID中，最小余氯目标通过Ctarget给出，目标仅仅在5日的开始阶段检查 (432,000秒)。为了保证代码更具有可读性，没有进行工具箱函数调用返回结果的错误检查。

例3 – C

```
#include "epanet2h.h"

float cl2dose(char *SourceID, float Ctarget)
{
  int i, nnodes, sourceindex, violation;
  float c, csource;
  long t, tstep;

  /* 打开工具箱并获取水力计算结果 */
  ENopen("example3.inp", "example3.rpt", "");
  ENsolveH();
```

```
 /* 获取节点总数和 */
 /* 源头节点索引   */
 ENgetcount(EN_NODES, &nnodes);
 ENgetnodeindex(SourceID, &sourceindex);

 /* 设置系统，进行氯分析 */
 /* (在本情况中，没有在输入文件中进行。) */
 ENsetqualtype(EN_CHEM, "Chlorine", "mg/L", "");

 /* 打开水质求解器 */
 ENopenQ();

 /* 开始搜索源头浓度 */
 csource = 0.0;
 do
 {

 /* 更新源头浓度到下一水平 */
   csource = csource + 0.1;
   ENsetnodevalue(sourceindex, EN_SOURCEQUAL, csource);

 /* 执行WQ模拟，检查目标是否违反情况 */
   violation = 0;
   ENinitQ(0);
   do
   {
     ENrunQ(&t);
     if (t > 432000)
     {
        for (i=1; i<=nnodes; i++)
        {
           ENgetnodevalue(i, EN_QUALITY, &c);
           if (c < Ctarget)
           {
              violation = 1;
              break;
           }
        }
     }
     ENnextQ(&tstep);

   /* 如果发现违反，结束WQ运行 */
   } while (!violation && tstep > 0);

 /* 如果发现违反，继续搜索 */
 } while (violation && csource <= 4.0);

 /* 关闭WQ求解器和工具箱 */
 ENcloseQ();
 ENclose();
 return csource;
}
```

例3 - Pascal

```
uses epanet2h; { 利用工具箱导入单元}

function cl2dose(SourceID: PChar; Ctarget: Single): Single;
var
  i, nlinks, nnodes, sourceindex, violation: Integer;
  c, csource: Single;
  t, tstep: LongInt;
begin
  { 打开工具箱，获取水力计算结果 }
  ENopen('example3.inp', 'example3.rpt', '');
  ENsolveH();

  { 获取节点总数     }
  { 以及源头节点索引 }
  ENgetcount(EN_NODES, nnodes);
  ENgetnodeindex(SourceID, sourceindex);

  { 为了分析氯而设置系统 }
  { (该情况中没有在输入文件中完成。) }
  ENsetqualtype(EN_CHEM, 'Chlorine', 'mg/L', '');

  { 打开水质求解器 }
  ENopenQ();

  { 开始搜索源头浓度 }
  csource := 0;
  repeat

  { 更新源头浓度到下一水平 }
    csource := csource + 0.1;
    ENsetnodevalue(sourceindex, EN_SOURCEQUAL, csource);

  { 执行WQ模拟，检查目标是否违反 }
    violation := 0;
    ENinitQ(0);
    repeat
      ENrunQ(t);
      if (t > 432000) then
      begin
         for i := 1 to nnodes do
         begin
            ENgetnodevalue(i, EN_QUALITY, c);
            if (c < Ctarget) then
            begin
               violation := 1;
               break;
            end;
         end;
      end;
      ENnextQ(tstep);
```

```
    { 如果发现违反，结束WQ运行 }
    until (violation = 1) or (tstep = 0);

  { 如果发现违反，继续搜索 }
  until (violation = 0) or (csource >= 4.0);

  { 关闭WQ求解器和工具箱 }
  ENcloseQ();
  ENclose();
  result := csource;
end;
```

例3 - Visual Basic

```
'将EPANET2H.BAS作为代码模块加入到工程

Function cl2dose(ByVal SourceID as String,_
 ByVal Ctarget as Single)as Single

Dim i as Long
Dim nlinks as Long
Dim nnodes as Long
Dim sourceindex as Long
Dim violation as Integer
Dim c as Single
Dim csource as Single
Dim t as Long
Dim tstep as Long

'打开工具箱，并获取水力结果
ENopen "example3.inp", "example3.rpt", ""
ENsolveH

'获取节点总数和源头节点索引
ENgetcount EN_NODES, nnodes
ENgetnodeindex SourceID, sourceindex

'为了氯分析而设置系统
'(该情况在输入文件中没有进行。)
ENsetqualtype EN_CHEM, "Chlorine", "mg/L", ""

'打开水质求解器
ENopenQ

'开始搜索源头浓度
csource = 0
Do

  '更新源头浓度到下一水平
  csource = csource + 0.1
  ENsetnodevalue sourceindex, EN_SOURCEQUAL, csource
```

```
  '执行WQ模拟，检查目标是否违反
  violation = 0
  ENinitQ 0
  Do
    ENrunQ t
    If t > 432000 Then
      For i = 1 to nnodes
         ENgetnodevalue i, EN_QUALITY, c
         If c < Ctarget Then
            violation = 1
            Exit For
          End If
       Next i
    End If
    ENnextQ tstep

  '终止WQ执行，如果违反发现
  Loop Until (violation = 1) Or (tstep = 0)

'继续搜索，如果违反发现
Loop Until (violation = 0) Or (csource >= 4.0)

'关闭WQ求解器和工具箱
ENcloseQ
ENclose
cl2dose = csource
End Function
```

5．效率事项

当对同一管网进行多重分析时(正如优化过程中那样)，不需要重复调用ENsolveH，而是重复采用ENrunH - ENnextH循环，见下：

```
 int stop;
 long t, tstep;
 ENopenH();
 stop = 0;
 do
 {
   setparams();
   ENinitH(0);
   do
   {
     ENrunH(&t);
     evalresults(t, &stop);
     ENnextH(&tstep);
   } while (tstep > 0 && !stop);
 } while (!stop);
 ENcloseH();
```

在以上代码中，setparams()是由用户定义的函数，其中以相同方式从一次迭代到下一次修改了管网。另一个用户定义函数evalresults()，将估计时刻t的结果，设置停止标志的数值，作为迭代结束的信号。注意通过ENinitH()的参数为0，说明不需要将水力结果保存到文件，因为当它们生成时直接被使用。这也将加速计算。

当需要进行重复水质运行时，利用相同的水力特性，那么调用ENsolveH一次，产生和保存水力结果，并为水质运行而利用以上类似编码(应使用ENopenQ, ENinitQ, ENrunQ, ENnextQ, 和ENcloseQ函数)。

6. 工具箱参考

6.1 错误代码

代码	描述
0	无错误
101	没有充分内存
102	没有需要处理的管网数据
103	水力求解器没有初始化
104	没有可用的水力结果
105	水质求解器没有初始化
106	没有可以报告的结果
110	不能够求解水力方程组
120	不能够求解WQ迁移方程组
200	输入文件中有一处或者多处错误
202	函数调用中具有非法数值
203	函数调用中具有未定义的节点
204	函数调用中具有未定义的管段
205	函数调用中具有未定义的时间模式
207	试图控制止回阀
223	管网中没有充分的节点
224	管网中不存在水池或者水库
240	函数调用中具有未定义的水源
241	函数调用中具有未定义的控制语句
250	函数参数格式非法
251	函数调用中具有非法参数代号

301	相同的文件名
302	不能够打开输入文件
303	不能够打开报告文件
304	不能够打开二进制输出文件
305	不能够打开水力文件
306	非法水力文件
307	不能够阅读水力文件
308	不能够将结果保存到文件
309	不能够将报告写入文件

6.2 警告代码

代码	描述
1	系统水力上不平衡-在允许试算次数内不能够收敛到水力结果
2	系统在水力上可能不稳定--在所有管段保持固定之后，才能达到水力收敛
3	系统不连通—具有正需水量的一处或者多处节点，没有连接到任何一个供水水源
4	水泵难以提供充分的流量或者扬程—一台或者多台水泵被迫关闭（由于不充分的扬程）或者运行超过了最大额定流量
5	阀门不能够输送充分的流量—即使当完全开启时，一处或者多处流量控制阀不能够输送需要的流量
6	系统出现负压—具有正需水量的一处或者多处连接节点出现负压

6.3 文件描述

支撑文件

EPANETH程序员工具箱源函数包含在几个文件中，为了支持不同的编程语言。

文件名	目的
epanet2h.h	C/C++的头文件
epanet2h.lib	Borland C/C++的库文件
epanet2h.lib	Microsoft Visual C++的库文件
epanet2h.pas	Delphi (Pascal)的导入单元
epanet2h.bas	Visual Basic的声明模块

输入文件

输入文件是标准的EPANETH输入数据文件，描述了分析的系统(参见输入文件格式)。它可以利用工具箱在外部创建，然后用于开发的应用程序；或者通过应用程序本身创建。首先将文件名

用到ENopen函数。在利用ENopen打开输入文件之前，不可以使用其它工具箱函数(除了ENepanet)。在工具箱系统利用ENclose函数关闭之前，可以访问与输入文件相关的数据。

水力文件

水力文件是非格式二进制文件，用于存储水力分析的结果。所有时段结果的存储，包括当特殊水力时间发生时的瞬时结果（例如由于满足了控制条件，开启或者关闭水泵和水池)。

通常作为临时文件，在ENclose函数调用之后删除。可是，如果调用了ENsavehydfile函数，它将可以永久保存。

如果命令HYDRAULICS USE *filename*出现在输入文件的[OPTIONS]节中；或者如果调用了ENusehydfile函数，可以使用原先保存的水力文件。

当工具箱函数ENsolveH用于水力分析时，结果自动保存到水力文件。当使用了函数集ENinitH - ENrunH - ENnextH， ENinitH的*saveflag*参数，确定结果是否保存。保存水力结果的需求是依赖于应用程序的。如果继续进行水质分析，它们必须保存到水力文件。

报告文件

报告文件是用于ENopen (或者ENepanet)函数的第二个文件名。它用于记录输入文件处理过程中的任何错误信息，以及记录水力模拟过程中产生的所有状态信息。此外，如果调用ENreport函数，结果报告也写入该文件。报告的格式受到输入文件[REPORT]节的语句，以及通过包含在调用ENsetreport函数中类似语句的控制。仅仅指定统一报告时间间隔的结果写入到该文件。

输入文件[REPORT]节中包括命令MESSAGES NO；或者调用工具箱函数("MESSAGES NO")，可将所有错误和警告信息写入到报告文件。

通过在输入文件[REPORT]节中包含命令FILE *filename*，或者调用工具箱函数ENsetreport ("FILE *filename*")，可将格式化报告演算到不同的文件，而不是报告文件，其中*filename*为将要利用的文件名。

输出文件

输出文件是一个非格式化二进制文件，用于以均匀报告间隔存储水力和水质结果(参见输出文件格式)。这是用在ENopen函数中的第三个文件名。如果将空字符串("")用作它的名称，那么将使用一个草稿性临时文件。否则在调用ENclose函数之后，将保存输出文件。如果输入结果需要进一步处理，保存该文件是有益的。如果不进行水质分析，函数ENsaveH将水力结果转换到输出文件。利用ENsolveQ执行水质分析，水力和水质结果将自动保存到该文件。如果函数集ENinitQ - ENrunQ - ENnextQ用于执行水质分析，那么仅仅在ENinitQ的*saveflag*参数设置为1时保存结果。此外，将结果保存到输出文件的需求依赖应用程序。如果采用ENreport产生格式化输出报告，那么必须首先将结果保存到输出文件。

6.3.1 输入文件格式

EPANETH工具箱与描述被分析管网的输入文本文件一起工作。文件通过节组织，每一节以方括号中的关键词开始。各种关键词如下。

管网组件	系统操作	水质	选项和报告
[TITLE]	**[CURVES]**	**[QUALITY]**	**[OPTIONS]**
[JUNCTIONS]	**[PATTERNS]**	**[REACTIONS]**	**[TIMES]**
[RESERVOIRS]	**[ENERGY]**	**[SOURCES]**	**[REPORT]**
[TANKS]	**[STATUS]**	**[MIXING]**	
[PIPES]	**[CONTROLS]**		
[PUMPS]	**[RULES]**		
[VALVES]	**[DEMANDS]**		
[EMITTERS]			

节的次序并不重要。可是，当一个节需要节点或者管段作为参考时，必须先定义[JUNCTIONS], [RESERVOIRS]，[TANKS]，[PIPES]，[PUMPS]或者[VALVES]节。于是建议首先放置这些节。

每一节包含一行或者多行数据。可以在文件中的任何位置设置空行，分号用于说明其后面是注释而不是数据。每行最大显示字符为255个。

ID标签用于确定节点、管段、曲线和格式，可以为多达15个字符和数字的任意组合。

[TITLE]

目的:
为分析管网设置描述性标题。

格式:
任何数量的文本行。

备注:
[TITLE]节是可选的。

[JUNCTIONS]

目的:
定义管网中包含的连接节点。

格式:
每一连接节点为一行，包含了：
- ID标签
- 标高，m (ft)
- 基本需水量 (流量单位) (可选)
- 需水量模式ID (可选)

备注:

1. [JUNCTIONS]节至少需要一个连接节点。
2. 如果没有提供需水量模式，那么连接节点需水量遵从[OPTIONS]节提供的缺省需水量模式，或者如果没有指定缺省模式，采用模式1。如果缺省模式（或者模式1）不存在，那么需水量保持恒定。
3. 需水量可在 [DEMANDS]节中输入，每一连接节点可包含多种需水量类型。

示例:

```
[JUNCTIONS]
;ID    标高    需水量    模式
;-------------------------------
J1    100     50      Pat1
J2    120     10            ;使用缺省需水量模式
J3    115                   ;在该连接节点没有需水量
```

[RESERVOIRS]

目的:

定义管网中包含的所有水库节点。

格式:

每一水库占一行，包括：

- ID标签
- 水头，m (ft)
- 水头模式ID (可选)

备注:

1. 水头是水库中的测压管水头（标高 + 压力水头）。
2. 水头模式用于表达水库水头随时间的变化。
3. 管网中必须包含至少一座水库或者水池。

例子:

```
[RESERVOIRS]
;ID    水头     模式
;---------------------
R1    512             ;水头保持恒定
R2    120    Pat1     ;水头随时间变化
```

[TANKS]

目的:

定义管网中包含的所有水池节点。

格式:

每一水池占一行，包括：

- ID标签
- 池底标高，m (ft)
- 初始水位，m (ft)

- 最低水位，m (ft)
- 最高水位，m (ft)
- 公称直径，m (ft)
- 最小容积，立方米(立方ft)
- 容积曲线ID (可选的)

备注:

1. 水面标高等于池底标高加水位。
2. 非圆筒形水池通过在[CURVES]节中指定容积与水深曲线模拟。
3. 如果提供了容积曲线，直径可以是任何非零数值。
4. 对于圆筒形水池，或者提供了容积曲线，最小容积（在最低水位时水池容积）可以为零。
5. 管网必须至少包含一座水池或者水库。

例子:

```
[TANKS]
;ID   标高   初始水位   最低水位   最高水位   直径   最小容积   容积曲线
;-----------------------------------------------------------
;圆筒形水池
T1    100    15      5      25     120   0
;具有任意直径的非圆筒形水池
T2   100    15      5      25     1     0       VC1
```

[PIPES]

目的:

定义包含在管网中的所有管道。

格式:

每一管道占一行，包含：

- ID标签
- 起始节点ID
- 终止节点ID
- 长度，m (ft)
- 直径，mm (英寸)
- 粗糙系数
- 局部损失系数
- 状态(**OPEN**，**CLOSE**或**CV**)

备注:

1. Hazen-Williams和Chezy-Manning水头损失公式中粗糙系数无量纲，Darcy-Weisbach公式中，单位为mm（毫英尺）。水头损失公式的选择在[OPTIONS]节提供。
2. 设置状态为**CV**，意味着管道包含了限制流向的止回阀。
3. 如果局部损失系数为0，管道状态为OPEN，那么这两项可以在输入行中省略。

例子:

```
[PIPES]
;ID   节点1  节点2   长度    直径  粗糙系数  局部损失    状态
;-------------------------------------------------------------
P1    J1     J2     1200      12      120       0.2    OPEN
P2    J3     J2      600       6      110         0     CV
P3    J1     J10    1000      12      120
```

[PUMPS]

目的:

定义包含在管网中的所有水泵。

格式:

每一水泵占一行，包含：

- ID标签
- 起始节点ID
- 终止节点ID
- 关键词和数值 (可以重复进行)

备注:

1. 关键词包括：
   - **POWER** – 恒定能量水泵的功率，kw (hp)
   - **HEAD** – 描述水泵扬程与流量关系的曲线ID
   - **SPEED** – 相对转速设置 (常规转速为1.0；0意味着水泵关闭)
   - **PATTERN** – 描述转速设置怎样随时间变化的时间模式ID
2. 每台水泵必须提供**POWER**或者**HEAD**。其它关键词是可选的。

例子:

```
[PUMPS]
;ID    节点1    节点2   属性
;-----------------------------------------
Pump1   N12     N32     HEAD Curve1
Pump2   N121    N55     HEAD Curve1  SPEED 1.2
Pump3   N22     N23     POWER 100
```

[VALVES]

目的:

定义包含在管网中的所有控制阀门。

格式:

每一阀门占一行，包含：

- ID标签
- 起始节点ID
- 终止节点ID
- 直径，mm (英寸)
- 阀门类型
- 阀门设置
- 局部损失系数

备注:
1. 阀门类型和设置包括:

阀门类型	设置
PRV (减压阀)	压强，m (psi)
PSV (稳压阀)	压强，m (psi)
PBV (压力制动阀)	压强，m (psi)
FCV (流量控制阀)	流量(流量单位)
TCV (节流控制阀)	损失系数
GPV (常规阀门)	水头损失曲线ID

隔断阀和止回阀认为是管道的一部分，而不是独立的控制阀门组件(见[PIPES])。

[EMITTERS]

目的:
定义模拟为扩散器（喷嘴或者孔口）的连接节点。

格式:
每一扩散器为一行，包含了:
- 连接节点ID标签
- 流量系数，1米(1psi)压强降落时的流量单位

备注:
1. 扩散器用于模拟通过喷头或者管道渗漏的流量。
2. 扩散器的出流等于流量系数与连接节点压力上升幂的乘积。
3. 幂可以利用[OPTIONS]节中的EMITTER EXPONENT选项指定。缺省幂为0.5, 通常用于喷嘴。
4. 程序结果中实际报告的需水量，包括连接节点的常规需水量和通过扩散器的流量之和。
5. [EMITTERS]节是可选的。

[CURVES]

目的:
定义数据曲线及其X，Y点。

格式:
每一曲线的每一X，Y点占一行，包含了:
- 曲线ID标签
- X值
- Y值

备注:

1. 曲线可用于表示以下关系:
   - 水泵扬程与流量
   - 水泵效率与流量
   - 水池容积与水深
   - 常规阀门水头损失与流量
2. 曲线点必须以X数值递增方式输入（从小到大）。
3. 如果利用EPANETH的Windows版本的输入文件，那么添加一个包含了曲线类型和描述的注释，以分号分隔，直接在曲线的进口上方，确保这些像直接显示在EPANETH的曲线编辑器中。曲线类型包括水泵、效率、容积和水头损失。示例见下。

例子:

```
[CURVES]
;ID   流量     水头
;水泵: 水泵1曲线
C1    0       200
C1    1000    100
C1    3000    0

;ID   流量     效率
{PRIVATE ForeHelp xCell
ID=1446};效率:
E1    200     50
E1    1000    85
E1    2000    75
E1    3000    65
```

[PATTERNS]

目的:

定义时间模式。

格式:

每一模式为一行或者多行，包含了:

- 模式ID标签
- 一个或者多个乘子

备注:

1. 乘子定义了一些基本量怎样在每一时段进行调整（例如需水量）。
2. 所有模式共享在[TIMES]节中定义的相同时段间隔。
3. 每一模式可以具有不同量的时段。
4. 当模拟时间超过模式长度，模式返回到它的第一时段。
5. 利用多行，以包含每一模式的所有乘子。

例子:

```
[PATTERNS]
;模式 P1
P1   1.1   1.4   0.9   0.7
P1   0.6   0.5   0.8   1.0
;模式 P2
P2   1    1    1    1
P2   0    0    1
```

[ENERGY]

目的:

定义用于计算水泵提升能量和成本的参数。

格式:

GLOBAL **PRICE/PATTERN/EFFIC** *value*
PUMP *PumpID* **PRICE/PATTERN/EFFIC** *value*
DEMAND CHARGE *value*

备注:

1. 第一种格式用于设置所有水泵的能量价格、价格模式和提升效率的全局缺省数值。
2. 第二种格式对应于特定水泵，重载全局缺省。
3. 参数定义如下：
   PRICE = 平均成本每kW时，
   PATTERN = 描述能量价格怎样随时间变化的时间模式ID标签，
   EFFIC = 全局设置的单一百分比效率，或者特定水泵的效率曲线ID标签，
   DEMAND CHARGE = 模拟时段每最大kW耗能的成本。
4. 缺省全局水泵效率为75%，且缺省能量价格为0。
5. 该节的所有输入是可选的。反斜杠 (/)旁的项表示允许选项。

例子:

```
[ENERGY]
GLOBAL PRICE     0.05   ;设置全局能量价格
GLOBAL PATTERN    PAT1   ;以及日历时间模式
PUMP   23  PRICE  0.10   ;重载水泵23的价格
PUMP   23  EFFIC  E23    ;将效率曲线赋给水泵23
```

[STATUS]

目的:

定义模拟开始时被选择管段的初始状态。

格式:

被控制的一条管段为一行，包含了：

- 管段ID标签
- 状态或者设置

备注:

1. 该节没有列出的管段具有缺省状态**OPEN** (对于管道和水泵)或者**ACTIVE** (对于阀门)。
2. 在该节所赋状态值可以为**OPEN**或者**CLOSED**。对于控制阀门（例如PRV、FCV等)，这意味着阀门为全开或者全关，在它的控制设置中不是活动的。
3. 设置值可以为水泵的转速设置，或者阀门的阀门设置。
4. 管道的初始状态也可以在[PIPES]节中设置。
5. 止回阀的状态不能够预先设置。
6. 利用[CONTROLS]或者[RULES]，改变模拟中一些点的状态或者设置。
7. 如果一个**CLOSED**或者**OPEN**的控制阀门再次变为ACTIVE，那么必须在控制或者规则中指定它的压力（或者流量）设置，以便重新激活它。

例子:

```
[STATUS]
; 管段    状态/设置
;----------------------
  L22    CLOSED       ;管段L22关闭
  P14    1.5          ;水泵P14的转速
  PRV1   OPEN         ;PRV1被迫开启
                       ;(重载常规运行)
```

[DEMANDS]

目的:

[JUNCTIONS]节的补充，定义连接节点的多种需水量。

格式:

连接节点每一类需水量为一行，包含了：

- 连接节点ID标签
- 基本需水量(流量单位)
- 需水量模式ID (可选的)
- 需水量类型名称，之前为分号（可选的）

备注:

1. 仅用于需水量需要变化的连接节点，或者补充[JUNCTIONS]节中的输入。
2. 本节中的数据替换了相同连接节点在[JUNCTIONS]节中输入的任何需水量。
3. 每一连接节点可以输入无限数量的需水量类型。
4. 如果没有提供需水量模式，那么连接节点需水量遵从由[OPTIONS]节提供的缺省需水量模式；或者如果没有提供缺省模式，采用模式1。如果缺省模式（或者模式1）不存在，那么需水量保持恒定。

例子:

```
[DEMANDS]
;ID    需水量    模式    类型
;-------------------------------
J1     100       101     ;住宅
J1     25        102     ;学校
J256   50        101     ;住宅
```

[CONTROLS]

目的:
定义了根据简单条件修改管段的简单控制。

格式:
每一控制为一行，形式为：

```
LINK linkID status IF NODE nodeID ABOVE/BELOW value
LINK linkID status AT TIME time
LINK linkID status AT CLOCKTIME clocktime AM/PM
```

式中：

linkID	=	管段ID标签
status	=	**OPEN**或者**CLOSED**，水泵转速设置，或者控制阀门设置
nodeID	=	节点ID标签
value	=	连接节点压强或者水池水位
time	=	模拟开始后的时间，以小时计
clocktime	=	24时钟表时间 (小时:分钟)

备注:
1. 简单控制将根据水池水位、连接节点压强、进入模拟的时间或者一日内的时间，改变管段状态或者设置。
2. 为了指定管段状态和设置，尤其对于控制阀门，参见[STATUS]节的备注。

例子:

```
[CONTROLS]
;如果Tank 23的水位超过20 ft，关闭Link 12。
LINK 12 CLOSED IF NODE 23 ABOVE 20

;如果Node 130的压强低于30 psi，开启Link 12
LINK 12 OPEN IF NODE 130 BELOW 30

;水泵PUMP02的转速比在进入模拟后的第16小时设置为1.5
LINK PUMP02 1.5 AT TIME 16

;模拟过程中，Link 12在上午10时关闭，下午8时开启
LINK 12 CLOSED AT CLOCKTIME 10 AM
LINK 12 OPEN AT CLOCKTIME 8 PM
```

[RULES]

目的:
根据条件的组合定义和规则控制，修改管线。

格式:
每一规则是以下系列格式语句：

```
RULE ruleID
IF  condition_1
AND condition_2
OR  condition_3
AND condition_4
etc.
THEN action_1
AND  action_2
etc.
ELSE action_3
AND  action_4
etc.
PRIORITY value
```

其中：

ruleID	=	赋给规则的ID标签
conditon_n	=	条件语句
action_n	=	行动语句
priority	=	优选数值(例如编号从1到5)

备注:
1. 仅仅需要**RULE**, **IF**和**THEN**部分；其他是可选的。
2. 当混合了**AND**和**OR**语句，**OR**算子与**AND**相比，具有较高优先权，即，
   ```
   IF A or B and C
   ```
   等价于
   ```
   IF (A or B) and C.
   ```
   如果想解释为
   ```
   IF A or (B and C)
   ```
   那么需要利用两条规则表达
   ```
   IF A THEN ...
   IF B and C THEN ...
   ```
3. 当两个或者多个规则对一条管段发生冲突时，**PRIORITY**数值用于确定使用的规则。与具有这样数值的相比，没有优先性数值的规则总是具有较低优先性。对于具有相同优先性数值的规则，首先出现的规则给出较高优先性。

例子:
```
[RULES]

RULE 1
IF   TANK   1 LEVEL ABOVE 19.1
THEN PUMP 335 STATUS IS CLOSED
AND  PIPE 330 STATUS IS OPEN

RULE 2
IF   SYSTEM CLOCKTIME >= 8 AM
AND  SYSTEM CLOCKTIME < 6 PM
AND  TANK 1 LEVEL BELOW 12
THEN PUMP 335 STATUS IS OPEN
```

```
RULE 3
IF   SYSTEM CLOCKTIME >= 6 PM
OR   SYSTEM CLOCKTIME < 8 AM
AND  TANK 1 LEVEL BELOW 14
THEN PUMP 335 STATUS IS OPEN
```

规则条件语句

基于规则控制中条件语句采用形式为：

```
object id attribute relation value
```

式中：

```
object     =  管网对象的类型
id         =  对象的ID标签
attribute  =  对象的属性或者特性
relation   =  关系操作符
value      =  属性值
```

一些条件语句示例为：

```
JUNCTION 23 PRESSURE > 20
TANK T200 FILLTIME BELOW 3.5
LINK 44 STATUS IS OPEN
SYSTEM DEMAND >= 1500
SYSTEM CLOCKTIME = 7:30 AM
```

对象关键词可以为：

```
NODE          LINK        SYSTEM
JUNCTION      PIPE
RESERVOIR     PUMP
TANK          VALVE
```

当条件中使用了SYSTEM，不提供ID。

以下属性可用于节点类型对象：

```
DEMAND
HEAD
PRESSURE
```

以下属性可用于水池类型：

```
LEVEL
FILLTIME (水箱注水所需小时数)
DRAINTIME (水箱放水所需小时数)
```

以下属性可用于管段类型对象：

```
FLOW
STATUS (OPEN, CLOSED或者ACTIVE)
SETTING (水泵转速或者阀门设置)
```

SYSTEM对象可使用以下属性：
DEMAND (总系统需水量)
TIME (模拟开始后的小时数，可表达为小数数值或者小时:分钟格式)
CLOCKTIME (带有AM或者PM后缀的24小时钟表时间)

关系操作符包括：

=	IS
<>	NOT
<	BELOW
>	ABOVE
<=	
>=	

规则行动语句

基于规则控制中的行动语句采用形式为：

```
object id STATUS/SETTING IS value
```

式中：

object	=	LINK, PIPE, PUMP或者VALVE关键词
id	=	对象的ID标签
value	=	状态条件(OPEN或者CLOSED)，水泵转速设置或者阀门设置

行动语句的一些例子为：

```
LINK 23 STATUS IS CLOSED
PUMP P100 SETTING IS 1.5
VALVE 123 SETTING IS 90
```

指定管段状态和设置，尤其控制阀门常规使用的，见[STATUS]节的备注。

[QUALITY]

目的:
定义节点的初始水质。

格式:
每一节点为一行，包含了：
- 节点ID标签
- 初始水质

备注:
1. 对于没有列出的节点，水质假设为零。
2. 水质表示了化学成分的浓度，水龄的小时数，或者源头跟踪的百分比。
3. [QUALITY]节是可选的。

[REACTIONS]

目的:
定义了管网中化学反应相关的参数。

格式:

```
ORDER BULK/WALL/TANK    value
GLOBAL BULK/WALL        value
BULK/WALL/TANK          pipeID  value
LIMITING POTENTIAL      value
ROUGHNESS CORRELATIONR  value
```

备注:
1. 注意增长反应系数采用正值，衰减系数为负值。
2. 所有反应系数的时间单位为1/日。
3. 本节的所有输入是可选的。反斜杠(/)后的项说明了允许选项。

ORDER用于设置分别发生在主流水体、管壁或者水池中的反应级数。管壁反应的数值必须为0或者1。如果没有提供，缺省反应级数为1.0。

GLOBAL用于设置所有主流水体反应系数（管道和水池）或者所有管壁系数的全局数值。缺省值为零。

BULK, WALL和TANK用于对指定管道和水池重新设置全局反应系数。

LIMITING POTENTIAL指定了反应速率正比于当前浓度和一些限制值之间的差异。

ROUGHNESS CORRELATION将使所有缺省管壁反应系数，以以下方式，相关于管道粗糙系数：

水头损失公式	粗糙相关性
Hazen-Williams	F/C
Darcy-Weisbach	F/log(e/D)
Chezy-Manning	F*n

式中F——粗糙系数相关性；
C——Hazen-Williams C因子；
e——Darcy-Weisbach粗糙系数；
D——管道直径；
n——Chezy-Manning粗糙系数。
这种方式计算的缺省值能够通过利用WALL格式，对任何使用特定数值的管道重载。

例子:

```
[REACTIONS]
ORDER WALL    0    ;管壁反应为零级
GLOBAL BULK  -0.5  ;全局主流衰减系数
GLOBAL WALL  -1.0  ;全局管壁衰减系数
WALL   P220  -0.5  ;指定管道管壁系数
WALL   P244  -0.7
```

[SOURCES]

目的:
定义了水质源头的位置。

格式:
每一水质源头为一行，包含了：
- 节点ID标签
- 源头类型(**CONCEN, MASS, FLOWPACED**或**SETPOINT**)
- 基线源头强度
- 时间模式ID (可选的)

备注:
1. **MASS**类型源头的强度计量为质量流量每分。所有其它类型测试源头强度，以浓度单位计。
2. 通过指定时间模式，源头强度可以设为随时间而变化。
3. **CONCEN**源头：
   - 表示任何进入节点的外部源头进流浓度
   - 仅当节点具有负需水量时使用（水在该节点进入管网）
   - 如果节点为连接节点，报告的浓度为混合了源头流量和从管网其它点来的进流结果
   - 如果节点为水库，报告的浓度为源头浓度
   - 如果节点为水池，报告的浓度为水池的内部浓度
   - 适合于表示源头供水或者处理厂的节点 (例如赋以负需水量的水库或者节点)
   - 不适用于同时具有进流/出流的蓄水池。
4. **MASS, FLOWPACED**或者**SETPOINT**源头：
   - 表示了注射源头，这里物质被直接注射到管网，忽略了节点的需水量多少
   - 以下方式影响了离开节点到管网其它位置的水：
     - **MASS**注入，将质量流量添加到节点进流
     - **FLOWPACED**注入，将固定浓度添加到节点进流浓度中
     - **SETPOINT**注入，固定了任何离开节点的水流浓度（只要进流来的浓度低于设置值)
   - 在连接节点或者水库注射源头的报告浓度，是在应用注射之后的结果浓度；具有注射源头的水池报告浓度为水池的内部浓度
   - 适合于模拟示踪剂的直接注入，或者进入管网的消毒剂，或者模拟污染物入侵。
5. 对于模拟水龄或者源头跟踪，[SOURCES]节是不需要的。

例子:
```
[SOURCES]
;节点   类型     强度   模式
;------------------------------
 N1   CONCEN    1.2      Pat1    ;浓度随时间变化
 N44  MASS      12               ;恒定质量注入
```

[MIXING]

目的:
确定控制蓄水池混合的模型。

格式:
每一水池为一行，包含了：
- 水池ID标签
- 混合模型(**MIXED, 2COMP, FIFO**或者**LIFO**)
- 室的容积(分数)

备注:
1. 混合模型包括：
   - 完全混合(**MIXED**)
   - 双室混合(**2COMP**)
   - 柱塞流(**FIFO**)
   - 堆栈式柱塞流(**LIFO**)
2. 室容积参数仅用于双室模型，表示了总容积贡献于进水/出水室的分数。
3. [MIXING]节是可选的。本节没有描述到的水池假设为完全混合。

例子:
```
[MIXING]
;水池      模型
;--------------------
T12        LIFO
T23        2COMP    0.2
```

[OPTIONS]

目的:
定义各种模拟选项。
格式:

UNITS	**CFS/GPM/MGD/IMGD/AFD/ LPS/LPM/MLD/CMH/CMD**
HEADLOSS	**H-W/D-W/C-M**
HYDRAULICS	**USE/SAVE** filename
QUALITY	**NONE/CHEMICAL/AGE/TRACE id**
VISCOSITY	value
DIFFUSIVITY	value
SPECIFIC GRAVITY	value
TRIALS	value
ACCURACY	value
UNBALANCED	**STOP/CONTINUE/CONTINUE n**
PATTERN	id
DEMAND MULTIPLIER	value
EMITTER EXPONENT	value
TOLERANCE	value
MAP	filename

UNITS设置了流量被表达的单位：

LPS—升/秒
LPM—升/分
MLD—百万升/日
CMH—立方米/小时
CMG—立方米/日
CFS—立方英尺/秒
GPM—加仑/分
MGD—百万加仑/日
IMGD—英制MGD
AFD—英亩-英尺/日

如果流量单位为升或者立方米，那么公制单位也必须用于所有其它输入量。对于CFS, GPM, MGD, IMGD和AFD，其它输入量表达为美制单位。缺省流量单位为GPM。

HEADLOSS选择了用于计算通过管道水流的水头损失公式。包括Hazen-Williams（H-W），Darcy-Weisbach(D-W)或Chezy-Manning (C-M)公式。缺省为H-W。

HYDRAULICS选项允许将当前水力结果SAVE（保存）到文件，或者USE（利用）以前保存的水力结果。当研究仅仅影响水质行为的因子时，很有用。

QUALITY选择了执行水质分析的类型。选择包括NONE（无），CHEMICAL（化学药剂），AGE（水龄）和TRACE（跟踪）。在CHEMICAL情况中，化合物的实际名称之后为其浓度单位（例如CHLORINE mg/L）。如果TRACE被选择，必须跟踪节点的ID标签。缺省选项为NONE（没有水质分析）。

VISCOSITY是被模拟流体的运动粘度，相对于20摄氏度时的情况（1.0厘斯）。缺省值为1.0。

DIFFUSIVITY是化合物的分析扩散系数，相对于水中的氯情况。缺省值为1.0。扩散系数仅仅适用于管壁反应中考虑质量转换限制时。数值0将造成EPANETH忽略质量转换限制。

SPECIFIC GRAVITY是被模拟流体密度与4摄氏度水的密度之比（无量纲）。

TRIALS是在每一模拟水力时间步长中，求解管网水力特性使用的最大试算次数。缺省为40。

ACCURACY指定了确定何时达到水力结果的收敛准则。当所有流量总和改变，来自原先求解除以所有管段的总流量低于该数值时，试算中止。缺省为0.001。

UNBALANCED确定了何时进行，如果水力结果在指定的TRIAL数值内不能够达到，对于水力时间步长来模拟。“STOP”将在该点终止整个分析。“CONTINUE”将在公布警告消息的情况下继续分析。“CONTINUE n”将在另外“n”次试算中搜索结果，所有管线状态保持它们的当前设置。模拟将在该点继续，关于收敛是否达到，具有消息公布。缺省选项为“STOP”。

PATTERN提供了用于没有指定需水量模式的所有节点缺省需水量模式的ID标签。如果没有这样的模式在[PATTERNS]节中存在，那么通过缺省的模式，包含一个等于1.0的单一乘子。如果没有使用该选项，总体缺省需水量模式具有标签“1”。

DEMAND MULTIPLIER用于调整所有连接节点的基本需水量数值，以及所有需水量的类型。例如，数值2为两倍的基准需水量，而数值0.5将为它们的一半。缺省值为1.0。

EMITTER EXPONENT指定了当计算扩散器的流量时，节点压力上升的幂指数。缺省为0.5。

MAP用于提供包含了管网节点坐标的文件名称，以便绘制管网地图。对于任何水力或者水质计算这是无用的。

TOLERANCE是水质水平精度。对于所有水质分析（化合物、水龄（以小时计），或者源头跟踪（以百分比计）），缺省值为0.01。

备注:

1. 如果在本节没有明确指定，所有选项假设为缺省值。
2. 反斜杠(/)后的项为允许选项。

例子:

```
[OPTIONS]
UNITS       CFS
HEADLOSS     D-W
QUALITY      TRACE  Tank23
UNBALANCED  CONTINUE  10
```

[TIMES]

目的:

定义用于模拟的各种时间步长参数。

格式:

DURATION	value (units)
HYDRAULIC TIMESTEP	value (units)
QUALITY TIMESTEP	value (units)
RULE TIMESTEP	value (units)
PATTERN TIMESTEP	value (units)
PATTERN START	value (units)
REPORT TIMESTEP	value (units)
REPORT START	value (units)
START CLOCKTIME	value (**AM/PM**)
STATISTIC	**NONE/AVERAGED/ MINIMUM/MAXIMUM/ RANGE**

备注:

1. 单位可以为**SECONDS (SEC), MINUTES (MIN), HOURS**或者**DAYS。**缺省为小时。
2. 如果没有提供单位，那么时间值可以表达为小数小时或者小时:分钟格式。
3. [TIMES]节中的所有输入是可选的。反斜杠(/)后的项为可选项。

DURATION是模拟的历时。设为0来运行简单的瞬时分析。缺省为0。

HYDRAULIC TIMESTEP定义了管网新的水力状态计算频率。如果它大于PATTERN或者REPORT时间步长，将自动降低。缺省为1小时。

QUALITY TIMESTEP用于跟踪水质通过管网变化的时间步长。缺省为水力时间步长的1/10。

RULE TIMESTEP用于检查水力时间步长之间，基于规则控制引起的系统状态变化的时间步长。缺省为1/10的水力时间步长。

PATTERN TIMESTEP是所有事件模式中时段之间的间隔。缺省为1小时。

PATTERN START是所有模式开始时的时间分量。例如，6小时的数值将开始模拟，在时段中的每一模式，对应于6小时。缺省为0。

REPORT TIMESTEP设置了输出结果被报告的时间间隔。缺省为1小时。

REPORT START是进入模拟的时间长度，此时输出结果开始报告。缺省为0。

START CLOCKTIME是模拟开始的钟表时间（例如3:00 PM）。缺省为子夜12:00 AM。

STATISTICS确定了在产生模拟结果的时间序列中，统计后处理的类型。AVERAGED报告了时间平均结果集合，MINIMUM仅仅报告最小值，MAXIMUM为最大值，以及RANGE报告了最大值和最小值之间的差异。NONE报告了所有节点和管段量的完整时间序列，它为缺省的。

例子:

```
[TIMES]
DURATION           240 HOURS
QUALITY TIMESTEP   3 MIN
QUALITY TIMESTEP   0:03
REPORT START       120
START CLOCKTIME    6:00 AM
```

[REPORT]

目的:

描述了从模拟产生输出报告的内容。

格式:

```
PAGESIZE        value
FILE            filename
STATUS          YES/NO/FULL
SUMMARY         YES/NO
MESSAGES        YES/NO
ENERGY          YES/NO
NODES           NONE/ALL/node1 node2 ...
LINKS           NONE/ALL/link1 link2 ...
variable        YES/NO
variable        BELOW/ABOVE/PRECISION  value
```

备注:

1. 如果没有在本节指定，所有选项假设为缺省值。
2. 反斜杠(/)后的项为可选项。
3. 对于任何节点或者管段，不报告缺省值，如果希望报告这些项的结果，因此必须提供**NODES**或者**LINKS**选项。

PAGESIZES设置了输出报表中每一页的行数。缺省为0，意味着事实上每一页没有行数限制。对于将要写入的输出报告（在EPANETH的Windows版本中忽略），**FILE**提供了文件的名字。

STATUS确定了应怎样生成水力状态报告。如果YES被选择，在模拟的每一时间步长中改变状态的所有管网组件将输出到报告。如果FULL被选择，那么也将包括每一水力分析的每一试算中的信息输出到报告。详细水平仅仅对于调试管网是有用的，这时水力不平衡。缺省为NO。

SUMMARY确定了管网组件数量的总结表，以及产生的关键分析选项。缺省为YES。

MESSAGE确定了水力/水质分析过程中产生的错误和警告信息是否写入报告文件。缺省为YES。

ENERGY确定是否提供表格报告平均能量使用和每一台水泵的成本。缺省为NO。

NODES确定了哪些节点将被报告。可以列出单个节点ID标签，或者利用关键词NONE或者ALL。额外NODES行可用于继续该表。缺省为NONE。

LINKS确定了哪些管段将被报告。可以列出单个管段ID标签，或者使用关键词NONE或者ALL。额外LINKS行可用于继续该表。缺省为NONE。

“参数”报告选项，用于确定报告哪些量，多少小数位被显示，哪种类型的过滤用于限制输出报告。可以被报告的节点参数包括：

- 标高；
- 需水量；
- 水头；
- 压强；
- 水质。

管段参数包括：

- 长度；
- 直径；
- 流量；
- 流速；
- 水头损失；
- 位置（与状态相同-开启、活动、关闭）；
- 设置（对应于管道的粗糙系数、水泵的转速、阀门的压力/流量设置）；
- 反应（反应速率）；
- F-因子（摩擦因子）。

报告的缺省量对于节点的需水量、水头、压强和水质，以及管段的流量、流速和水头损失。缺省精度为两个小数位。

例子:
下例报告是关于节点N1, N2, N3和N17，以及所有流速大于3.0的管段。标准节点变量(需水量, 水头, 压强和水质)被报告，同时管段仅仅报告流量、流速和F因子（摩擦因子）。

```
[REPORT]
NODES N1 N2 N3 N17
LINKS ALL
FLOW YES
VELOCITY PRECISION 4
F-FACTOR PRECISION 4
VELOCITY ABOVE 3.0
```

6.3.2 输出文件格式

工具箱利用非格式化二进制输出文件在均匀报告间隔上，存储水力和水质结果。写入文件的数据有4字节整数、4字节浮点数或者4字节倍数的固定尺寸字符串。这允许文件方便分隔为4字节的记录。文件包含了以下以字节计的四个部分：

部分	尺寸，字节
前言	852 + 20*Nnodes + 36*Nlinks + 8*Ntanks
能量利用	28*Npumps + 4
动态结果	(16*Nnodes + 32*Nlinks)*Nperiods
后序	28

式中

Nnodes = 节点（连接节点 + 水库 + 水池）总数
Nlinks = 管段(管道 + 水泵 + 阀门)总数
Ntanks = 水池和水库总数
Npumps = 水泵总数
Nperiods = 报告时段总数

以及所有这些数值写入到文件的前言或者后序部分。

输出文件 – 前言

二进制输出文件的前言部分包含了以下数据：

数据项	类型	字节总数
幻数(= 516114521)	整型	4
版本(= 200)	整型	4
节点总数 (连接节点 + 水库 + 水池)	整型	4
水库和水池总数	整型	4

管段总数 (管道 + 水泵 + 阀门)	整型	4
水泵总数	整型	4
阀门总数	整型	4
水质选项 0 = 无 1 = 化学成分 2 = 水龄 3 = 源头跟踪	整型	4
源头跟踪的节点索引	整型	4
流量单位选项 0 = cfs 1 = gpm 2 = mgd 3 = 英制mgd 4 = acre-ft/day 5 = 升/秒 6 = 升/分 7 = 百万升/日 8 = 立方米/时 9 = 立方米/日	整型	4
压强单位选项 0 = psi 1 = 米 2 = kPa	整型	4
时间统计标识 0 = 无 (报告时间序列) 1 = 报告时间均值 2 = 报告最小值 3 = 报告最大值 4 = 报告范围	整型	4
报告起始时间 (秒)	整型	4
报告时间步长 (秒)	整型	4
模拟历时 (秒)	整型	4
问题标题 (第1行)	字符型	80
问题标题(第2行)	字符型	80
问题标题 (第3行)	字符型	80
输入文件名	字符型	260
报告文件名	字符型	260
化学成分名称	字符型	16

化学成分浓度单位	字符型	16
每一节点的ID字符串	字符型	16*Nnodes
每一管段的ID字符串	字符型	16*Nlinks
每一管段起始节点的索引	整型	4*Nlinks
每一管段终止节点的索引	I整型	4*Nlinks
每一管段的类型代码 0 = 具有CV的管道 1 = 管道 2 = 水泵 3 = PRV 4 = PSV 5 = PBV 6 = FCV 7 = TCV 8 = GPV	整型	4*Nlinks
每一水池的节点索引	整型	4*Ntanks
每一水池的断面积 (值0表明为水库)	浮点型	4*Ntanks
每一节点的标高	浮点型	4*Nnodes
每一管段的长度	浮点型	4*Nlinks
每一管段的直径	浮点型	4*Nlinks

输出文件 – 能量利用

输出文件的耗能部分包含了以下数据：

数据项	类型	字节数
为每一水泵重复：		
● 管段列表中的水泵索引	整型	4
● 水泵利用率 (%)	浮点数	4
● 平均效率 (%)	浮点数	4
● 平均千瓦/MGal (或者千瓦/cu m)	浮点数	4
● 平均千瓦数	浮点数	4
● 高峰千瓦数	浮点型	4
● 每日的平均成本	浮点型	4
高峰耗能(kw-hrs)	浮点型	4

输出文件 – 动态结果

二进制输出文件的动态结果部分包含了**每一报告时段**的以下数据集(报告时间步长写在输出文件的前言部分，这些步长的数量写在后序部分)：

数据项	类型	字节数
每一节点的需水量	浮点型	4*Nnodes
每一节点的水头 (梯度)	浮点型	4*Nnodes
每一节点的压强	浮点型	4*Nnodes
每一节点的水质	浮点型	4*Nnodes
每一管段的流量 (逆向流为负值)	浮点型	4*Nlinks
每一管段中的流速	浮点型	4*Nlinks
每一管段长度为1000单位时的水头损失(水泵的总水头和阀门的水头损失)	浮点型	4*Nlinks
每一管段中的平均水质	浮点型	4*Nlinks
每一管段的状态代码 ● 0 = 关闭(超过的最大水头) ● 1 = 临时关闭 ● 2 = 关闭 ● 3 = 开启 ● 4 = 活动 (部分开启) ● 5 = 开启 (超过最大流量) ● 6 = 开启(不满足流量设置) ● 7 = 开启(不满足压力设置)	浮点型	4*Nlinks
每一管段的设置 ● 管道的粗糙系数 ● 水泵的转速 ● 阀门的设置	浮点型	4*Nlinks
每一管段的反应速率 (mass/L/day)	浮点型	4*Nlinks
每一管段的摩擦因子	浮点型	4*Nlinks

输出文件 – 后序

二进制输出文件的后序部分包含了以下数据：

数据项	类型	字节数
主流区平均反应速率 (mass/hr)	浮点型	4
管壁处平均反应速率 (mass/hr)	浮点型	4
水池平均反应速率(mass/hr)	浮点型	4
平均源头进流量 (mass/hr)	浮点型	4
报告时段总数	整型	4
警告标志 ● 0 = 无警告 ● 1 = 产生警告	整型	4
幻数 (= 516114521)	整型	4

输入文件示例

以下摘自EPANETH输入文件示例，采用在输入文件格式中描述的格式：

```
[TITLE]
Example EPANET Input File

[JUNCTIONS]
;ID    Elev.
;-----------
J1     100
J2     120
< etc. >

[RESERVOIRS]
;ID    Head
;-----------
R1     55

[TANKS]
;           Init.  Min.   Max.
;ID   Elev.   Level  Level  Level  Diam.
;---------------------------------------
T1    200     12     2     20     120

[PIPES]
;                                    Minor
;ID   Node1  Node2  Length  Diam.  Rough.  Loss   Status
;-----------------------------------------------------
P1    J1     J2     1200    12     100     0      Open
P2    J2     J3     2400    16     100     0      Open
P3    J3     J20     400     8     100     2.4    CV
< etc. >
```

```
[PUMPS]
;ID   Node1  Node2  Characteristics
;----------------------------------
PMP1  R1     J1     HEAD CURVE1

[DEMANDS]
;         Base   Demand
;Junction  Demand Pattern  Category
;-----------------------------------
J1        50     PAT1    ;Domestic
J1        100    PAT2    ;Hospital
J2        55     PAT1    ;Domestic
< etc. >

[PATTERNS]
;ID   Multipliers
;--------------------------------------
PAT1  1.1  1.2  0.95  0.87  0.65  0.77
PAT1  0.83  1.0  1.1  1.4  1.2  1.1
PAT2  1.0
< etc. >

[CURVES]
;ID    X-value   Y-value
;-----------------------
CURVE1   0       120
CURVE1   150      60
CURVE1   500       0

[REACTIONS]
GLOBAL BULK -0.5

[SOURCES]
;Node  Type  Strength
;--------------------
R1    CONCEN  1.0

[TIMES]
DURATION  24 HRS
PATTERN TIMESTEP 2 HRS

[OPTIONS]
QUALITY  Chlorine mg/L

[END]
```

计量单位

注意：当流量单位表达为升或者立方米时，应用公制单位。当CFS, GPM, MGD, IMGD或者AFD作为流量单位时，应用美制单位。怎样选择流量单位，参见[OPTIONS]。

参数	公制	美制单位
浓度	mg/L或μg/L	mg/L或μg/L
需水量	(参见流量单位)	(参见流量单位)
(管道) 直径	毫米	英寸
(水池) 直径	米	英尺
效率	百分比	百分比
标高	米	英尺
扩散器系数	流量单位 @ 1米降落	流量单位 @ 1 psi 降落
能量	千瓦时	千瓦时
流量	LPS (升/秒) LPM (升/分) MLD (百万升/日) CMH (立方米/时) CMD (立方米/日)	CFS (立方英尺/秒) GPM (加仑/分) MGD (百万加仑/日) IMGD (英制MGD) AFD (英亩-英尺/日)
摩擦因子	无量纲	无量纲
水头	米	英尺
长度	米	英尺
局部损失系数	无量纲	无量纲
功率	千瓦	马力
压强	米	psi
反应系数(主流区)	1/日(1级)	1/日 (1级)
反应系数(管壁处)	mass/sq-m/day (0级) 米/日(1级)	mass/sq-ft/day (0级) ft/日(1级)
粗糙系数	mm (Darcy-Weisbach) 否则无量纲	毫英尺 (Darcy-Weisbach) 否则无量纲
源头质量注射	质量/分	质量/分
速度	米/秒	ft/sec
容积	立方米	立方英尺
水龄	小时	小时

ENepanet

声明:

int ENepanet(char* f1, char* f2, char* f3, void (*) (vfunc))

描述:

执行完整的EPANETH模拟。

参数:

f1: 输入文件名
f2: 输出报告文件名
f3: 可选二进制输出文件名
vfunc: 将字符串作为参数的用户使用函数的指针。

返回值:

返回错误代码。

注意:

ENepanet是单独使用的函数，不能够与工具箱中的任何其它函数相互作用。

如果不需要保存EPANETH的二进制输出文件，那么*f3*可以为空字符串("")。

*vfunc*函数指针允许调用程序显示由EPANETH计算过程中产生的过程信息。控制台应用程序的典型函数如下：

```
void  writecon(char *s)
{
   puts(s);
}
```

有时在调用程序中，出现以下声明：

```
void (* vfunc) (char *);
vfunc = writecon;
ENepanet(f1,f2,f3,vfunc);
```

如果不希望这样的函数，该参数应为*NULL* (Delphi/Pascal为*NIL*；Visual Basic为*VBNULLSTRING*)。

ENepanet主要用于将EPANETH引擎连接到建立管网输入文件和显示管网分析结果的第三方用户接口。

ENopen

声明:

int ENopen(char* f1, char* f2, char* f3)

描述:

打开工具箱，为了分析特定配水系统。

参数:

f1: EPANETH输入文件名
f2: 输出报告文件名
f3: 可选二进制输出文件名。

返回:
返回错误代码。

备注:
如果不需要保存EPANETH的二进制输出文件，*f3*可以为空字符串("")。

在使用任何其它工具箱函数以前，必须调用ENopen (除了ENepanet)。

参见:
Enclose

ENclose

声明:
```
int   ENclose( void )
```

描述:
关闭工具箱系统(包括所有正在处理的文件)。

返回:
返回错误代码。

备注:
当完成所有处理之后，必须调用Enclose，即使遇到错误状况。

参见:
ENopen

ENgetnodeindex

声明:
```
int   ENgetnodeindex( char* id, int* index )
```

描述:
检索具有指定ID的节点索引。

参数:
id: 节点ID标签
index: 节点索引

返回:
返回错误代码。

备注:
节点索引是从1开始的连续整数。

参见:
ENgetnodeid

ENgetnodeid

声明:

```
int   ENgetnodeid( int index, char* id )
```

描述:

检索具有指定索引的ID标签。

参数:

index: 节点索引
id: 节点的ID标签

返回:

返回错误代码。

备注:

ID标签字符串的尺寸应至少为15个字符。

节点索引是从1开始的连续整数。

参见:

ENgetnodeindex

ENgetnodetype

声明:

```
int   ENgetnodetype( int index, int* typecode )
```

描述:

检索指定节点的节点类型编号。

参数:

index: 节点索引
typecode: 节点类型编号(见下)

返回:

返回错误代码。

备注:

节点索引为从1开始的连续整数。

节点类型编号包含以下常量:

EN_JUNCTION	0	连接节点
EN_RESERVOIR	1	水库节点
EN_TANK	2	水池节点

ENgetnodevalue

声明:

int ENgetnodevalue(int index, int paramcode, float* value)

描述:

检索特定管段参数值。

参数:

index: 节点索引
paramcode: 参数代号(见下)
value: 参数值

返回:

返回错误代号。

备注:

节点索引是从1开始的连续整数。

节点参数代号包含以下常量:

EN_ELEVATION	0	标高
EN_BASEDEMAND	1	基本需水量
EN_PATTERN	2	需水量模式索引
EN_EMITTER	3	扩散器系数
EN_INITQUAL	4	初始水质
EN_SOURCEQUAL	5	源头水质
EN_SOURCEPAT	6	源头模式索引
EN_SOURCETYPE	7	源头类型 (见以下注释)
EN_TANKLEVEL	8	水池中初始水位
EN_DEMAND	9	实际需水量
EN_HEAD	10	水力水头
EN_PRESSURE	11	压强
EN_QUALITY	12	实际水质
EN_SOURCEMASS	13	每分钟化学成分源头的质量流量

参数9 - 13 (EN_DEMAND到EN_SOURCEMASS)是计算数值。其他是输入设计参数。

源头类型利用以下常量识别:

EN_CONCEN	0
EN_MASS	1
EN_SETPOINT	2
EN_FLOWPACED	3

参见这些源头类型描述的[SOURCES]。

返回数值，单位取决于EPANETH输入文件中流量使用的单位(参见计量单位)。

ENgetlinkindex

声明:

```
int   ENgetlinkindex( char* id, int* index )
```

描述:

检索具有特定ID的管段索引。

参数:

id: 管段ID标签
index: 管段索引

返回:

返回错误代码。

备注:

管段索引是从1开始的连续整数。

参见:

ENgetlinkid

ENgetlinkid

声明:

```
int   ENgetlinkid( int index, char* id )
```

描述:

检索具有特定指标的管段的ID标签。

参数:

index: 管段索引
id: 管段的ID标签

返回:

返回错误代码。

备注:

ID标签字符串应保证至少15个字符尺寸。

管段索引是从1开始的连续整数。

参见:

ENgetlinkindex

ENgetlinktype

声明:

int ENgetlinktype(int index, int* typecode)

描述:

检索特定管段的类型编号。

参数:

index: 管段索引
typecode: 管段类型代号(见下)

返回:

返回错误代码。

备注:

管段索引是从1开始的连续整数。

管段类型代号包含了以下常量:

EN_CVPIPE	0	具有止回阀的管道
EN_PIPE	1	管道
EN_PUMP	2	水泵
EN_PRV	3	减压阀
EN_PSV	4	稳压阀
EN_PBV	5	压力制动阀
EN_FCV	6	流量控制阀
EN_TCV	7	节流控制阀
EN_GPV	8	常规阀门

参见:

ENgetlinkindex

ENgetlinknodes

声明:

int ENgetlinknodes(int index, int* fromnode, int* tonode)

描述:

检索指定管段端点节点的索引。

参数:

index: 管段索引
fromnode: 管段起始节点索引
tonode: 管段终止节点索引

返回:

返回错误代号。

备注:

节点和管段索引是从1开始的连续整数。

管段起始和终止节点定义在EPANETH输入文件中。没有考虑管段的实际流向。

参见:

ENgetlinkindex

ENgetlinkvalue

声明:

```
int   ENgetlinkvalue( int index, int paramcode, float* value )
```

描述:

检索指定管段参数值。

参数:

index:	管段索引
paramcode:	参数代号(见下)
value:	参数值

返回:

返回错误代码。

备注:

管段索引是从1开始的连续整数。

管段参数代码包含以下常量:

EN_DIAMETER	0	直径
EN_LENGTH	1	长度
EN_ROUGHNESS	2	粗糙系数
EN_MINORLOSS	3	局部损失系数
EN_INITSTATUS	4	初始管段状态 (0 = 关闭, 1 = 开启)
EN_INITSETTING	5	初始管道粗糙度 初始水泵转速 初始阀门设置
EN_KBULK	6	主流反应系数
EN_KWALL	7	管壁反应系数
EN_FLOW	8	流量
EN_VELOCITY	9	流速
EN_HEADLOSS	10	水头损失
EN_STATUS	11	实际管段状态 (0 = 关闭, 1 = 开启)
EN_SETTING	12	管道粗糙系数 实际水泵转速 实际阀门设置
EN_ENERGY	13	消耗能量，以千瓦计

参数8 - 13 (EN_FLOW到EN_ENERGY)为计算值。其他为设计参数。

如果从指定管段起始节点流向指定终止节点，流量为正；否则为负。

返回的数值单位，取决于EPANETH输入文件中流量使用的单位(见计量单位)。

参见:
ENgetlinkindex

ENgetpatternid

声明:
```
int   ENgetpatternid( int index, char* id )
```

描述:
检索特定时间模式的ID标签。

参数:
index: 模式索引
id: 模式的ID标签

返回:
返回错误代码。

备注:
ID标签字符串的尺寸应至少保持15个字符。

模式索引是从1开始的连续整数。

ENgetpatternindex

声明:
```
int   ENgetpatternindex( char* id, int* index )
```

描述:
检索特定时间模式的索引。

参数:
id: 模式ID标签
index: 模式索引

返回:
返回错误代码。

备注:
模式索引是从1开始的连续整数。

ENgetpatternlen

声明:

```
int   ENgetpatternlen( int index, int* len )
```

描述:

检索特定时间模式中的时段总数。

参数:

index: 模式索引
len: 模式中时段总数

返回:

返回错误代码。

备注:

模式索引是从1开始的连续整数。

ENgetpatternvalue

声明:

```
int   ENgetpatternvalue( int index, int period, float* value )
```

描述:

检索时间模式中特定时段的乘子。

参数:

index: 时间模式索引
period: 时间模式范围内的时段
value: 时段的乘子

返回:

返回错误代码。

备注:

模式索引和时段是从1开始的连续整数。

参见:

ENgetpatternindex, ENgetpatternlen, ENsetpatternvalue

ENgetcontrol

声明:

```
int   ENgetcontrol( int cindex, int* ctype, int* lindex, float* setting, int*
            nindex, float* level )
```

描述:
检索简单控制语句的参数。控制的索引在cindex中指定，以及剩余的参数返回控制参数。

参数:
cindex: 控制语句索引
ctype: 控制类型代码
lindex: 控制管段的索引
setting: 控制设置数值
nindex: 控制节点的索引
level: 对于基于时间控制的控制行为（以秒计）的控制水位或者水位控制的压力或者控制时间的数值

返回:
返回错误代码。

备注:
控制是从1开始的索引，次序为输入到EPENETH输入文件的[CONTROLS]节中。

控制类型代码包含以下：
0 (低水位控制) 当水池水位或者节点压力降落到指定水位以下时使用
1 (高水位控制) 当水池水位或者节点压力上升高于指定水位时使用
2 (计时器控制) 在指定时间用于模拟
3 (钟表时间控制) 使用指定日历时间

对于管道，*设置*为0，意味着管道被关闭；为1意味着它是开启的。对于水泵，*设置*包含了水泵转速，为0，意味着水泵被关闭，为1意味着处于常规转速。对于阀门，*设置*是指阀门的压力、流量或者损失系数值，取决于阀门类型

对于计时器或者钟表时间控制，*nindex*参数等于0。

对于使用该函数的例子，参见ENsetcontrol。

ENgetcount

声明:
```
int   ENgetcount( int countcode, int *count )
```

描述:
检索指定类型的管网组件数量。

参数:
countcode: 组件编号（见下）
count: 管网中*countcode*组件的数量

返回:
返回错误代码。

备注:

组件编号包含以下：

EN_NODECOUNT	0	节点
EN_TANKCOUNT	1	水库和水池节点
EN_LINKCOUNT	2	管段
EN_PATCOUNT	3	时间模式
EN_CURVECOUNT	4	曲线
EN_CONTROLCOUNT	5	简单控制

管网中连接节点的数量等于节点总数减去水池和水库总数。

对于在输入文件描述的组件，工具箱内没有设施添加或者删除功能。

ENgetflowunits

声明:

```
int   ENgetflowunits( int* unitscode )
```

描述:

检索表明使用单位的代码数，为了表达所有流量。

参数:

unitscode: 流量范围代码号的数值(见下)。

返回:

返回错误代码。

备注:

流量范围代码如下：

0	= EN_CFS	立方英尺每秒
1	= EN_GPM	加仑每分
2	= EN_MGD	百万加仑每日
3	= EN_IMGD	英制mgd
4	= EN_AFD	英亩—英尺每日
5	= EN_LPS	升每秒
6	= EN_LPM	升每分
7	= EN_MLD	百万升每日
8	= EN_CMH	立方米每时
9	= EN_CMD	立方米每日

在EPANETH输入文件的[OPTIONS]节指定流量单位。

流量单位取升或者立方米，说明所有其它量将采用公制单位；否则使用美制单位。(见计量单位)。

ENgettimeparam

声明:

int ENgettimeparam(int paramcode, long* timevalue)

描述:

检索指定分析时间参数的数值。

参数:

paramcode: 时间参数编号（见下）
timevalue: 时间参数值，以秒计

返回:

返回错误代码。

备注:

时间参数代码包括以下常量：

EN_DURATION	0	模拟历时
EN_HYDSTEP	1	水力时间步长
EN_QUALSTEP	2	水质时间步长
EN_PATTERNSTEP	3	时间模式时间步长
EN_PATTERNSTART	4	时间模式起始时间
EN_REPORTSTEP	5	报告时间步长
EN_REPORTSTART	6	报告起始时间
EN_RULESTEP	7	估计基于规则控制的时间步长
EN_STATISTIC	8	使用时间序列后处理的类型： 0 = 无 1 = 均值 2 = 最小值 3 = 最大值 4 = 范围
EN_PERIODS	9	保存到二进制输出文件的报告时段数

ENgetqualtype

声明:

int ENgetqualtype(int* qualcode, int* tracenode)

描述:

检索调用水质分析的类型。

参数:

qualcode: 水质分析代码(见下)
tracenode: 源头跟踪分子中跟踪的节点索引

返回:

返回错误代码。

备注:

水质分析代码如下：

EN_NONE	0	不进行水质分析
EN_CHEM	1	化学成分分析
EN_AGE	2	水龄分析
EN_TRACE	3	源头跟踪

当*qualcode*不为EN_TRACE时，*tracenode*的数值将为0。

参见:

ENsetqualtype

ENgetoption

声明:

```
int   ENgetoption( int optioncode, float* value )
```

描述:

检索特定分析选项的数值。

参数:

optioncode: 选项编号（见下）
value: 选项值

返回:

返回错误代码。

备注:

选项标号包含以下常量：

EN_TRIALS	0
EN_ACCURACY	1
EN_TOLERANCE	2
EN_EMITEXPON	3
EN_DEMANDMULT	4

ENsetcontrol

声明:

```
int   ENsetcontrol( int cindex, int ctype, int lindex, float setting, int
                    nindex, float level )
```

描述:

设置特定简单控制语句的参数。

参数:

cindex: 控制语句索引
ctype: 控制类型代码

lindex: 被控制管段的索引
setting: 控制设置的数值
nindex: 控制节点的索引
level: 对于水位控制的控制水位或者压力的数值，或者控制行为的时间（以秒计），对于基于时间的控制

返回:
返回错误代码。

备注:
控制索引是从1开始的，顺序中它们被输入，在EPANETH的输入文件的 [CONTROLS]节。

控制类型代码包含以下：

EN_LOWLEVEL	0 当水池水位或者节点压力低于指定数值时应用
EN_HILEVEL	1 当水池水位或者节点压力低于指定值时应用控制
EN_TIMER	2 进入模拟指定的时间应用控制
EN_TIMEOFDAY	3 指定钟表时间应用控制

对于管道, *setting*为0，意味着管道关闭，为1意味着它是开启的。对于水泵, *setting*包含了水泵转速，为0意味着水泵关闭，为1意味着在常规转速下开启。对于阀门, *setting*指阀门压力、流量或者损失系数，取决于阀门类型。

对于计时器或者钟表时间控制，设置*nindex*参数为0。

对于水位控制，如果控制节点*nindex*为水池，那么*level*参数应为高于池底的水位（而不是标高）。否则*level*应为连接节点压力。

为除去关于特定管段的控制，设置*lindex*参数为0。函数中其它参数的数值将被忽略。

例子:

本例将ENgetcontrol and ENsetcontrol用于改变节点的低水位设置，控制管段，具有索引 *thelink* 到新的数值*newlevel*。

```
ENgetcount(EN_CONTROLS, &numctrls);
for (i=1; i<=numctrls; i++)
{
  ENgetcontrol(i, &ctype, &lindex, &setting,
    &nindex, &level);
  if (ctype == EN_LOWLEVEL && lindex == thelink)
  {
    ENsetcontrol(i, ctype, lindex, setting,
      nindex, newlevel);
    break;
  }
}
```

参见:
ENgetcontrol

ENsetnodevalue

声明:

int ENsetnodevalue(int index, int paramcode, float value)

描述:

设置特定节点的参数值。

参数:

index: 节点索引
paramcode: 参数代码(见下)
value: 参数值

返回:

返回错误代码。

备注:

节点索引为从1开始的连续整数。

节点参数代码包含以下常量:

EN_ELEVATION	0	标高
EN_BASEDEMAND	1	基线需水量
EN_PATTERN	2	时间模式索引
EN_EMITTER	3	扩散器系数
EN_INITQUAL	4	初始水质
EN_SOURCEQUAL	5	源头水质
EN_SOURCEPAT	6	源头模式
EN_SOURCETYPE	7	源头类型:(见以下注释)
EN_TANKLEVEL	8	水池的初始水位

源头类型利用以下常量标识:

EN_CONCEN	0
EN_MASS	1
EN_SETPOINT	2
EN_FLOWPACED	3

这些源头类型的描述参见[SOURCES]。

提供数值的单位，取决于EPANETH输入文件中使用的流量单位(参见计量单位)。

ENsetlinkvalue

声明:

int ENsetlinkvalue(int index, int paramcode, float value)

描述:

设置指定管段的参数值。

参数:

index:	管段索引
paramcode:	参数代码(见下)
value:	参数值

返回:

返回错误代码。

备注:

管段索引是从1开始的连续整数。

管段参数代码包含了以下常量:

EN_DIAMETER	0	直径
EN_LENGTH	1	长度
EN_ROUGHNESS	2	粗糙系数
EN_MINORLOSS	3	局部损失系数
EN_INITSTATUS	4	初始管段状态 (0 = 关闭, 1 = 开启)
EN_INITSETTING	5	管道粗糙度 初始水泵转速 初始阀门设置
EN_KBULK	6	主流反应系数
EN_KWALL	7	管壁反应系数
EN_STATUS	11	当前水泵或者阀门状态 (0 = 关闭, 1 = 开启)
EN_SETTING	12	当前水泵转速或者阀门设置

数值单位取决于在EPANETH输入文件中使用的流量范围(参见计量单位)。

对于存在的管段状态或者设置，在模拟开始之前，利用EN_INITSTATUS和EN_INITSETTING设置设计数值。在模拟执行的同时，利用EN_STATUS和EN_SETTING改变这些数值(在ENrunH - ENnextH循环范围内)。

如果控制阀具有明确的状态，设置为OPEN或者CLOSED，需要再次激活，在模拟过程中必须采用EN_SETTING参数提供虚拟的阀门设置数值。

对于管道，EN_ROUGHNESS或者EN_INITSETTING用于改变粗糙系数。

ENsetpattern

声明:

```
int   ENsetpattern( int index, float* factors, int nfactors )
```

描述:

设置指定时间模式的所有乘子。

参数:

index: 时间模式索引
factors: 整个模式的乘子
nfactors: 模式中因子总数

返回:

返回错误代码。

备注:

模式索引是从1开始的连续整数。

*Factors*点到基于零的数组，包含了*nfactors*的元素。

使用该函数重新定义（和重新设置尺寸）时间模式；利用ENsetpatternvalue，修改模式指定时段的模式因子。

参见:

ENgetpatternindex, ENgetpatternlen, Ngetpatternvalue, ENsetpatternvalue

ENsetpatternvalue

声明:

```
int   ENsetpatternvalue( int index, int period, float value )
```

描述:

设置时间模式内特定时段的乘子因子。

参数:

index: 时间模式索引
period: 时间模式内的时段
value: 该时段的乘子因子

返回:

返回错误代码。

备注:

模式索引是从1开始的连续整数。

利用ENsetpattern重新设置时间模式内的所有因子。

参见:

ENgetpatternindex, ENgetpatternlen, ENgetpatternvalue, ENsetpattern

ENsetqualtype

声明:

```
int   ENsetqualtype( int qualcode, char* chemname, char* chemunits,
                     char* tracenode )
```

描述:

设置被调用水质分析的类型。

参数:

qualcode: 水质分析代码(见下)
chemname: 被分析化学成分名称
chemunits: 化学成分计量单位
tracenode: 源头跟踪分析中被跟踪节点的ID

返回:

返回错误代码。

备注:

水质分析代码如下：

EN_NONE	0	不进行水质分析
EN_CHEM	1	化学成分分析
EN_AGE	2	水龄分析
EN_TRACE	3	源头跟踪

如果不进行化学成分分析，化学成分名称和单位可以为空字符串。如果不进行源头跟踪分析，跟踪节点保留名称，。

注意跟踪节点通过ID指定而不是索引。

参见:

ENgetqualtype

ENsettimeparam

声明:

```
int   ENsettimeparam( int paramcode, long timevalue )
```

描述:

设置时间参数值。

参数:

paramcode: 时间参数代码(见下)
timevalue: 时间参数值，以秒计

返回:

返回错误代码。

备注:

时间参数代码包含以下常量:

EN_DURATION	0	模拟历时
EN_HYDSTEP	1	水力时间步长
EN_QUALSTEP	2	水质时间步长
EN_PATTERNSTEP	3	时间模式时间步长
EN_PATTERNSTART	4	时间模式起始时刻
EN_REPORTSTEP	5	报告时间步长
EN_REPORTSTART	6	报告起始时刻
EN_RULESTEP	7	评价基于规则控制的时间步长
EN_STATISTIC	8	用于时间序列后处理的类型: EN_NONE (0) = 无 EN_AVERAGE (1) = 均值 EN_MINIMUM (2) = 最小值 EN_MAXIMUM (3) = 最大值 EN_RANGE (4) = 范围

在水力分析中调用ENinitH或者在水质分析中调用ENinitQ之后不能改变时间参数。

ENsetoption

声明:

```
int   ENsetoption( int optioncode, float value )
```

描述:

设置特定分析选项的数值。

参数:

optioncode: 选项编号 (见下)
value: 选项值

返回:

返回错误代码。

备注:

选项编号包含以下常量:

EN_TRIALS	0
EN_ACCURACY	1
EN_TOLERANCE	2
EN_EMITEXPON	3
EN_DEMANDMULT	4

ENsavehydfile

声明:

int ENsavehydfile(char* fname)

描述:

将当前二进制水力文件的内容保存到一个文件。

参数:

fname: 水力结果应被保存的文件名。

返回:

返回错误代码。

备注:

将该函数用于将当前水力结果集保存到一个文件，便于后处理或者以后时间使用，通过调用ENusehydfile函数。

水力文件包含了节点需水量和水头，以及管段流量，状态和设置，对于所有水力时间步长，甚至中间数值。

调用该函数之前，水力结果必须产生和保存，通过调用ENsolveH或者ENinitH - ENrunH - ENnextH序列，具有ENinitH的保存标志参数设置为1。

参见:

ENusehydfile, ENsolveH, ENinitH

ENusehydfile

声明:

int ENusehydfile(char* fname)

描述:

将指定文件的内容作为当前二进制水力文件。

参数:

fname: 包含了当前管网水力分析结果的文件名。

返回:

返回错误代码。

备注:

调用该函数，拒绝原先保存的水力分析结果集。这些结果被检查，为了查看是否它们匹配一下参数，相关于当前被分析的管网：节点总数、水池和水库总数、管段总数、水泵总数、阀门总数和模拟历时。

当水力分析系统仍旧开启时，不能够调用该函数(即ENopenH已经被调用，但是ENcloseH还没有)。

参见:
ENsavehydfile

ENsolveH

声明:
int ENsolveH(void)

描述:
执行一次完整的水力模拟，所有时段的结果写入二进制水力文件。

返回:
返回错误代码。

备注:
利用ENsolveH产生独立完整的水力结果，或者作为水质分析的输入。为了将水力结果写入报告文件，其后也调入到ENsaveH和ENreport。不要使用ENopenH，ENinitH，ENrunH，ENnextH和ENcloseH连接到ENsolveH。

示例:

```
ENopen("net1.inp", "net1.rpt", "");
ENsolveH();
ENsolveQ();
ENreport();
ENclose();
```

ENopenH

声明:
int ENopenH(void)

描述:
打开水力分析系统。

返回:
返回错误代码。

备注:
在第一次执行水力分析之前调用ENopenH ，利用ENinitH - ENrunH - ENnextH系列。在调用ENcloseH，以关闭水力分析系统之前，可以进行多次分析。

如果ENsolveH用于运行完整的水力分析，不要调用该函数。

参见:
ENinitH, ENrunH, ENnextH, ENcloseH

ENinitH

声明:

int ENinitH(int saveflag)

描述:

在执行水力分析之前，初始化蓄水池水位、管段状态和设置，以及模拟钟表时间。

参数:

saveflag: 0-1标志，说明水力结果是否保存到水力文件。

返回:

返回错误代码。

备注:

利用ENrunH和ENnextH执行水力分析之前，应调用ENinitH。
调用ENinitH之前，必须已经调用ENopenH。
如果整个水力分析调用了ENsolveH，将不再调用ENinitH。
如果将进行随后的水质运行，*saveflag*设置为1，利用ENreport产生报告，或者利用ENsavehydfile保存二进制水力文件。

参见:

ENopenH, ENrunH, ENnextH, ENcloseH

ENrunH

声明:

int ENrunH(long* t)

描述:

执行简单时段水力分析，检索当前模拟钟表时间*t*。

参数:

t: 当前模拟钟表时间，以秒计。

返回:

返回错误代码。

备注:

利用ENrunH结合ENnextH在do ..while循环中，为了分析延时模拟中每一时段的水力特性。该过程自动更新模拟钟表时间，因此将t处理为只读变量。

在执行ENrunH – EnnextH循环以前，必须调用ENinitH。

对于该函数使用的例子，参见ENnextH。

参见:

ENopenH, ENinitH, ENnextHo, ENcloseH

ENnextH

声明:

```
int  ENnextH( long* tstep )
```

描述:

确定延时模拟下一次水力时间发生前的时间长度。

参数:

tstep: 时间(以秒计)，直到下一次水力时间发生；或者为0，如果在模拟时段的末端。

返回:

返回错误代码。

备注:

该函数与ENrunH一起使用，执行延时水力分析（见以下例子）。

*tstep*的数值，应处理为只读变量。作为以下较小值自动计算：

- 时间间隔，直到下一水力时间步长开始
- 时间间隔，直到下一报告时间步长开始
- 时间间隔，直到下一需水量发生改变开始
- 时间间隔，直到水池注满或者排空
- 时间间隔，直到规则控制停止

例子:

```
long t, tstep;
ENopenH();
ENinitH(0);
do
{
   ENrunH(&t);
   /*
      检索时刻t的水力结果
   */
   ENnextH(&tstep);
} while (tstep > 0);
ENcloseH();
```

参见:

ENopenH, ENinitH, ENrunH, ENcloseH, ENsettimeparam

ENcloseH

声明:
int ENcloseH(void)

描述:
关闭水力分析系统，释放所有分配内存。

返回:
返回错误代码。

备注:
在利用ENinitH - ENrunH - ENnextH执行所有水力分析之后，调用ENcloseH。如果使用ENsolveH，不要调用该函数。

参见:
ENopenH, ENinitH, ENrunH, ENnextH

ENsolveQ

声明:
int ENsolveQ(void)

描述:
执行完整的水质模拟，结果具有均匀报告间隔，写入EPANETH二进制输出文件。

返回:
返回错误代码。

备注:
在调用ENsolveQ之前，必须已经执行水力分析，并保存到二进制水力文件。为了将水力和水质结果写入报告文件，后面跟着调用ENreport。不能够将ENopenQ, ENinitQ, ENrunQ, ENnextQ和ENcloseQ，与ENsolveQ一起使用。

示例:
```
ENopen("net1.inp", "net1.rpt", "");
ENsolveH();
ENsolveQ();
ENreport();
ENclose();
```

ENopenQ

声明:
int ENopenQ(void)

描述:
打开水质分析系统。

返回:
返回错误代码。

备注:
调用ENopenQ之前，首先执行水质分析，依次利用ENinitQ - ENrunQ - ENnextQ (or ENstepQ)。为了关闭水质分析系统调用ENcloseQ之前，可以进行多重水质分析。

如果利用ENsolveQ进行完整水质分析，不要调用该函数。

参见:
ENintQ, ENrunQ, ENnextQ, ENstepQ, ENcloseQ

ENinitQ

声明:
```
int   ENinitQ( int saveflag )
```

描述:
在执行水质分析之前，初始化水质和模拟钟表时间。

参数:
saveflag: 0-1标志，说明分析结果是否以均匀报告时段保存到EPANETH二进制输出文件。

返回:
返回错误代码。

备注:
在利用ENrunQ联合ENnextQ或者ENstepQ执行水质分析之前，调用ENinitQ。

必须在调用ENinitQ之前调用ENopenQ。

如果调用ENsolveQ执行完整水质分析，不需调用ENinitQ。

如果希望利用ENreport产生报告，或者希望将计算结果保存为二进制输出文件，设置*saveflag*为1。

参见:
ENopenQ, ENrunQ, ENnextQ, ENstepQ, ENcloseQ

ENrunQ

声明:
```
int   ENrunQ( long * t )
```

描述:
在下一时段水质分析的开始，使水力和水质分析结果可用，其中时段的开始返回为*t*。

参数:

t: 当前模拟钟表时间，以秒计。

返回:

返回错误代码。

备注:

在do...while循环中利用ENrunQ和ENnextQ，为了访问延时模拟每一水力时段开始时的水质结果。或者在do...while循环中将它与ENstepQ一起使用，为了访问每一水质时间步长开始时的结果。怎样编制这种循环的例子，参见每一函数说明。

在执行ENrunQ - ENnextQ (或者ENstepQ)循环之前，必须调用ENinitQ。

模拟的当前时刻*t*由保存在水力分析中的信息确定，然后进行水质分析。它为只读变量。

参见:

ENopenQ, ENinitQ, ENnextQ, ENstepQ, ENcloseQ

ENnextQ

声明:

```
int   ENnextQ( long *tstep )
```

描述:

水质模拟前进到下一水力时段的开始。

参数:

tstep: 时刻(以秒计)，直到下一水力事件发生；或者为零，如果在模拟时段的末端。

返回:

返回错误代码。

备注:

该函数用在do-while循环中，利用ENrunQ执行延时水质分析。允许在模拟的每一水力时段访问水质结果。水质演算和反应以更小的时间步长，在内部执行。利用ENstepQ代替该函数，如果需要访问每一水质事件步长后的结果。

*tstep*的数值根据保存在水力分析中的信息确定，之前为水质分析。将它处理为只读变量。

例子:

```
long t, tstep;
ENsolveH(); /* 产生和保存水力特性 */
ENopenQ();
ENinitQ(0);
do
{
   ENrunQ(&t);
   /*
```

```
      监视时刻t的结果，
      开始新的水力时段
   */
   ENnextQ(&tstep);
} while (tstep > 0)
ENcloseQ();
```

参见:

ENopenQ, ENinitQ, ENrunQ, ENcloseQ

ENstepQ

声明:

```
int  ENstepQ( long* tleft )
```

描述:

水质模拟前进一个水质时间步长。保留整个模拟中的时间范围在*tleft*中。

参数:

tleft: 保留在整个模拟历时中的秒。

返回:

返回错误代码。

备注:

该函数用在ENrunQ的do-while循环中，执行延时水质模拟。允许访问模拟每一水质时间步长的水质结果，而不是像ENnextQ的每一水力时段的开始。

使用参数*tleft*，确定何时不再需要调用ENrunQ，因为达到模拟时段的末尾(即当*tleft* = 0时)。

将*tleft*处理为只读变量(不需要赋值)。

例子:

```
long t, tleft;
ENsolveH(); /* 产生和保存水力特性 */
ENopenQ();
ENinitQ(0);
do
{
   ENrunQ(&t);
   /*
      监视时刻t的结果
   */
   ENstepQ(&tleft);
} while (tleft > 0)
ENcloseQ();
```

参见:

ENopenQ, ENinitQ, ENrunQ, ENcloseQ

ENcloseQ

声明:

int ENcloseQ(void)

描述:

关闭水质分析系统，释放所有分配的内存。

返回:

返回错误代码。

备注:

在利用ENinitQ - ENrunQ - ENnextQ (或者ENstepQ)系列函数调用执行所有水质分析之后，调用ENcloseQ。如果使用了ENsolveQ，将不需要调用该函数。

参见:

ENopenQ, ENinitQ, ENrunQ, ENstepQ, ENnextQ

ENsaveH

声明:

int ENsaveH(void)

描述:

将水力模拟结果从二进制水力文件转换到二进制输出文件，其中结果仅仅以均匀时间间隔进行报告。

返回:

返回错误代码。

备注:

ENsaveH仅在水力分析后执行，并在均匀报告间隔的结果需要转换为EPANETH的二进制输出文件时使用。这种情况，利用ENreport将输出报告写入EPANETH报告文件。
在EPANETH输入文件([TIMES]节)或者通过利用ENsettimeparamE函数，可以设置报告时间。

参见:

ENreport, ENsettimeparam

ENsaveinpfile

声明:

int ENsaveinpfile(char* fname)

描述:

将所有当前管网输入数据写入到文件，利用EPANETH输入文件的格式。

参数:

fname: 数据保存的文件名。

返回:

返回错误代码。

备注:

保存的数据反映了任何变化，通过调用到函数的ENsetxxx族，当EPANETH数据首次被调用，使用ENopen。

ENreport

声明:

```
int   ENreport( void )
```

描述:

将模拟结果的格式化文本报告写入到报告文件。

返回:

返回错误代码。

备注:

调用ENreport之前，必须已经进行了完整的水力分析或者完整的水力水质分析，结果保存在文件中。在前者情况中，也必须首先调用ENsaveH，为了将结果从水力文件转换到输出文件。

报告的格式受到EPANETH输入文件[REPORT]节的命令所控制，或者通过与ENsetreport函数一起公布的类似命令。

ENresetreport

声明:

```
int   ENresetreport( void )
```

描述:

清除任何出现在EPANETH输入文件[REPORT]节中的报告格式命令，或者利用ENsetreport函数所公布的。

返回:

返回错误代码。

Notes:

调用该函数之后，缺省报告选项生效。这些为：

- 没有状态报告
- 没有能量报告
- 没有节点报告
- 没有管段报告

- 节点变量报告为2位小数位
- 管段变量报告为两位小数位 (摩擦因子为3位)
- 报告的节点变量为标高、水头、压强和水质
- 报告的管段变量为流量、流速和水头损失

参见:
ENreport, ENsetreport, ENsetstatusreport

ENsetreport

声明:
int ENsetreport(char* command)

描述:
公布报告格式命令。格式命令与EPANETH输入文件的[REPORT]节中使用的相同。

参数:
command: 报告格式命令文本。

返回:
返回错误代码。

备注:
调用ENresetreport，明确原来报告格式命令，出现在输入文件中，或者通过调用ENsetreport或者ENsetstatusreport公布。

模拟的格式结果可以写入到报告文件，利用ENreport函数。

参见:
ENreport, ENresetreport, ENsetstatusreport

ENsetstatusreport

声明:
int ENsetstatusreport(int statuslevel)

描述:
设置水力状态报告的水平。

参数:
statuslevel: 状态报告的水平(见下)。

返回:
返回错误代码。

备注:
当水力模拟逐渐展开时，状态报告将管网元素的水力状态变化写入到报告文件。具有三种水平的报告：

0 – 无状态报告
1 – 常规报告
2 – 完整状态报告

完整状态报告包含了求解系统水力方程组，在模拟的每一时间步长的收敛信息。主要用于调试目的。

如果将在应用程序中执行许多次水力分析，建议状态报告关闭(*statuslevel* = 0)。

ENgeterror

声明:

```
int   ENgeterror( int errcode, char* errmsg, int nchar )
```

描述:

检索与特定错误或者警告代码相关的信息文本。

参数:

errcode: 错误或者警告代码
errmsg: 对应于*errcode*的错误或者警告信息的文本
nchar: *errmsg*可以拥有的最大字符数

返回:

返回错误代码。

备注:

错误信息字符串至少为80个字符长度。

以任务组织的工具箱函数