**一．nVinPython使用方法**

1）首先 import nV，导入nV模块

2）nV模块里有5个函数：eval, execute, system, globals, require

a) execute和eval参数为一个字符串，可以用于直接以nV的方式输入命令，比如a = nV.eval(“100!”), nV.execute(“b = 1”)等等。nV命令输入错误应该有提示。

b) system函数用于引入System名字空间，globals用于引入Global名字空间，它们两者都没有参数，比如ns = nV.system(). ng = nV.globals()。

b)require用于引入其它我现在不知道的其它顶级名字空间，参数是一个字符串，为名字空间的名字,比如ns = nV.require(“System”), ng = nV.require(“Global”)。

取到名字空间以后，就能访问nV的功能，比如刚执行了nV.execute(“b = 1”)后，使用ng.a就能取得Global名字空间里变量a的值1。比如还可以执行ns.GCD(nV.eval(“100!”), nV.eval(“99!)”)，返回他们的最大公约数，ns.D(ns, ns)求导返回1等等。

**二．nVinPython程序说明**

1）主要数据结构：nVObject扩展自PyObject，添加一个新的字段r，类型为nV里的var。nVObject有三个相关的函数，nVObject\_New（新建并初始化一个nVObject），nVObject\_dealloc（销毁一个nVObject）, PynV\_Check判断一个PyObject是否是nVObject

2）函数nVConvert主要用于将一个nV里的var转换到相应的Python数据结构，如果是symbol，先对其求值，返回symbol对应的值，如果求值后，仍是symbol，返回一个nVObject。如果是Key，就返回相应的Python基本数据结构，如果是Object，是大整数，有理数，字符串等，直接转换到相应的Python基本数据结构，其它Object则转换到nVObject。Tuple转换到PyListObject。

3）此外，还有作为Python扩展要求的函数，在nVObject\_Type类型里有说明。