**Glossary 1**

*problem solving:*

解决问题*:*

The process of formulating a problem, finding a solution, and expressing it.

阐述一个问题、找到解决方案并表达出来的过程。

*high-level language:*

高级语言*:*

A programming language like Python that is designed to be easy for humans to read and write.

一种像Python这样的编程语言，被设计为便于人类读和写。

*low-level language:*

低级语言*:*

A programming language that is designed to be easy for a computer to run; also called “machine language” or “assembly language”.

一种为使计算机易于运行而设计的程序设计语言;也称为“机器语言”或“汇编语言”。

*portability:*

可移植性*:*

A property of a program that can run on more than one kind of computer.

能在一种以上计算机上运行的程序的一种属性。

*interpreter:*

翻译*:*

A program that reads another program and executes it.

读取另一个程序并执行它的程序。

*prompt:*

提示*:*

Characters displayed by the interpreter to indicate that it is ready to take input from the user.

解释器显示的字符，表明它已经准备好接受用户的输入。

*program:*

程序*:*

A set of instructions that specifies a computation.

一组指定计算的指令。

*print statement:*

*print*语句*:*

An instruction that causes the Python interpreter to display a value on the screen.

使*Python*解释器在屏幕上显示一个值的指令。

*operator:*

接线员*:*

A special symbol that represents a simple computation like addition, multiplica‐ tion, or string concatenation.

一种特殊的符号，用于表示简单的运算，如加法、乘法或字符串连接。

*value:*

值*:*

One of the basic units of data, like a number or string, that a program manipulates.

一种基本的数据单位，如数字或字符串，由程序操作。

*type:*

类型*:*

A category of values. The types we have seen so far are integers (type int), floating-point numbers (type float), and strings (type str).

价值的一类。到目前为止，我们看到的类型有整数(int类型)、浮点数(float类型)和字符串(str类型)。

*integer:*

整数*:*

A type that represents whole numbers.

一种表示整数的类型

*floating-point:*

浮点数*:*

A type that represents numbers with fractional parts.

一种表示带有小数部分的数字的类型。

*string:*

字符串*:*

A type that represents sequences of characters.

一种表示字符序列的类型。

*natural language:*

自然语言*:*

Any one of the languages that people speak that evolved naturally.

任何一种人类自然进化的语言。

*formal language:*

正式的语言*:*

Any one of the languages that people have designed for specific purposes, such as representing mathematical ideas or computer programs; all programming lan‐ guages are formal languages.

任何一种人们为特定目的而设计的语言，如表示数学思想或计算机程序;所有的编程语言都是形式语言。

*token:*

令牌*:*

One of the basic elements of the syntactic structure of a program, analogous to a word in a natural language.

程序语法结构的基本元素之一，类似于自然语言中的词。

*syntax:*

语法*:*

The rules that govern the structure of a program.

控制程序结构的规则。

*parse:*

解析*:*

To examine a program and analyze the syntactic structure.

检查一个程序并分析其语法结构。

*bug:* 错误*:*

An error in a program

程序中的一个错误。

*debugging:*

调试*:*

The process of finding and correcting bugs.

发现并纠正错误的过程。

**Glossary 2**

*variable:*

变量*:*

A name that refers to a value.

指值的名称。

*assignment:*

任务*:*

A statement that assigns a value to a variable.   
给变量赋值的语句。

*state diagram:*

状态图*:*

A graphical representation of a set of variables and the values they refer to.

一组变量及其所引用的值的图形表示

*keyword:*

关键字*:*

A reserved word that is used to parse a program; you cannot use keywords like if, def, and while as variable names.

用于解析程序的保留字;不能使用if、def和while等关键字作为变量名。

*operand:*

操作数*:*

One of the values on which an operator operates.

运算符操作的值之一。

*expression:*

表达式*:*

A combination of variables, operators, and values that represents a single result.

表示单个结果的变量、操作符和值的组合。

*evaluate:*

评估*:*

To simplify an expression by performing the operations in order to yield a single value.

通过执行操作来简化表达式以产生单个值。

*statement:*

声明*:*

A section of code that represents a command or action. So far, the statements we have seen are assignments and print statements.

表示命令或动作的一段代码。到目前为止，我们看到的语句是赋值语句和打印语句

*execute:*

执行*:*

To run a statement and do what it says.

运行一个语句并执行它所说的操作。

*interactive mode:*

交互模式*:*

A way of using the Python interpreter by typing code at the prompt.

一种通过在提示符处输入代码来使用Python解释器的方法。

*script mode:*

脚本模式*:*

A way of using the Python interpreter to read code from a script and run it.

一种使用Python解释器从脚本中读取代码并运行它的方法。

*script:*

脚本*:*

A program stored in a file.

存储在文件中的程序。

*order of operations:*

顺序操作*:*

Rules governing the order in which expressions involving multiple operators and operands are evaluated.

控制涉及多个操作符和操作数的表达式计算顺序的规则。

*concatenate:*

连接*:*

To join two operands end-to-end.

端到端的连接两个操作数。

*comment:*

备注*:*

Information in a program that is meant for other programmers (or anyone read‐ ing the source code) and has no effect on the execution of the program.

程序中对其他程序员(或任何读取源代码的人)没有影响程序执行的信息。

*syntax error:*

语法错误*:*

An error in a program that makes it impossible to parse (and therefore impossi‐ ble to interpret).

程序中的错误，使其无法解析(因此也无法解释)。

*exception:*

异常*:*

An error that is detected while the program is running.

程序运行时检测到的错误。

*semantics:*

语义*:*

The meaning of a program.

程序的意义。

*semantic error:*

语义错误*:*

An error in a program that makes it do something other than what the program‐ mer intended.

一个程序中的错误，使它做了与程序预期的不同的事情。

**Glossary 3**

*function:*

函数*:*

A named sequence of statements that performs some useful operation. Functions may or may not take arguments and may or may not produce a result.

执行一些有用操作的已命名语句序列。函数可以接受也可以不接受参数，也可以不产生结果。

*function definition:*

函数定义*:*

A statement that creates a new function, specifying its name, parameters, and the statements it contains.

创建新函数的语句，指定其名称、参数和包含的语句。

*function object:*

函数对象*:*

A value created by a function definition. The name of the function is a variable that refers to a function object.

由函数定义创建的值。函数名是一个指向函数对象的变量。

*header:*

标题*:*

The first line of a function definition.

函数定义的第一行。

*body:*

身体*:*

The sequence of statements inside a function definition.

函数定义内的语句序列。

*parameter:*

参数*:*

A name used inside a function to refer to the value passed as an argument.

函数内用于引用作为参数传递的值的名称。

*function call:*

函数调用*:*

A statement that runs a function. It consists of the function name followed by an argument list in parentheses.

运行函数的语句。它由函数名后跟圆括号中的参数列表组成。

*argument:*

参数*:*

A value provided to a function when the function is called. This value is assigned to the corresponding parameter in the function.

函数调用时提供给函数的值。这个值被赋给函数中相应的参数。

*local variable:*

局部变量*:*

A variable defined inside a function. A local variable can only be used inside its function.

函数内定义的变量。局部变量只能在其函数内部使用。

*return value:*

返回值*:*

The result of a function. If a function call is used as an expression, the return value is the value of the expression.

函数的结果。如果函数调用被用作表达式，则返回值就是表达式的值。

*fruitful function:*

有返回值的函数*:*

A function that returns a value.

返回值的函数。

*void function:*

无返回值的函数*:*

A function that always returns None.

返回值是*None*。

None*:*

无：A special value returned by void functions.

void函数返回的特殊值

*module:*

模块*:*

A file that contains a collection of related functions and other definitions.

包含一组相关函数和其他定义的文件。

*import statement:*

导入语句*:*

A statement that reads a module file and creates a module object.

读取模块文件并创建模块对象的语句。

*module object:*

模块对象*:*

A value created by an import statement that provides access to the values defined in a module.

一个由import语句创建的值，该语句提供了对模块中定义的值的访问。

*dot notation:*

点符号*:*

The syntax for calling a function in another module by specifying the module name followed by a dot (period) and the function name.

在另一个模块中调用函数的语法，通过指定模块名后跟点(句点)和函数名。

*composition:*

成分*:*

Using an expression as part of a larger expression, or a statement as part of a larger statement.

将表达式用作较大表达式的一部分，或将语句用作较大语句的一部分。

*flow of execution:*

流的执行*:*

The order statements run in.

订单语句运行。

*stack diagram:*

堆栈图*:*

A graphical representation of a stack of functions, their variables, and the values they refer to.

函数栈、函数的变量和它们引用的值的图形表示。

*frame:*

框架*:*

A box in a stack diagram that represents a function call. It contains the local vari‐ ables and parameters of the function.

堆栈图中表示函数调用的框。它包含了函数的局部变量和参数。

*traceback:*

回溯*:*

A list of the functions that are executing, printed when an exception occurs.

正在执行的函数列表，当发生异常时打印出来。

**Glossary 4**

*method:*

方法*:*

A function that is associated with an object and called using dot notation.

与对象相关联并使用点表示法调用的函数。

*loop:*

循环*:*

A part of a program that can run repeatedly.

程序中可以重复运行的部分。

*encapsulation:*

封装*:*

The process of transforming a sequence of statements into a function definition.

将一系列语句转换成函数定义的过程。

*generalization:*

泛化*:*

The process of replacing something unnecessarily specific (like a number) with something appropriately general (like a variable or parameter).

将不必要的特定内容(如数字)替换为适当的通用内容(如变量或参数)的过程。

*keyword argument:*

关键字参数*:*

An argument that includes the name of the parameter as a “keyword”.

将参数的名称作为“关键字”包含的参数。

*interface:*

接口*:*

A description of how to use a function, including the name and descriptions of the arguments and return value.

关于如何使用函数的描述，包括参数和返回值的名称和描述。

*refactoring:*

重构*:*

The process of modifying a working program to improve function interfaces and other qualities of the code.

修改工作程序以改进函数接口和其他代码质量的过程。

*development plan:*

编码*:*

A process for writing programs.

编写程序的过程。

*docstring:*

文档字符串*:*

A string that appears at the top of a function definition to document the func‐ tion’s interface.

一个出现在函数定义顶部的字符串，用来记录函数的接口。

*precondition:*

先决条件*:*

A requirement that should be satisfied by the caller before a function starts.

在一个函数开始之前，调用者应该满足的需求。

*postcondition:*

后置条件*:*

A requirement that should be satisfied by the function before it ends.

函数在结束之前应该满足的需求。

**Glossary 5**

*floor division:*

向下划分*:*

An operator, denoted //, that divides two numbers and rounds down (toward zero) to an integer.

一个运算符，表示为//，它将两个数整除并舍入(接近0)为整数。

*modulus operator:*

模运算符*:*

An operator, denoted with a percent sign (%), that works on integers and returns the remainder when one number is divided by another.

一种操作符，用百分号(%)表示，用于整数，并在一个数被另一个数除时返回余数。

*boolean expression:*

布尔表达式*:*

An expression whose value is either True or False.

一个值为True或False的表达式。

*relational operator:*

关系操作符*:*

One of the operators that compares its operands: ==, !=, >, <, >=, and <=.

比较其操作数的操作符之一:==，!=，>，<，>=和<=。

*logical operator:*

逻辑运算符*:*

One of the operators that combines boolean expressions: and, or, and not.

组合布尔表达式的运算符之一:and、or和not。

*conditional statement:*

条件语句*:*

A statement that controls the flow of execution depending on some condition.

根据某些条件控制执行流的语句。

*condition:*

条件*:*

The boolean expression in a conditional statement that determines which branch runs.

条件语句中的布尔表达式，确定运行哪个分支。

*compound statement:*

复合语句*:*

A statement that consists of a header and a body. The header ends with a colon (:). The body is indented relative to the header.

由头文件和正文组成的语句。标题以冒号(:)结束。正文相对于头部是缩进的。

*branch:*

分支*:*

One of the alternative sequences of statements in a conditional statement.

条件语句中的可选语句序列之一。

*chained conditional:*

链接条件*:*

A conditional statement with a series of alternative branches.

具有一系列可选分支的条件语句

*nested conditional:*

嵌套的条件*:*

A conditional statement that appears in one of the branches of another condi‐ tional statement.

条件句出现在另一个条件句的分支中的一个条件句

*return statement:*

返回语句*:*

A statement that causes a function to end immediately and return to the caller.

使函数立即结束并返回给调用者的语句。

*recursion:*

递归*:*

The process of calling the function that is currently executing.

调用当前正在执行的函数的过程。

*base case:*

基本情况*:*

A conditional branch in a recursive function that does not make a recursive call.

递归函数中不进行递归调用的条件分支。

*infinite recursion:*

无限递归*:*

A recursion that doesn’t have a base case, or never reaches it. Eventually, an infin‐ ite recursion causes a runtime error.

一个没有基本情况的递归，或者永远达不到它。最终，无限递归导致运行时错误。

**Glossary 6**

*temporary variable:*

临时变量*:*

A variable used to store an intermediate value in a complex calculation.

在复杂计算中用来存储中间值的一种变量。

*dead code:*

死代码*:*

Part of a program that can never run, often because it appears after a return statement.

永远不能运行的程序的一部分，通常是因为它出现在返回语句之后。

*incremental development:*

增量开发*:*

A program development plan intended to avoid debugging by adding and testing only a small amount of code at a time.

一种程序开发计划，旨在通过每次只添加和测试少量代码来避免调试。

*scaffolding:*

支架*:*

Code that is used during program development but is not part of the final version.

在程序开发过程中使用的代码，但不是最终版本的一部分。

*guardian:*

A programming pattern that uses a conditional statement to check for and hadle circumstances that might cause an error.

一种编程模式，使用条件语句来检查可能导致错误的情况。

**Glossary 7**

*reassignment:*

重新分配*:*

Assigning a new value to a variable that already exists.

给已经存在的变量赋值。

*update:*

更新*:*

An assignment where the new value of the variable depends on the old.

变量的新值依赖于旧值的赋值。

*initialization:*

初始化*:*

An assignment that gives an initial value to a variable that will be updated.

将初始值赋给将要更新的变量的赋值。

*increment:*

增量*:*

An update that increases the value of a variable (often by one).

使变量的值增加(通常为1)的更新。

*decrement:*

递减*:*

An update that decreases the value of a variable.

减少变量值的更新。

*iteration:*

迭代*:*

Repeated execution of a set of statements using either a recursive function call or a loop.

使用递归函数调用或循环重复执行一组语句。

*infinite loop:*

无限循环*:*

A loop in which the terminating condition is never satisfied.

永远不满足终止条件的循环。

*algorithm:*

算法*:*

A general process for solving a category of problems.

解决一类问题的一般过程。

**Glossary 8**

*object:*

对象*:*

Something a variable can refer to. For now, you can use “object” and “value” interchangeably.

变量可以引用的东西。现在，您可以交替使用“对象”和“值”。

*sequence:*

序列*:*

An ordered collection of values where each value is identified by an integer index.

一个有序的值集合，其中每个值由一个整数索引标识。

*item:*

元素*:*

One of the values in a sequence.

序列中的一个值。

*index:*

索引*:*

An integer value used to select an item in a sequence, such as a character in a string. In Python indices start from 0.

一种整数值，用于选择序列中的一项，如字符串中的一个字符。在Python中，索引从0开始。

*slice:*

片*:*

A part of a string specified by a range of indices.

由索引范围指定的字符串的一部分。

*empty string:*

空字符串*:*

A string with no characters and length 0, represented by two quotation marks.

一个不包含字符且长度为0的字符串，由两个引号表示。

*immutable:*

不变的*:*

The property of a sequence whose items cannot be changed.

不能更改其项的序列的属性。

*traverse:*

遍历*:*

To iterate through the items in a sequence, performing a similar operation on each.

遍历序列中的项，对每个项执行类似的操作。

*search:*

搜索*:*

A pattern of traversal that stops when it finds what it is looking for.

一种遍历模式，当它找到它要找的东西时停止。

*counter:*

计数器*:*

A variable used to count something, usually initialized to zero and then incre‐ mented.

一种用于计数的变量，通常初始化为零然后再递增。

*invocation:*

调用*:*

A statement that calls a method.

调用方法的语句。

*optional argument:*

可选参数*:*

A function or method argument that is not required.

非必需的函数或方法参数。

**Glossary 9**

*file object:*

文件对象*:*

A value that represents an open file.

代表打开文件的值。

*reduction to a previously solved problem:*

还原为以前解决的问题*:*

A way of solving a problem by expressing it as an instance of a previously solved problem.

一种解决问题的方法，通过将其表示为以前已解决问题的实例。

*special case:*

特殊情况*:*

A test case that is atypical or non-obvious (and less likely to be handled correctly).

一个非典型的或者不明显的测试用例(并且不太可能被正确处理)。

**Glossary 10**

*list:*

列表*:*

A sequence of values.

一组值。

*element:*

元素*:*

One of the values in a list (or other sequence), also called items.

列表(或其他序列)中的一个值，也称为项。

*nested list:*

嵌套列表*:*

A list that is an element of another list.

是另一个列表的元素的列表。

*accumulator:*

累加器*:*

A variable used in a loop to add up or accumulate a result.

循环中用于累加或累积结果的变量。

*augmented assignment:*

增量赋值*:*

A statement that updates the value of a variable using an operator like +=.

使用+=等操作符更新变量值的语句。

*reduce:*

累积*:*

A processing pattern that traverses a sequence and accumulates the elements into a single result.

一种处理模式，遍历一个序列并将元素累加到单个结果中。

*map:*

映射*:*

A processing pattern that traverses a sequence and performs an operation on each element.

一种处理模式，遍历一个序列并对每个元素执行操作。

*filter:*

过滤器*:*

A processing pattern that traverses a list and selects the elements that satisfy some criterion.

遍历列表并选择满足某些条件的元素的处理模式。

*object:*

对象*:*

Something a variable can refer to. An object has a type and a value.

变量可以引用的东西。对象有类型和值。

*equivalent:*

值相等*:*

Having the same value.

具有相同的值。

*identical:*

相同（*ID*）*:*

Being the same object (which implies equivalence).

同一物体的(意味着相等)。

*reference:*

引用*:*

The association between a variable and its value.

变量与其值之间的关联。

*aliasing:*

别名*:*

A circumstance where two or more variables refer to the same object.

两个或多个变量引用同一个对象的情况

*delimiter:*

分隔符*:*

A character or string used to indicate where a string should be split.

用于指示字符串应在何处分割的字符或字符串。

**第一问：谈谈对SciPy相关的几个不同实体的认识。**

SciPy是一个基于Python的开源数据、科学和工程软件**生态系统**。SciPy是指几个相关但不同的实体：

SciPy Stack，是Python中科学计算的开源软件的集合，特别是一组指定的核心包。主要包括，Numpy，Scipy库，Matplotlib，Pandas等。

使用和开发这个栈的人的社区。

几个致力于Python中科学计算的会议——SciPy, EuroSciPy和SciPy.in。

SciPy库是SciPy栈的一个组件，提供了许多数值例程。

**第二问：谈谈Python数据处理的核心包，如NumPy/SciPy lib/Matplotlib等；如果有使用这些包的经验，谈谈使用心得**。

NumPy —— 数值计算的基础包。它定义了数组和矩阵类型以及基本操作。

SciPy lib —— 科学计算基础库。包括信号处理，优化，统计等等。

Matplotlib ——是一种成熟和流行的绘图软件，供出版品质的2D绘图以及初步的3D绘图。

IPython —— 是一个丰富的交互式界面，可让您快速处理数据和测试想法；

Sympy ——用于符号数学和计算机代数。

Pandas ——提供高性能，易于使用的数据结构。

标准安装的Python中用列表(list)保存一组值，但对于数值运算来说这种结构显然比较浪费内存和CPU计算时间。array模块不支持多维，因此也不适合做数值运算。而NumPy可提供便捷的N维数组操作，是基于向量化的运算，进行数值运算时Numpy数组比list效率高。使用NumPy要比直接编写Python代码便捷得多；NumPy比纯Python代码高效得多

Pandas是一种构建于Numpy的高级数据结构和精巧工具，快速简单的处理数据。要是没有pandas，对某行元素的操作可能需要很多循环