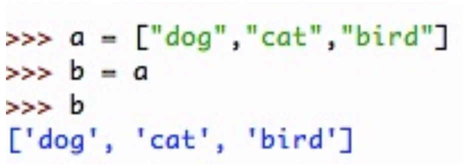
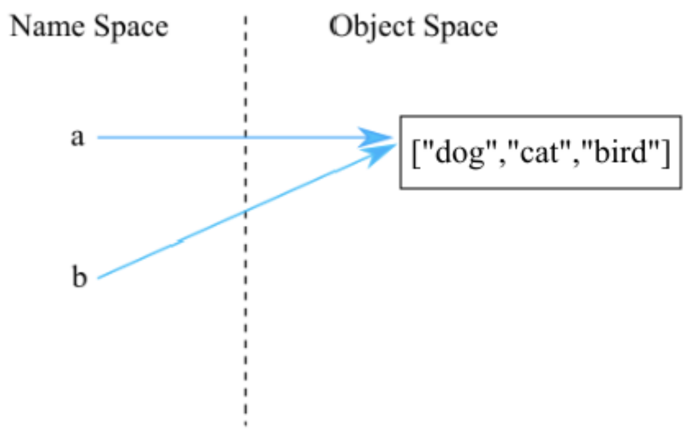
Programming Task 1:

Part 1. – Python commands of two different references point to the same collection-type object in memory:



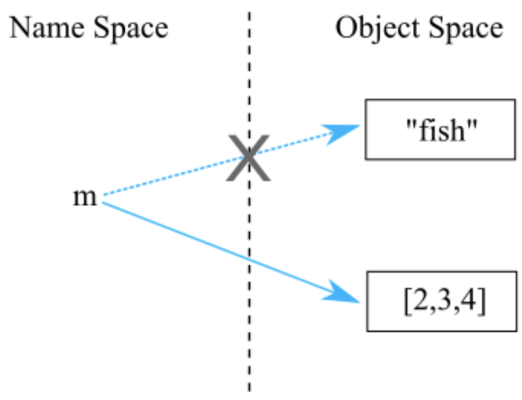
```
>>> a = ["dog","cat","bird"]
>>> b = a
>>> b
['dog', 'cat', 'bird']
```

Figure 1.1 – Reference diagram of two different references point to the same collection-type object in memory



Part 2. – It is not possible to have a single reference point to two or more distinct objects at the same time.

Figure 1.2 – Reference diagram of one reference point to two or more distinct objects at the same time



As Python keeps track of how many references each value has, it automatically cleans up values that have no reference. Which means, as one reference point to second or more distinct objects, the reference is no longer point to the first object. As shown in *Figure 1.2*, when we assign a new distinct object `[2,3,4]` to the same variable `m`, the variable `m` refers to a new value `[2,3,4]`, and the original reference to the old data object `"fish"` goes away. So the single reference cannot point to two ore more distinct objects at the same time.

Programming Task 2:

Advantages & Disadvantages of `readline()` vs. `readlines()`

`readline()` – reads one line at a time from a file. Newline is converted to `'\n'`, and line endings are also recognized as newlines. When the end of file is reached, `'\n'` is omitted at the end of the string, and it returns an empty string if keep reading.

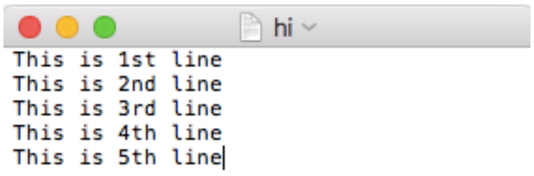
- It generates lines from a file much slower than `readlines()`, but because it does not store the entire file contents at once, it is a better method than `readlines()` when processing very large files that may exceed available memory.
- Cannot use `for … in …` with `readline()`, since it will read only one line and iterate it by character.
- More efficient in extracting from each line. `readline(n)`: `n` is the number of bytes to be read from the line/file. When `n` exceeds the line length, the entire line will be returned; or if `n` exceeds the rest of the unread line length, it will return the remaining line data of that line. In both cases, the rest of the line data will not be discarded and will be return on subsequent read operations.

`readlines()` – returns a list of each lines in the file. Newline is converted to `'\n'`, and line endings are also recognized as newlines. `'\n'` is only omitted when the end of file is reached.

- Automatically load all contents into a list of lines. Thus, we can use `for … in …` with `readlines()`.
- Works efficiently for small text file, but not memory efficient for large files since `readlines()` reads entire file content into memory which consumes more memory space than `readline()`. This can be a problem when the program is running on a computer that serves multiple users, especially when the file is large.

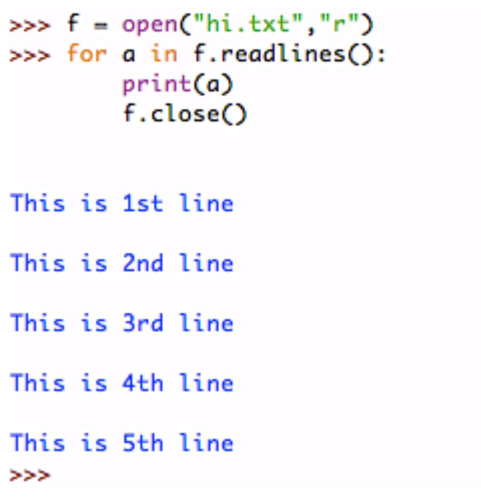
Python Codes

Text file `'hi.txt'`

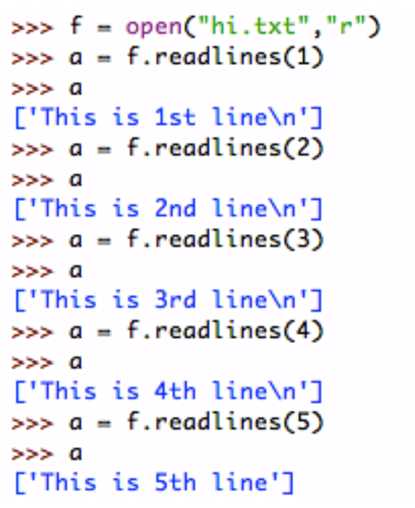


```
This is 1st line
This is 2nd line
This is 3rd line
This is 4th line
This is 5th line
```

`readlines()` within a loop:

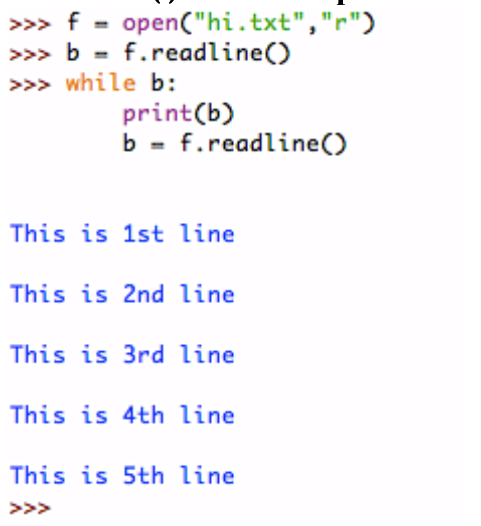


```
>>> f = open("hi.txt","r")
>>> for a in f.readlines():
        print(a)
        f.close()


This is 1st line

This is 2nd line

This is 3rd line

This is 4th line

This is 5th line
>>>
```

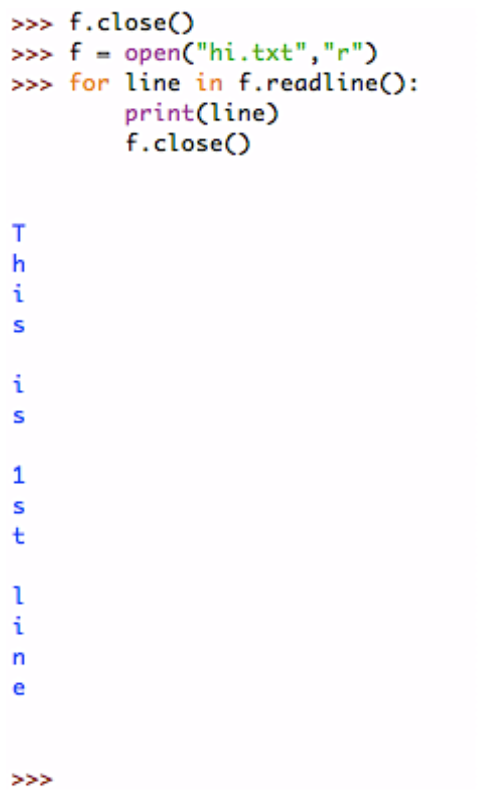


```
>>> f = open("hi.txt","r")
>>> a = f.readlines(1)
>>> a
['This is 1st line\n']
>>> a = f.readlines(2)
>>> a
['This is 2nd line\n']
>>> a = f.readlines(3)
>>> a
['This is 3rd line\n']
>>> a = f.readlines(4)
>>> a
['This is 4th line\n']
>>> a = f.readlines(5)
>>> a
['This is 5th line']
```

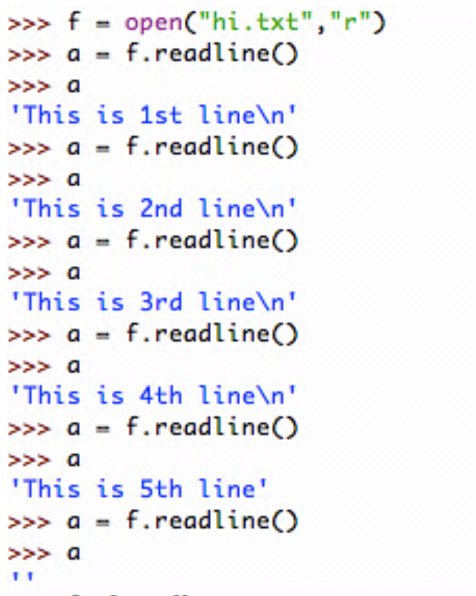
`readline()` within loop:



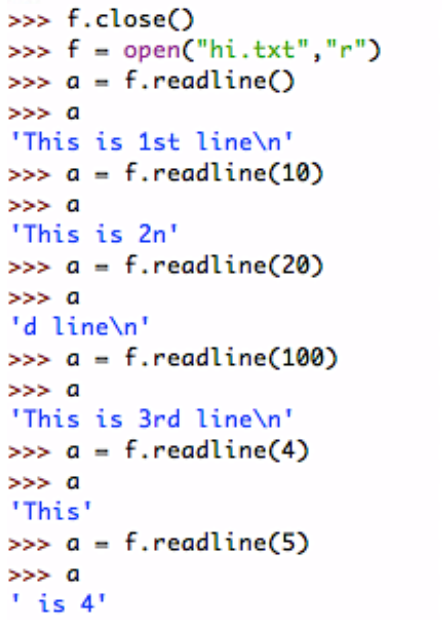
```
>>> f = open("hi.txt","r")
>>> b = f.readline()
>>> while b:
        print(b)
        b = f.readline()


This is 1st line

This is 2nd line

This is 3rd line

This is 4th line

This is 5th line
>>>
```



```
>>> f.close()
>>> f = open("hi.txt","r")
>>> for line in f.readline():
        print(line)
        f.close()


T
h
i
s

i
s

1
s
t

l
i
n
e


>>>
```



```
>>> f = open("hi.txt","r")
>>> a = f.readline()
>>> a
'This is 1st line\n'
>>> a = f.readline()
>>> a
'This is 2nd line\n'
>>> a = f.readline()
>>> a
'This is 3rd line\n'
>>> a = f.readline()
>>> a
'This is 4th line\n'
>>> a = f.readline()
>>> a
'This is 5th line'
>>> a = f.readline()
>>> a
''
```



```
>>> f.close()
>>> f = open("hi.txt","r")
>>> a = f.readline()
>>> a
'This is 1st line\n'
>>> a = f.readline(10)
>>> a
'This is 2n'
>>> a = f.readline(20)
>>> a
'd line\n'
>>> a = f.readline(100)
>>> a
'This is 3rd line\n'
>>> a = f.readline(4)
>>> a
'This'
>>> a = f.readline(5)
>>> a
' is 4'
```

Research Task 1:
PYTHON
The idea of Python first appeared in the late 1980s. In December 1989, the language is implemented by Guido van Rossum, a Dutch computer programmer at Centrum Wiskunde & Informatica (CWI) in the Netherlands. According to Python's author van Rossum, he planned to write an interpreter for the new scripting language: a descendant of an interpreted language called ABC. The programming language is named Python because its designer is a big fan of a BBC comedy series "Monty Python's Flying Circus". The new language is released in February 1991. Python is generally extensible as it keeps many of the features from ABC but also corrects its problems. Some of the other Python features and exceptions are inspired by Modula-3 since van Rossum has experience in Modula-2+ and he also talked to the author of Modula-3.

Python, as a multi-paradigm programming language, supports object-oriented programming, structured programming, functional programming, and aspect-oriented programming (including by metaprogramming and by magic methods). Extensions, including Design-by-Contracts (DbC) and logic programing, support many other paradigms in Python. Although both Java and Python are strongly typed languages, one of their differences is that Python uses dynamic typing, while Java uses static typing. This means that in Python, each reference except null variable only points to one object.

The goal of van Rossum is to make Python into "an easy and intuitive language, so anyone can contribute to its development code that is understandable as plain English suitability for everyday tasks, allowing for short development times". That is why we find the syntax and semantics of Python quite readable and simply to use. The main philosophy of Python is demonstrated by the 19 aphorisms in "PEP 20 - The Zen of Python" in Python Developer's Guide, aphorisms such as "Now is better than never."

Since Python has a comprehensive standard library, which includes string processing, software engineering, Internet protocols, and operating system interfaces, along with many third-party extensions and Python packages, all these built-in features it works great for small projects. However, because Python is a dynamically-typed language unlike Java as a statically-typed language, Python may not be a good choice for large projects that needs high-precision and large-scale data analysis since it has no compile-time checking feature, which makes it difficult to refactoring large project without extensive unit tests.

JAVA
Java was developed in 1991 and released in 1995 as a revolutionary programming language by a group of Sun Microsystems Inc. engineers called "Green Team" consists of James Gosling, Mike Sheridan, and Patrick Naughton. The initial setting for Java was a new language for digital devices in digital cable television industry, to be more specific, for small and embedded systems in consumer electronics such as set-top boxes, televisions, VCRs, and

toasters. The concept was too advanced at that time; however, the language turned out right for Internet programming. The name “Java” comes from the name of an island of Indonesia that produces the first coffee.

There are five goals in the design of Java: 1. Supports object-oriented programming. 2. Allows the same program to run on multiple operating systems. (Platform independent) 3. Contains built-in support for computer networks. 4. Able to execute code securely from remote sources. 5. Includes the good features of other similar languages. In the process of achieving these goals, Java programmers usually use extensions such as CORBA, Internet Communications Engine, or OSGI. Java has an automatic garbage collection. Similar to Python, Java also holds references to objects and automatically get rid of those has no referred objects.

Most of the Java syntax comes from C++. Some basic syntax to learn in Java includes: 1. Case sensitivity, 2. The first letter of all class names should be in uppercase. Different from Python, in Java, if a class name is consists of several words, and then the first letter of each word should also be in uppercase. 3. The first letter of all method names should be in lowercase. 4. File name should completely match the Class name. In Java, each element has a name. Identifiers are names for classes, variables, and methods.

Compared to Python, Java is more suitable for large projects. As it is an object-oriented programming language, it supports a large number of object classes. In addition, Java is quite suited for open-source project since it an easy to use language and its source code is portable than C and C++ languages. For example, you may notice that there are a lot of Java project repositories on GitHub. Because Java is a statically-typed language, it has type checking that can analyze the program during compile-time to prove the absence of type errors.

COMPARISON BETWEEN JAVA & PYTHON

After gained some basic knowledge about Java and Python, I believe that for a beginning programmer, the best choice to learn programming language is Python instead of Java. Since Python is a more readable language and its syntax is much more easier to remember and understand. The “Hello world!” code we familiar with is one of the good example of showing that Python can express the code in fewer lines than other languages like Java. For small projects, I would choose Python over Java since it is a single tool with extensive libraries and it is flexible to use for statistical and data analysis purposes.

If we are looking at a large-scale system, Java performs much better than Python. Java’s performing speed and scalability for building large-scale system is greater than Python, which is why many social networks rely on Java since they have a enormous data base. Python is poor at projects that require the language to perform highly specialized data tasks, since it does not have a platform like Java and there are still many packages that are not available to the Python community. However, if a project requires statically modeling and data visualization, Python can be a better choice than Java as it performs better on workflow integration.

References

Colenak.ptkpt.net. *Python (programming language)*. Retrieved from: http://colenak.ptkpt.net/_lain.php?_lain=3721#cite_note-venners-interview-pt-1-15

Colenak.ptkpt.net. *Python Syntax And Semantics*. Retrieved from: http://colenak.ptkpt.net/IT/108-1/Python-syntax-and-semantics_21107_colenak-ptkpt.html

Dreamincode.net. (2012, October 29). *What Is Python Best Suited For?* Retrieved from: http://www.dreamincode.net/forums/topic/297604-what-is-python-best-suited-for/

Ferg, S. (2009, October 3). *Static vs. Dynamic Typing of Programming Languages*. Retrieved from: https://pythonconquerstheuniverse.wordpress.com/2009/10/03/static-vs-dynamic-typing-of-programming-languages/

Freejabaguide.com. *History of Java programming language*. Retrieved from: http://www.freejavaguide.com/history.html

Javatpoint.com. *History of Java*. Retrieved from: http://www.javatpoint.com/history-of-java

Jelvis, T. (2014, April 13). *What sort of applications is Python ill-suited for?* Retrieved from: https://www.quora.com/What-sort-of-applications-is-Python-ill-suited-for

Keenan, T. (2016, March 23). *R vs. Java vs. Python: Which Is Right for Your Project?* Retrieved from: http://www.business2community.com/brandviews/upwork/r-vs-java-vs-python-right-project-01551890#slCkzmEECD3llAOI.97

Lebrero, D. (2016, May 19). *The Broken Promise of Static Typing*. Retrieved from: https://labs.ig.com/static-typing-promise

Oracle.com. Java SE Overview: *The History of Java Technology*. Retrieved from: http://www.oracle.com/technetwork/java/javase/overview/javahistory-index-198355.html

Pillai, P. (2004, June 18). *Introduction to Static and Dynamic Typing*. Retrieved from:

https://www.sitepoint.com/typing-versus-dynamic-typing/

Shiffman, H. *Making Sense of Java*. Retrieved from: http://www.disordered.org/Java-QA.html

The University of Maryland. (1007, December 9). The Python Programming Language. Retrieved from: http://groups.engin.umd.umich.edu/CIS/course.des/cis400/python/python.html

Tutorialspoint.com. *Java-Tutorial: Java – Basic Syntax*. Retrieved from: https://www.tutorialspoint.com/java/java_basic_syntax.htm

Research Task 2:

R

R is a computer programming language and run-time environment that is first created in 1991 and first announced tot the public in 1993 by Ross Ihaka and Robert Gentleman in New Zealand. In 1995, Martin Mächler, a mathematician, statistician from Zurich, Switzerland convinces R's authors to use the GNU General Public License to makes R free and open-source software. In 1997, the R Core Group consists of 20 leading statisticians and computer scientists from all around the world is formed to control the source code for R. The name of "R" comes from the initial of its authors Ross and Robert's initials. One of the best project that R community creates is the Comprehensive R Archive Network, CRAN, which is a repository where any users can contribute an extension to R as a package, which can downloaded and installed in R, as long as the extension meets the quality and licensing requirements set by the CRAN maintainers. There has been a steady growth of the number of R packages published on CRAN, from a zero start in 1996, there are now over 9,000 packages in 2016.

The syntax of R is in fact, quite similar to a statistical programming language called S programming language. R's software environment was written primarily in C, Fortran, and R. The R prompt is written as >. At the prompt, type an expression in R and hit the enter key. Then R will evaluate the expression by performing a computation, and returns a value. Expressions in R are similar but also different from that of Python. Some of the expressions that are different from Python are: In R, 7^2 is the same as 7**2 in Python. 9%/%2 is same as 9//2 in Python. 11%% 7 returns the same value as 11%7 in Python. Also, in R, results that are thirteen digits and more will be expressed in scientific notation. While Python uses scientific notation to express the results when the result of a flowing-point operations gets really big.

In R, there are two rules on the variable names: 1. The name may not start with a digit or underscore. 2. The name may contain numbers, characters, and some punctuation – period and underscore are ok, but most others are not. There are many features in R that are quite useful for statistical purposes. R has a built-in search function where users can get help on a certain topic by typing help.search("topic"). Checking for the presence of NA values using is.na() is also a common feature in R. Another aspect of R is that users can subset a large set of data by inclusion and exclusion.

One of the most helpful and basic thing to learn about R is the various types of objects in R. Lists are ordered container of objects that can be anything includes vector, data frame, list, matrix, arrays, etc. A vector is an ordered container of literals. The elements in a vector must be the same type. For example, if one of the elements in vector Weight is 175, then all the other elements must all be numeric. Data frame is an ordered container of vectors. These vectors can be in different types, but they must all be the same length. For example, a data frame called Family contains vector age and vector firstName. Vector age has 10 numeric elements, then vector firstName must also has 10 elements, but it can be character. Matrices and arrays are rectangular collection of elements. The dimensions of them can be two, three, or more. The elements in a matrix or an array are homogeneous primitive, which means that elements should all be numeric or all be character.

COMPARISON BETWEEN PYTHON & R

There is statistical software with a GUI, such as Minitab or SPSS. There are also programming languages such as Java and Python, but R programming language and environment is more like a "home base" software for performing many data analytic tasks. R has many advantages. First, like Python, it is a free statistical software. This allows its users to have more flexibility and freedom to run and study the program. Second. It is very easy to replicate analysis. Third, it provides sound numerical methods. Last but not least, it has a large community of contributors that contributes a great importance to the successful development of R. It is a high-level scripting language, a statistical programming language that allows custom analysis and interactive exploratory data analysis. Compared to Python, R has more frequent releases and bug-fix releases credits to the large and active R community. For example, R-help, R-devel mailing lists, and the one I usually use, Stack Overflow.

For a data science project, both Python and R can perform statistical data analysis and data modeling. Python is designed to be a more general and easy to use language, while R is designed with the statistical concept which has very sophisticated graphics capabilities that are better than the majority of statistic packages. However, for a large data science project, we may choose Python over R. Since R is much slower in speed than Python, which can be big problem in dealing with large data objects because the size of data set can be large. Also, during computation, some of the data values can be copied multiple times which will make the case worse since all the data values are in the environment and user needs to rewrite the entire code base when need to switch to different evaluation model

As a user of R, I find RStudio to be quite useful tool. RStudio is a free and open-source integrated development environment for R. The feature that users most commonly use in

RStudio is the a CRAN package called rmarkdown which you can generate a report of your statistic analysis and automatically saves it under the same folder of your R code. The output is dynamic and it can be generate into many formats such as PDF, Word, and HTML. While, in Python, when you save a file as .py, it is just a text file, which is difficult to track the process of analysis process and to get back to where you left.

References

Ihaka, R. *The R Project: A Brief History and Thoughts About the Future*. Retrieved from: https://www.stat.auckland.ac.nz/~ihaka/downloads/Massey.pdf

Keenan, T. (2016, March 23). *R vs. Java vs. Python: Which Is Right for Your Project?* Retrieved from: http://www.business2community.com/brandviews/upwork/r-vs-java-vs-python-right-project-01551890#slCkzmEECD3llAOI.97

Smith, D. (2016, March 4). *Over 16 years of R Project History.* Retrieved from: http://blog.revolutionanalytics.com/2016/03/16-years-of-r-history.html