Access the slides and files here:

https://github.com/j-berg/bioinformatics_bootcamp

#3.1

Transferring files between CHPC and personal computer

IGV

Data types

Functions

Internal and External libraries

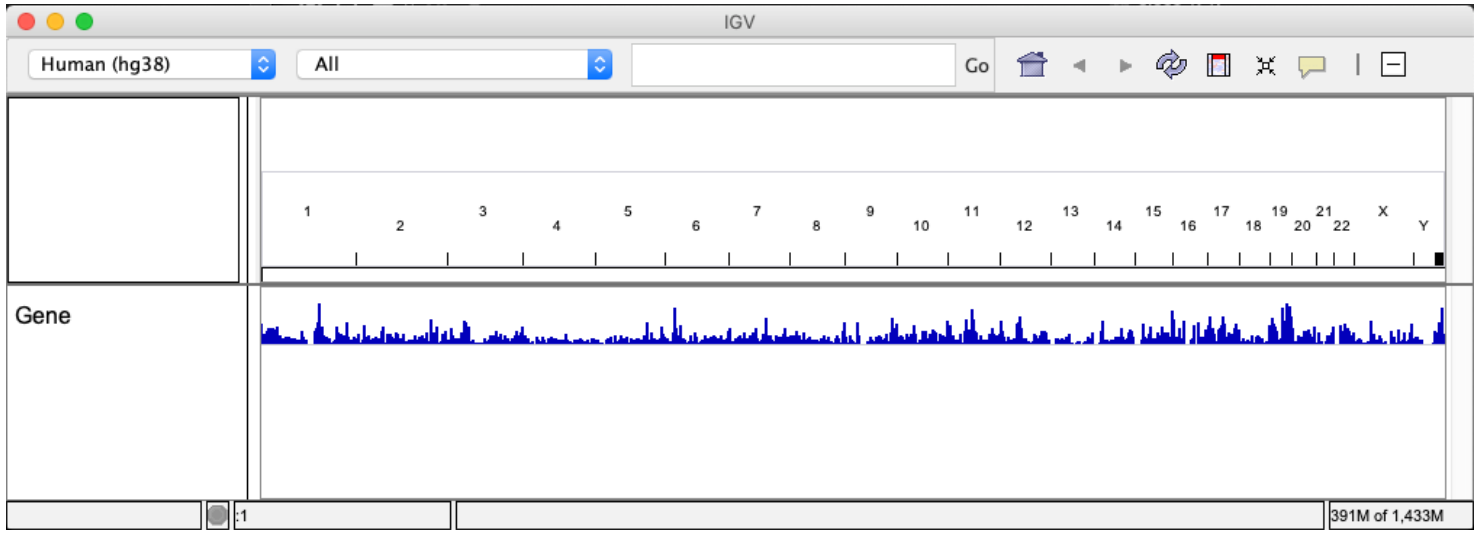
Transferring files to your personal computer

- From your personal computer/terminal

```
$ scp uNID@notchpeak.chpc.utah.edu:~/path/to/file.bam ./
$ scp uNID@notchpeak.chpc.utah.edu:~/path/to/file.bam.bai ./
```

Visualizing read pile-ups with IGV

- Download IGV:
  - https://software.broadinstitute.org/software/igv/download
- Drag and drop BAM file into viewer



Open python in terminal or write a script

```
$ python
>>>

$ python script.py
```

The foundations of coding

- Data types:
  - Integers
  - Floats & Doubles
  - Strings
  - Arrays
  - Sets
  - Dictionaries
  - Objects

Integers

- Numeric types refer to the precision of the value
- Integers: whole numbers

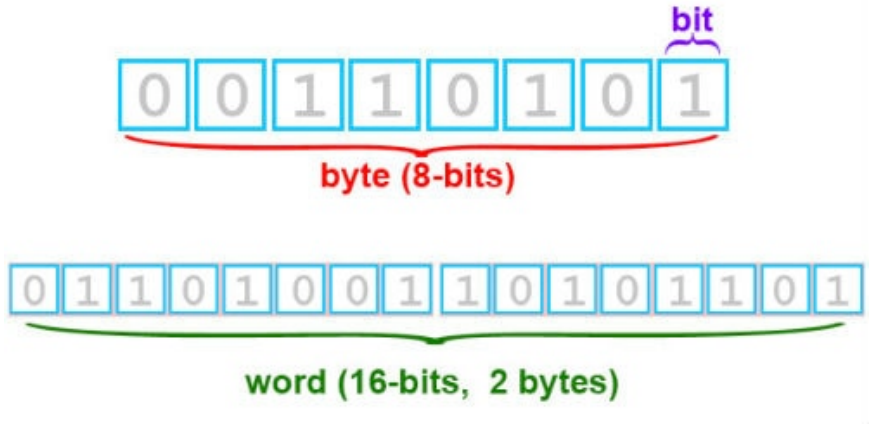
```
>>> x = 1
>>> type(x)
<class 'int’>

>>> y = int(1.0)
>>> type(y)
<class 'int’>
```

Floats

- Floating point value
- Represents a fractional value
- Depending on your system, can have increased precision
- Ever wonder what the difference is between a 32-bit and 64-bit OS?
  - 32-bit can access 2^{32} memory addresses
  - 64-bit can access 2^{64} memory addresses

Let’s talk about how memory works

- A bit is can be equal to 0 or 1 determined by an presence or absence or an electrical charge at a given position on the computer chip
- A byte is a collection of 8 bits that represent a higher value
  - A byte can store ”A”, or “X”, or “$”, etc.
    - 0010 0001 = “!”
    - 0011 0000 = “0”
  - In general: add 1 bit, double the number of patterns
    - 1 bit - 2 patterns
    - 2 bits - 4
    - 3 bits - 8
    - 4 bits - 16
    - 5 bits - 32
    - 6 bits - 64
    - 7 bits - 128
    - 8 bits - 256 - one byte
    - Mathematically: n bits yields 2^n patterns (2 to the nth power)



https://web.stanford.edu/class/cs101/bits-bytes.html
https://www.colocationamerica.com/images/bit-byte-comparison.jpg

Floats and doubles

- Floating point value
- Represents a fractional value
- Depending on your system, can have increased precision
- Ever wonder what the difference is between a 32-bit and 64-bit OS?
  - 32-bit can access 2^{32} memory addresses
  - 64-bit can access 2^{64} memory addresses
  - Float -> up to 32 points of precision
  - Double -> up to 64 points of precision (Python will call this a “float”)
- By being able to represent higher precision values, able to develop more complicated procedures, etc

Mixing data types

- Combining datatypes may lead to incompatibilities, errors, so be aware!
- Converting a float to an integer will lose precision

```
>>> x = 1.23456789012345678 9
>>> y = int(x)

>>> print(x)
1.2345678901234567
>>> type(x)
<class 'float’>

>>> type(y)
<class 'int’>
>>> print(y)
1
```

Strings

- Represents text instead of numbers
- Generally indicated with quotes
- Can force integers and floats/doubles to strings

```
>>> x = “Hello world!”
>>> print(x)
Hello world!

>>> y = 1.23456789
>>> y = str(y)
>>> print(x, y)
Hello world! 1.23456789
```

Arrays

- A sequential, ordered list of values

```
>>> x = [1,2,3]
>>> y = [‘a’, ‘b’, ‘c’]
>>> z = [1, 2, 3, “d”, “e”]
```

- What happens when we try this?

```
>>> zz = [x, y]
[[ 1 ,  2 ,  3 ],
 [‘a’, ‘b’, ‘c’]]
```

- Access something from an array
- What does this do?

```
>>> zz[1][1]
```

- Why is the output ‘b’?

Indexing

- Indices start at 0 in Python
- This will vary language to language (R starts at 1)

Sets

Try this:

```
>>>  v = set([1,2,3,4,4])
```

Or

```
>>>  v = {1,2,3,4,4}
```

```
>>> v
{1, 2, 3, 4}
```

Why does this happen? Let’s talk about another similar data type

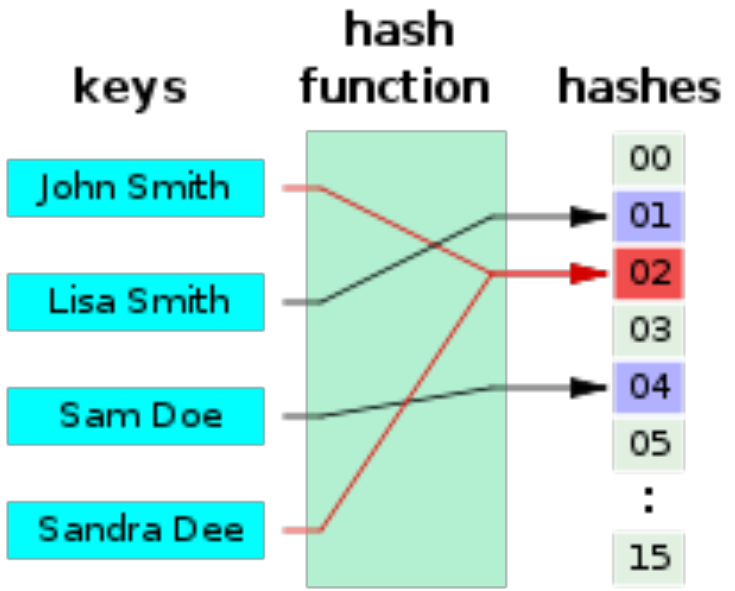
Dictionaries (or hashes)

```
>>> d = {}
>>> d[“a”] = “Hello”
>>> d[“b”] = “world!”
>>> d
{'a': 'Hello', 'b': 'world'}

>>> d[“a”]
‘Hello’
>>> d[“a”] = “Goodbye”
>>> d
{'a': Goodbye ', 'b': 'world'}
```

How to dictionaries, hashes, and sets work?

- One value associated with a given key
- Unordered (unlike a list/array)
- Two identical values go to same place in memory, so only returns one iteration

- What does a hash do?
  - A function that converts a given input to another (such as a position in memory)
  - The hash function is constant, the same input will always give the same output
  - Able to map a value of variable size to a fixed-size output



https://en.wikipedia.org/wiki/Hash_function

Array vs dictionary performance

- As arrays are ordered, they are searched in order
  - n steps
- A dictionary finds if a value is set because it translates the string to a hash position to see if its set
  - One step

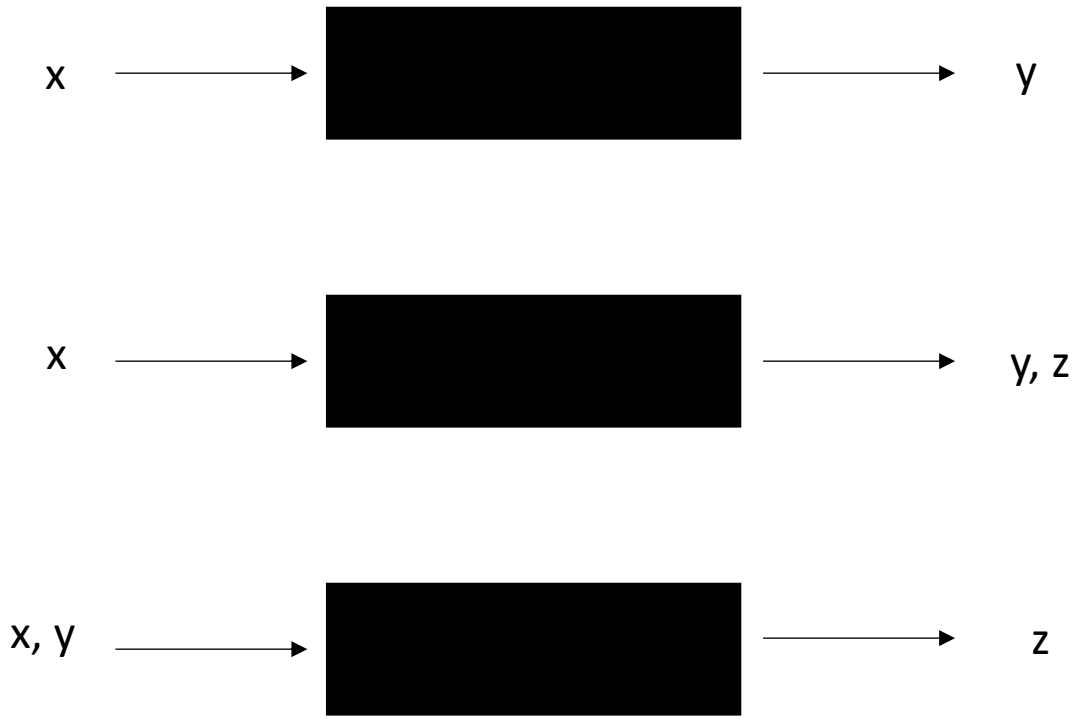
- Dictionary keys are converted to a hash which corresponds to a given spot in the computer memory
- To search for the value associated with a key, it only has to re-hash the key value you give it, go straight to that position in memory to check for a value, and return the value if it exists

Array vs dictionary performance

- For example, you want to find Sal's office, but don't know the office number, so you have to search office by office until you find Sal's office -> array/list
- Or, you know Sal's office number already, so you go straight there -> dictionary/hash

https://unsplash.com/photos/NIrgENd0sAY?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText

Functions: A not-so-black box



Functions

- Useful for commonly run tasks
- Useful for functions others might want to run themselves
  - Packaged and distributed as “libraries”

Libraries

- A collection of functions

- Download as follows:
  - Basic packages:
    - Included with Python
  - External packages:
    - Download with conda

```
$ conda install pandas
```

Loading a library

- Need to give the session or script access to the functions from the library

```
>>> import os
>>> import pandas
```

Homework

- Transfer one of the alignment files to your personal computer and open in IGV. Find a gene whose transcripts (isoforms) seem to be differentially expressed.

- Create a dictionary of a mock classroom of students and their test scores.
- Create an array of students' names, include names that were not in the dictionary.
- Access each student in the array, determine if they are in the dictionary programmatically, and print out a sentence that states "[student] got a [score] on their test."