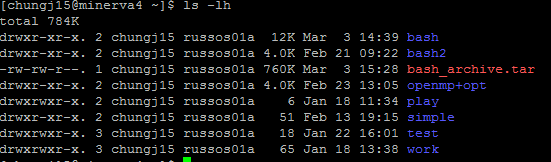
Introduction to Scientific Computing (BSR1015)

Assigned February 28th, Due March 14th

1. Create a single archive file of your $HOME/bash directory using the “tar” command and name this file “bash\_archive.tar”.
   1. Show how you did it.



* 1. Check/show the size of “bash\_archive.tar” file.

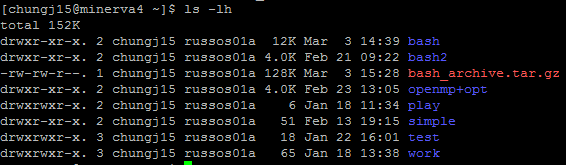


* 1. Check/show the size of $HOME/bash directory



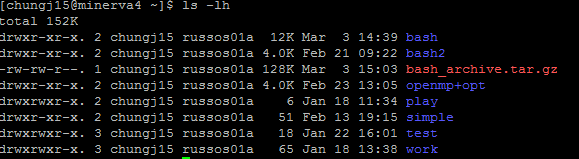
1. Compress the file “bash\_archive.tar” using the “gzip” command.
   1. Check/show the size of this compressed file.





* 1. Create a single gzipped archive file of your $HOME/bash directory using the “tar” command and name this file “bash\_archive.tar.gz”. Check/show the size of “bash\_archive.tar.gz” file.

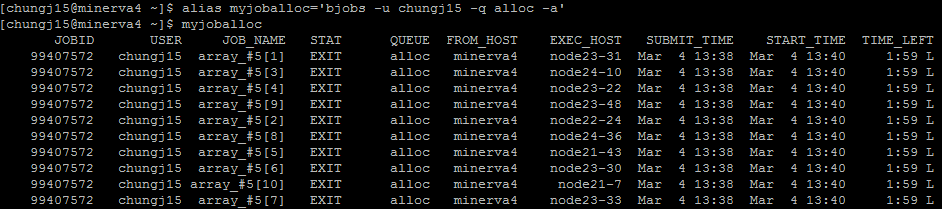




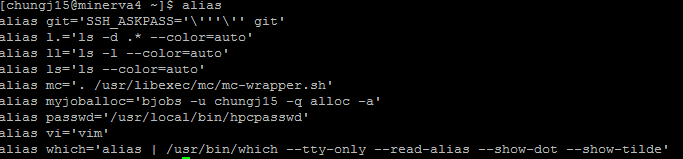
1. Aliases and bjobs command. (Hint: You will need to use the ‘bjobs’ command. You may want to read the documentation using the man page and /or bjobs -h.)
   1. Create an alias “myjobsrun” to show only your running jobs in all queues on the Minerva cluster.



* 1. Create an alias “myjoballoc” to show only your jobs in the alloc queue on the Minerva cluster.



* 1. Show all of your aliases.

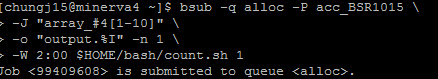


* 1. What command shows the information about all jobs on the Minerva cluster?

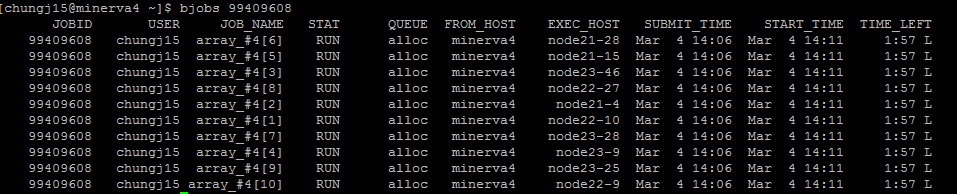
bjobs –u all –a (display jobs in all states)

Source: IBM

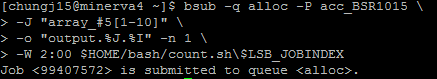
1. Submit 10 jobs as a single job array which will run the $HOME/bash/count.sh program. The job output files should be in the format: output.<arrayindex>. (Example: output.1, output.2 and etc.)
   1. Show how you did it.



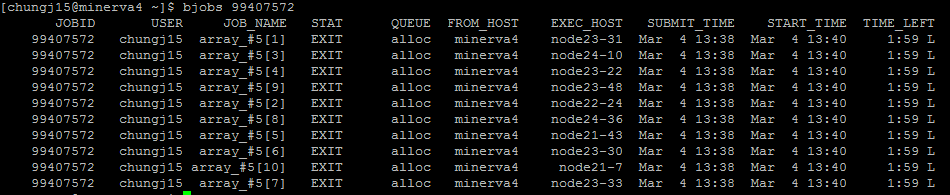
* 1. Report the JOB\_ID of your array job.



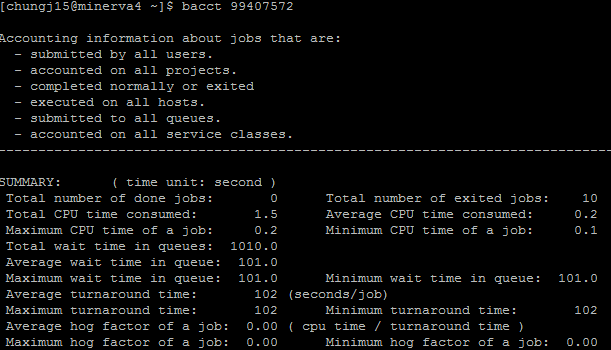
1. Submit 10 jobs as a single job array which each run the $HOME/bash/count.sh program with the input parameter corresponding to the index of the job in the array (E.g., job[1] in the array will run: $HOME/bash/count.sh 1, … job[10] in the array will run: $HOME/bash/count.sh 10).
   1. Show how you did it.



* 1. Report the JOB\_ID of your array job.



* 1. Check/show/print how long each job in job array ran.



1. Complete the lines with python statements that will print the corresponding output.
   1. >>> L = [1, 2, 3, 4, 5]

>>> **Sum (L)**

15

* 1. >>> L = [1.0, ‘two’, 3]

>>>**L\*3**

[1.0, 'two', 3, 1.0, 'two', 3, 1.0, 'two', 3]

* 1. >>> L = [12, 3, 52, -1, 77, 58, -99, 6, 27]

>>> **L =sorted(L)**

>>> print L

[-99, -1, 3, 6, 12, 27, 52, 58, 77]

* 1. >>> L = [‘ABCD’, ‘BCDE’, ’CDEF’, ‘DEFG’]

>>> **[word[1:3] for word in L]**

[‘BC’, ‘CD’, ‘DE’, ‘EF’]

* 1. >>> L = [1, 2, 3, 4, 5]

>>> **[L\*\*2 for L in L]**

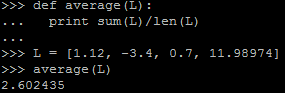
[1, 4, 9, 16, 25]

1. Write a python function, *average*(), that calculates the average of numbers given in a list. For example,

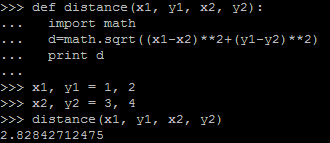
>>> L = [1.12, -3.4, 0.7, 11.98974]

>>> *average*(L)

2. 602435



1. Write a python function, *distance*(), that returns the distance between two points (*x*1, *y*1) and (*x*2, *y*2). For example,



>>> x1, y1 = 1, 2

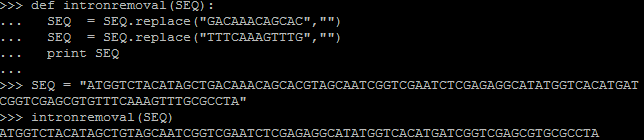
>>> x2, y2 = 3, 4

>>> *distance*(x1, y1, x2, y2)

2.8284271247461903

1. In the following DNA string, the substrings in red are introns. Write a python code that deletes these introns from the given string and concatenates the exons to form a new string that is ready for translation.

“ATGGTCTACATAGCTGACAAACAGCACGTAGCAATCGGTCGAATCTCGAGAGGCATATGGTCACATGATCGGTCGAGCGTGTTTCAAAGTTTGCGCCTA"



1. Given two strings ***a*** and ***b*** of equal length, the *Hamming distance* between these two strings, *dH*(*a,b*), is defined as the number of corresponding symbols that differ in ***a*** and **b**. Complete the lines in the following script computing the *Hamming distance dH*(*a,b*).

>>> a = “GAGCCTACTAACGGGAT”

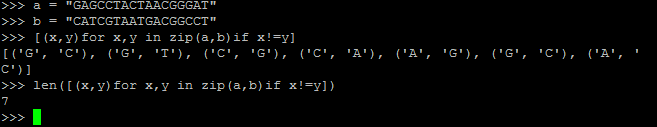
>>> b = “CATCGTAATGACGGCCT”

>>> *[(x,y)for x,y in zip(a,b)if x!=y]*

>>> [('G', 'C'), ('G', 'T'), ('C', 'G'), ('C', 'A'), ('A', 'G'), ('G', 'C'), ('A', 'C')]

>>> *len([(x,y)for x,y in zip(a,b)if x!=y])*

7



Hint. You may need to use *zip*() function.