MayoWorkflow.wdl Integration Test output in excel sheet format

* Excel sheet format (".tsv", ".tsv" or ".xlsx").
* The python working model of a spreadsheet is the pandas DataFrame.
* This code parses results directory(s) content into a DataFrame.
* Options to write DataFrame to a file, display on the command line or both.
* Option to combine with PASS | FAIL spreadsheet criteria for usage with auto-test suite (Jenkins etc).
* Demonstrate the use of jupyter notebook to develop python code in situ.

MayomicsVC Research branch (private) (https://git.ncsa.illinois.edu/mayomics/MayomicsVC/tree/master/testing)
StackOverflow readable time stamp (https://stackoverflow.com/questions/16060899/alphabet-range-python/31888217)

In [57]:
```
# %%writefile ../../MayomicsVC/testing/integration_test.py
import os
import sys
import string
import time
import datetime
import pandas as pd
import numpy as np

def get_time_sequence_string(decimal_shift=3):
    """ Usage: get_time_sequence_string = get_time_sequence_string(decimal_shift=3) """
    alpha_list = list(string.ascii_uppercase)[0:10]
    time_seq_int = np.int_(list(np.str_(int(time.time() * np.maximum(10**decimal_shift, 1)))))
    time_sequence_string = ''
    for d in time_seq_int:
        time_sequence_string += alpha_list[d]
    return time_sequence_string


def get_readable_time_stamp(n_digits=3):
    """ Usage: time_stamp_string = get_time_stamp(n_digits=3) localtime """
    return datetime.datetime.now(datetime.timezone.utc).strftime("%H_%M_%S_%f_%Z_%Y_%m_%d")


def get_test_results_dataframe(x_dir):
    """ Usage: return_codes_dataframe = get_test_results_dataframe(x_dir)
    args:
        x_dir:          the directory with the "call_..." subdirectories (else you get nothing)

    returns:
        rc_df:          pandas dataframe with the return codes and size of various output files
    """
    DATAFRAME_DEFAULT_EMPTY_VALUE = 'unk'
    FAILED_RETURN_CODE_READ = '-1'
    good_return_codes_list = ['0', '0\n']
    check_files_dict = {'stderr':['ERROR', 'error', 'Error'], 'stdout':['START', 'Finished']}

    #     This variable might be compared to the "call" entries in the .wdl tree.
    call_dirs = os.listdir(x_dir)
    call_dir_list = []
    call_dir_count = 0
    #     Get the rows list - parse the directories that begin with "call" vs getting them from the .
```

```
wdl files
    for call_dir in call_dirs:
        if os.path.isdir(os.path.join(call_dir, x_dir)) and call_dir[0:4] == 'call':
            call_dir_count += 1
            call_dir_list.append(call_dir)

    #     Create the list of things that will be reported in each call directory and initialize the d
ataframe
    cols_list = ['rc', 'bam', 'bam.bai', 'stderr', 'stdout']
    rc_df = pd.DataFrame(index=call_dir_list,columns=cols_list).fillna(DATAFRAME_DEFAULT_EMPTY_VALUE)

    #     Check the directories in this tree against the column list
    for dir_name, dir_list, files_list in os.walk(x_dir):
        for filename in files_list:
            full_filename = os.path.join(dir_name, filename)
            if filename in cols_list:
                if filename == 'rc':
                    with open(full_filename, 'r') as fh:
                        lines = fh.readlines()

                    if len(lines) > 0:
                        for call_dir in call_dir_list:
                            if call_dir in dir_name:
                                if lines[0] in good_return_codes_list:
                                    rc_df.loc[call_dir, 'rc'] = str(lines[0]).strip()
                                else:
                                    try:
                                        rc_df.loc[call_dir, 'rc'] = str(lines[0]).strip()
                                    except:
                                        rc_df.loc[call_dir, 'rc'] = FAILED_RETURN_CODE_READ
                                        pass

            if filename in list(check_files_dict.keys()):
                for call_dir in call_dir_list:
                    if call_dir in dir_name:
                        with open(full_filename, 'r') as fh:
                            lines = fh.readlines()

                        if len(lines) > 0:
                            for line in lines:
                                for check_word in check_files_dict[filename]:
                                    if check_word in line:
                                        rc_df.loc[call_dir, filename] = check_word
```

```
                                continue

            # Report the File sizes - code version pending full list of required files
            else:
                for call_dir in call_dir_list:
                    if call_dir in dir_name:
                        fname, fext = os.path.splitext(filename)
                        this_file_data = os.stat(full_filename)
                        if fext[1:] == 'bam':
                            rc_df.loc[call_dir, 'bam'] = str(this_file_data.st_size)
                        elif fext[1:] == 'bai':
                            rc_df.loc[call_dir, 'bam.bai'] = str(this_file_data.st_size)

    return rc_df
```

Example Output format - (or get PASS | FAIL options with tolerance table)

In [58]:
```
x_dir = '/Users/yo/zzIForge/fullyJan10/'
if not os.path.isdir(x_dir):
    print('directory not found\n', x_dir)

somedf = get_test_results_dataframe(x_dir)
somedf
```

Out[58]:

	rc	bam	bam.bai	stderr	stdout
call-DHVC	0	unk	unk	unk	unk
call-realign	0	65349394	1431360	unk	START
call-align	0	59500466	1431328	unk	Finished
call-bqsr	0	67462520	1431360	unk	Finished
call-dedup	0	65322724	1431360	unk	Finished
call-haplotype	0	67462520	1431360	unk	Finished
call-merge	0	59500466	1431328	unk	unk
call-vqsr	0	unk	unk	unk	Finished
call-trimseq	0	unk	unk	unk	Finished
call-DAB	0	65349394	1431360	unk	unk

Time stamp string format examples

In [39]:
```
t_seq_string = get_time_sequence_string(decimal_shift=4)
print('Sequence string: (file-system sortable)\t',t_seq_string, '\n')

t_stamp_str = get_time_stamp(n_digits=6)
print('Human-readable, UTC time stamp:\t\t', t_stamp_str)
```

```
Sequence string: (file-system sortable)  BFEHHHHBEJJDCJ

Human-readable, UTC time stamp:          02_05_49_933455_UTC_2019_01_18
```

In [ ]:
```

```