Task Runner Assignment

Problem statement

The assignment is to implement a simple task pool class in python (2.6 or 2.7) that supports the following methods and features. A task will consist of a list of shell commands and be considered a failure if any have a non-zero exit code, and a success if all commands return zero. The task execution must cease once it is considered failed. Please design the task pool for tasks that will each take seconds to minutes to complete entirely. You will have two hours to complete as much of the assignment as possible.

Task / TaskStatus / TaskRunner definitions

```
class Task(object):
  def __init__(self, commands, capture_output=False, exclusive=False):
    """Creates a task object
    Args:
      commands: a list of shell commands
      capture_output: true if the stdout / stderr of the commands needs to be recorded
      exclusive: true if the task cannot run concurrently with other tasks
    """
    self.commands = commands
    self.capture_output = capture_output
    self.exclusive = exclusive

class TaskStatus(object):
  STATUS_COMPLETE = "complete"
  STATUS_QUEUED = "queued"
  STATUS_RUNNING = "running"

  RESULT_OK = "ok"
  RESULT_FAIL = "fail"

  def __init__(self, status, result=None, stdout=None, stderr=None, info=None):
    """Create the TaskStatus

    Args:
      status: the status of the given task (use constants)
      result: the result of the given task if completed (use constants)
      stdout: the stdout if output was requested
      stderr: the stderr if output was requested
      info: a string of returncode and command that failed if result is failed
    """
    self.status = status
    self.result = result
    self.stdout = stdout
    self.stderr = stderr
    self.info = info
```

```
class TaskRunner(object):
  def __init__(self, concurrency):
    """Create the task runner.

    Args:
      concurrency: the maximum number of tasks that can run in parallel
    """

  def add_task(self, task):
    """Adds a task to the task pool.

    This may be called whether or not the task pool is currently running.

    Args:
      task: a Task object

    Returns:
      the task id associated with the task
    """

  def start(self):
    """Asynchronously starts the task pool.

    This should cause the TaskRunner to begin executing tasks, however, it does
    not need to wait for execution to begin in order to return.
    """

  def stop(self):
    """Synchronously stops the task pool.

    This function should wait until all running (but not queued) tasks have
    completed before returning.
    """

  def status(self, task_id):
    """Get the status of a given task.

    Args:
      task_id: the task id returned by add_task

    Returns:
      TaskStatus object for the given task
    """

  def cleanup(self, task_id):
    """Removes the task status from the task runner if the task has completed

    Args:
      task_id: the task id returned by add_task
    """

  def tasks(self, state=None):
    """Returns metadata regarding tasks currently added to the task pool.
```

```
  Args:
    state: a task state (see TaskStatus for enum defs) to filter for if
      specified, if not specified, return data regarding all tasks

  Returns:
    a list of (task_id, state, queue_position, task) defined as:
      task_id: the task id returned by add_task
      state: the state of the task (queued, running, complete)
      queue_position: the position of the task in the queue
      task: the task object
  """

# OPTIONAL - you may skip implementation of this function
def wait(self):
  """OPTIONAL) Wait for all tasks added to the task pool to execute.
  """

# OPTIONAL - you may skip implementation of this function
def cancel_task(self, task_id)
  """Cancels the task with the given task ID if it has not yet been started.

  Args:
    task_id: the task id returned by add_task

  Returns:
    True if the task was canceled successfully, false otherwise
  """
```

Evaluation criteria

1. Design - is the code readable and reusable where possible? How much thought has gone into the interface design?
2. Correctness - does the code do what it is supposed to with regard to the instructions?
3. Robustness - does the implementation handle errors and corner cases sensibly?

Submission

Your submission must run, and a test program which exercises the code is highly encouraged.

Once you have completed the assignment, please spend up to 30 minutes to document the design including tradeoffs made as well as shortcomings in the solution. This time will not be counted against the two hours allotted for the assignment.