Integrated Research – Graduate/Intern Software Engineer Assignment

An Agile Scrum 'Task Board'

We have a challenge for you!

Write a command-line based, object-oriented agile scrum task board simulator. Users should be able to create user stories and tasks for the task board, and move the tasks as they work on them until the task or story is completed.

Definition

User Story: a complete project or feature that can be delivered to a user.

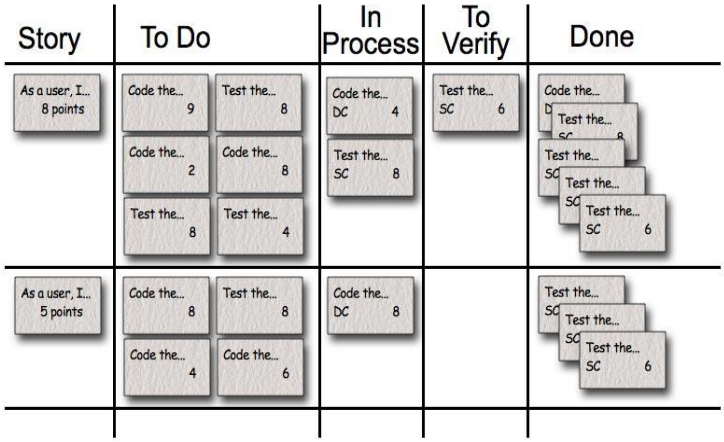
Task: a Story is broken down into actionable items, called tasks, which need to be completed to deliver the feature.

Sprint: a set period of time (2-4 weeks) for development. Multiple stories (features) are taken on for each sprint.

Backlog: a queue of Stories (features) that have yet to be taken on.

See https://www.scrum.org/Resources/What-is-Scrum and https://www.scrum.org/Resources/Scrum-Glossary

When practicing agile scrum, we make the sprint backlog visible by putting it on a scrum task board. Team members update the task board continuously throughout the sprint, adding and moving cards as work progresses. The scrum board looks like this:



Each row on the scrum board represents a user story. Each story is broken down into a number of actionable items called tasks. The card for each task starts in the "To Do" column and team members move it through the columns as they work. The definitions for the columns are:

Story: Contains user story descriptions for each row

To Do: Contains all the task-cards that are yet to be actioned for a story in the current sprint

In Process: Contains all the task-cards that are being worked on that day

To Verify: After a task is completed, it has to be verified by other team members before it is considered "Done"

Done: Cards pile up over here when they are done

(continued over page…)

Rules of the task board:

1. Every task has to be associated with a user story
2. Once the task is created it will be placed in the "To Do" column
3. Each task has to go through the "To Do", "In Process", "To Verify", and "Done" columns
4. Each task that is not in the "Done" column can be moved back to "To Do" or "In Process" column if necessary
5. The user story can only be marked completed once all the tasks have been completed.
6. When deleting a user story, all tasks associated with the story will be deleted
7. Board state must be persistent

User Input/Control:

Input can be via the command-line itself, or by an internal shell, or both.

Input	Description
`create story <id> <description>`	Create a new user story with the given ID and description
`list stories`	List all user stories that have been created, including Id's
`delete story <id>`	Delete the user story with the given ID
`complete story <id>`	Mark the user story with the given ID as completed
`create task <storyId> <id> <description>`	Create a new task with the given task ID and description, and associate it with the given storyId
`list tasks <storyId>`	List all the tasks associated with the given storyId
`delete task <storyId> <id>`	Deletes the task with the given ID associated with the given storyId
`move task <storyId> <id> <new column>`	Move the task to the new column (To Do, In Process, etc)
`update task <storyId> <id> <new description>`	Update/Modify a task's description

Constraints/Other Requirements:

Please note these further requirements:

- Your implementation should return a 0 errno on successful completion, 1 for an error
- Output should be undecorated – no headings, ascii lines, or other formatting
- List output should be simple item-per-line output. That is 'list stories' simply lists each story description one per line; 'list tasks <storyId>' simply lists each task of the story one per line.

Submission:

Please remember to consider all rules and constraints/requirements when implementing the commands listed, and consider error-handling wisely.

Submissions must be a ZIP archive of all files required for your solution, including but not limited to:

- Source files
- Project solution/make files
- Documentation
- Test cases

The submitted ZIP archive should NOT contain any executables or intermediary build/object files.

Please provide your solution within a week. Take your time and provide your best work.