CSE 1325: Object-Oriented Programming

Lecture 11

Exam #1 Retrospective
Intro to Agile Processes

Mr. George F. Rice

george.rice@uta.edu

Based on material by Bjarne Stroustrup

www.stroustrup.com/Programming

ERB 402

Office Hours:

Tuesday Thursday 11 - 12

Or by appointment

Tests are Graded

- Cross-checks are NOT complete
  - Thus, scores are not posted
  - Graded papers will be available shortly for review
- A few issues also were / need to be addressed
  - Duplicate MC answer - 2nd was graded as correct whether selected or not
  - First definition was skipped on at least a few exams (need to count) – perhaps thought it was a header?
  - Enum classes and string streams were... bad

Point Allocation by Skill

Scores ranged from 38.5 to 103.5 (104 possible)

Section	2018.0	2017.0
I Vocabulary	94%	93%
II Multiple Choice	71%	81%
Patterns	86%	68%
UML Use Case	88%	90%
UML Activity	100%	
UML Extend / Enum Class	61%	84%
Vectors	75%	86%
Instancing	75%	
Op Overload/Streams	23%	
Bonus	75%	67%
Overall	74%	

Statistics and Such

- Only a few questions were left blank
  - 99% still working after 40 minutes (97% last semester)
  - 79% still working after 60 minutes (71% last semester)
  - 34% still working at end (35% last semester)
  - Half of the exams had been returned by 70 minutes (67 minutes last semester)
- Minor typos (e.g., missing ;) were forgiven
- Much leeway was given on the models
  - A *reasonable* representation was sufficient for full credit
  - Rubrics were used to maintain consistency

Some Problem Areas

- Most missed multiple choice options:
  - String is NOT just another name for char*!
  - References can’t change to point to a different variable
  - Makefiles are NOT specific to Linux and don’t execute top to bottom
- Most students did well on modeling and coding
  - Use case and activity diagrams looked great!
  - UML extensions did not look good – same as last time, despite additional emphasis this semester
  - Enum classes and string streams were pretty disastrous

Identifying Your Test

- Three distinct tests were given for op sec
  - The *order* of many questions changed
  - Operational security :-)
- The tests were named “A”, “B”, and “C” based on the 1st letter of the first definition in Section I

	An instance o encapsulated		Code for whicl		Updating code f functionality

Test Markings

- Section I – red “X” marks any errors
- Section II – red “/” indicates incorrect answers, red letters or circles indicate missing answer, +N or N in left column per question indicates points gained for that question
  - Each answer is “true” or “false”. +½ if correct, +0 if not.
  - If more than 4 possible, ignore the rest
- Section III – corrections *may* be marked, +N in left column per question indicates points gained for that question
- Sum of points per page are indicated next to page number
- Final *raw* score will be on page 3 (or 2)
- Adjusted score will only be on Blackboard

Review of the Exam Key

- Correct answers show by variant
  - For sections I and II, order is A / B / C
- Rubric is in red

CSE 1325: Object-Oriented Programming

Agile Processes and Scrum

Mr. George F. Rice

george.rice@uta.edu

Based on material by Bjarne Stroustrup
www.stroustrup.com/Programming

ERB 402
Office Hours:
Tuesday Thursday 11 - 12
Or by appointment

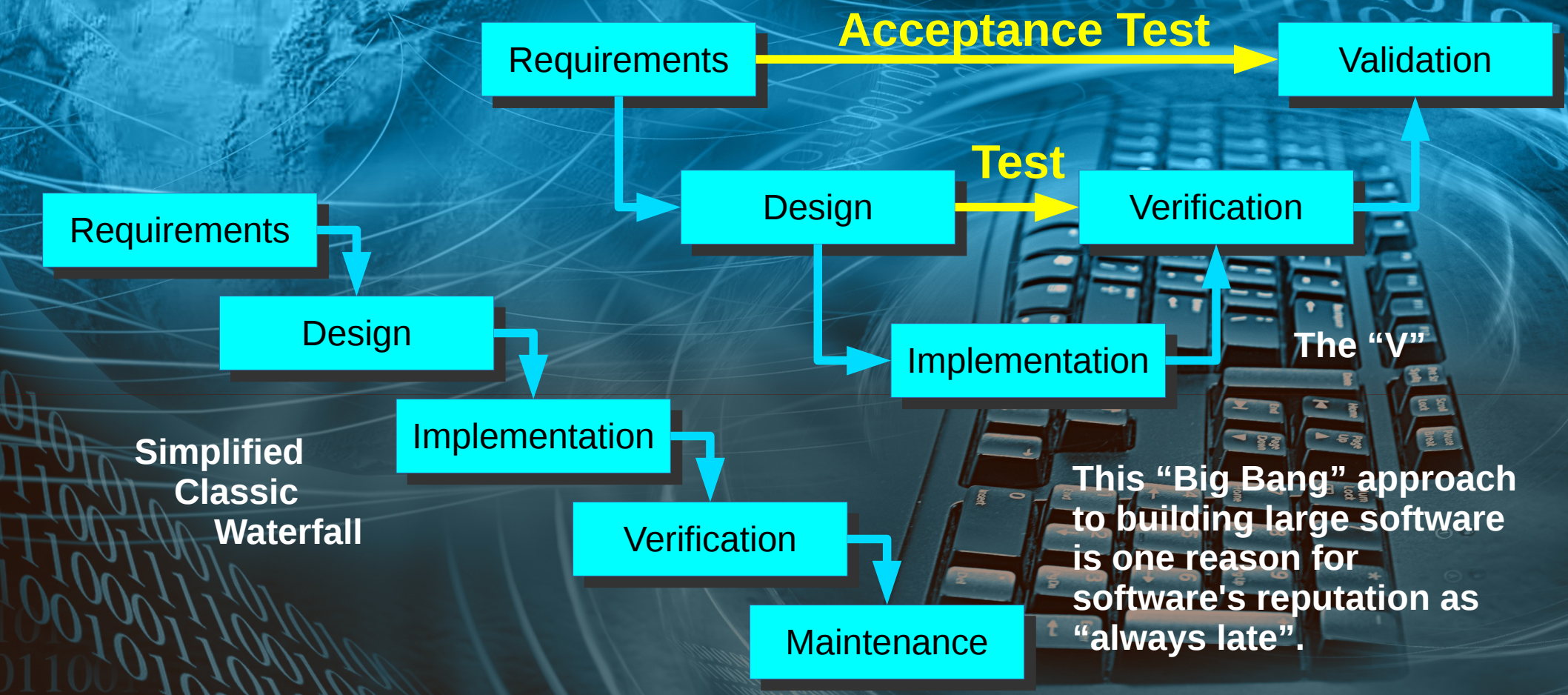
Software Process
How to Manage Large Software Development

- Small programs are easy to build
  - A few classes, some test code, and voilà
- Large programs are hard to build
  - Or are hard to build correctly
  - Or are hard to build on time and within budget
- Large teams are hard to manage
  - Communication paths grow quadratically n(n-1)/2
  - The project vision blurs with distance
- The Second System Effect identifies Version 2.0 as the most dangerous version ever attempted
  - Architects tend to include *every feature* omitted from Version 1.x
  - Duke Nukem Forever, Windows Longhorn, Animusic 3...

99 little bugs in the code.
99 little bugs.
Take one down,
Patch it around.
127 little bugs in the code...

The Waterfall Process

- The Waterfall Process was defined by Winston W. Royce in 1970



Waterfall Issues

- Getting requirements correct is **very** difficult
  - Communication issues / ambiguity
  - The Customer is always right... they just have no idea what they're talking about
- Requirements change throughout the development
  - Freezing requirements at the top of the waterfall **ensures** delivery of a suboptimal system at the bottom
- All requirements are typically completed before delivery
  - But 20% of requirements may deliver 80% of value

The Agile Manifesto

- **Individuals and interactions**
  over processes and tools
- **Working software**
  over comprehensive documentation
- **Customer collaboration**
  over contract negotiation
- **Responding to change**
  over following a plan

Agile can help, but there are no silver bullets.

There is value in the lower items
but greater value in the upper items

Scrum

- Scrum is an agile **process**, not a software development **method**
  - Excels at rapid, flexible response to change in requirements and other unexpected (or expected) challenges
  - Can manage any activity that delivers value
  - For software, often paired with Extreme Programming (which we will NOT cover)
- In use since 1986
  - Named Scrum in 1995
  - First used to design cars, printers, and photocopiers

Scrum Roles

- Product Owner
  - That's your TA and me :-)
- Scrum Master
  - The team's shepherd, champion, facilitator, and (typically) expert in Scrum
  - No teams, so that’s you
- Team Member
  - That's you!

4 Primary Scrum Artifacts
(for modestly sized projects)

- Product Backlog
  - Anything of value to the Product Owner
  - Prioritized by the Product Owner
- Sprint Backlog
  - The team's To Do List for a single sprint
  - Consists of tasks that must be done to clear a backlog item
  - Each task belongs to exactly one person
- Burn Chart
  - Simple line graph of work remaining over time
  - We hope the slope is downward!
- Task Board
  - List of tasks for the sprint arranged in columns – To Do, Doing, Done
  - Provides visibility of activity across the team!

Sprints

- All work is organized into **sprints**
- A sprint is a time-boxed period of work
  - Implements 1 or more Product Backlog items
  - Consists of a number of tasks required to implement those items
- Each sprint ends when scheduled
  - Any incomplete Product Backlog items go back on the backlog for a later sprint
  - The product **must** be in deliverable shape – compiles, passes all tests, needed docs, ready to deliver
  - A demo to the Product Owner follows every sprint

Simple Product Backlog with Burn Chart

Product Name:
Team ID:

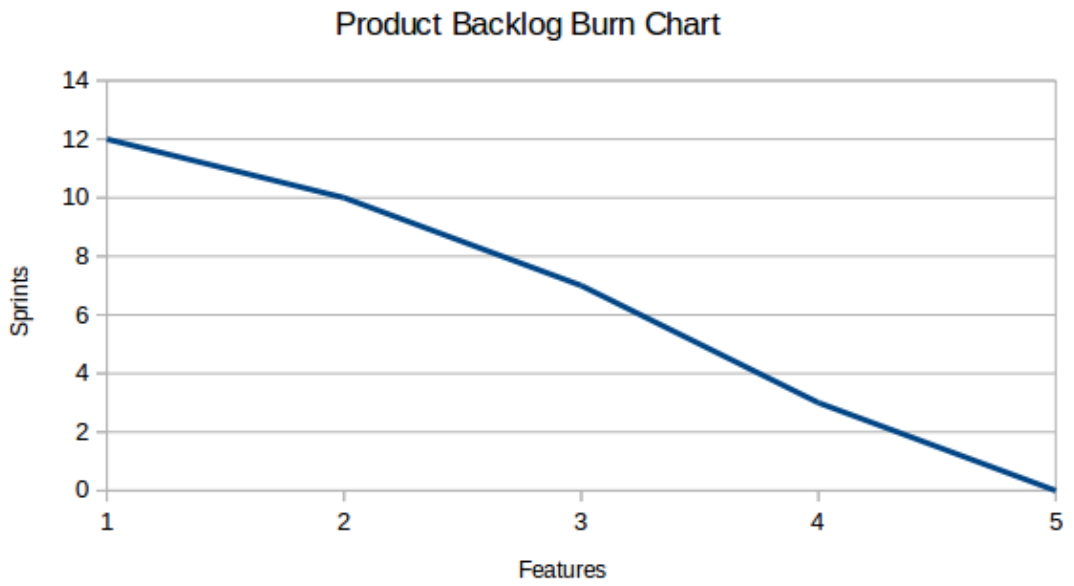
Name:
Initials:
Student ID:

Name:
Initials:
Student ID:

Name:
Initials:
Student ID:

Name:
Initials:
Student ID:

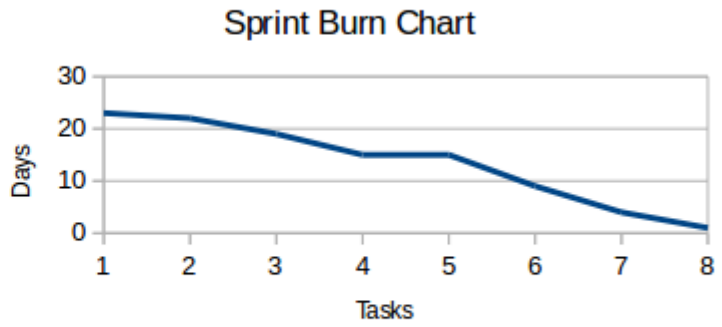
Total Features	**12**
Sprint 1 Left	**10**
Sprint 2 Left	**7**
Sprint 3 Left	**3**
Sprint 4 Left	**0**



Feature ID	Sprint #	Category	As a...	I want to...	So that...	Status	Notes

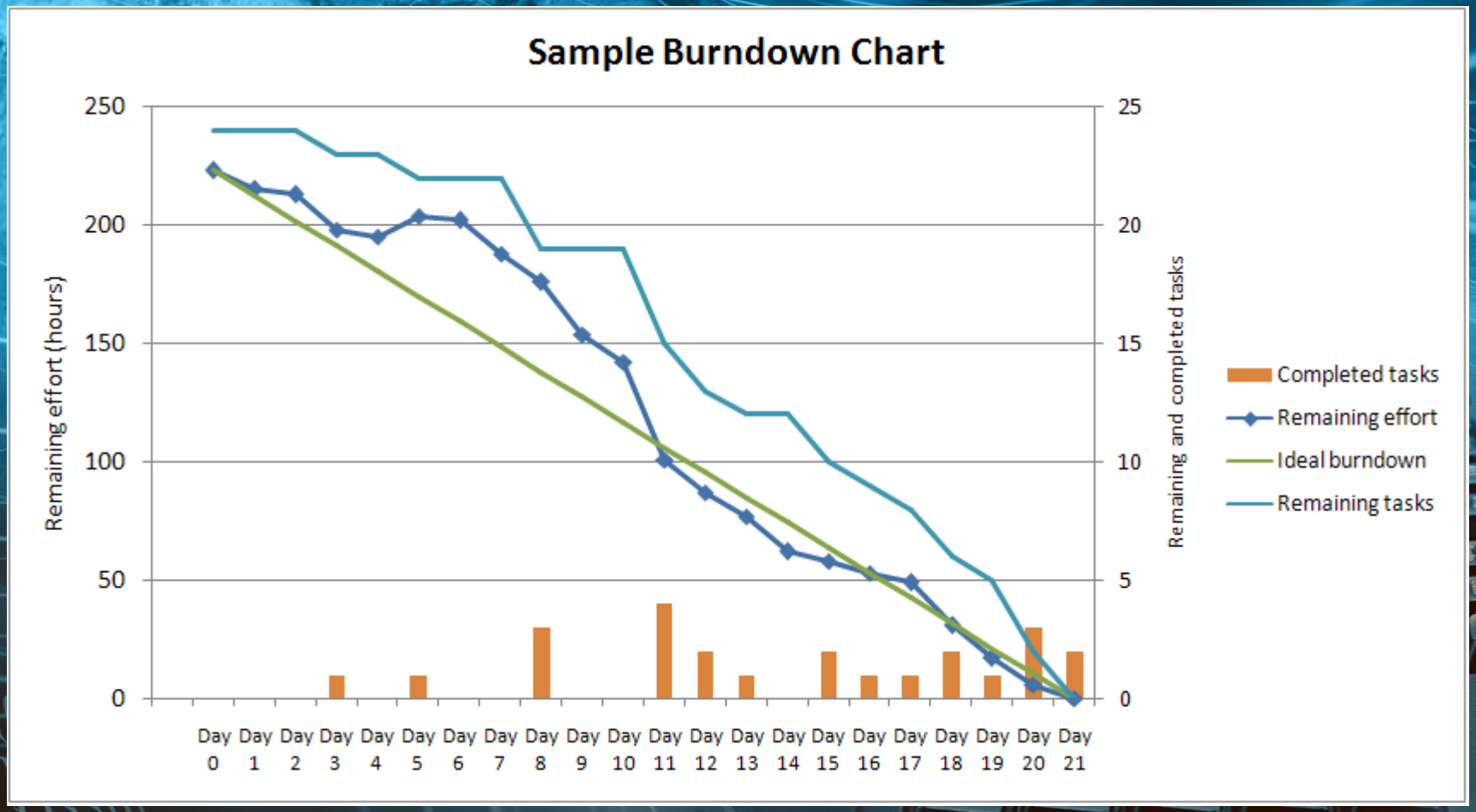
Simple Sprint Backlog with Burn Chart

Sprint #	
Start on	
End on	
Demo on	
Total Tasks	23
Day 1 Left	22
Day 2 Left	19
Day 3 Left	15
Day 4 Left	15
Day 5 Left	9
Day 6 Left	4
Day 7 Left	1



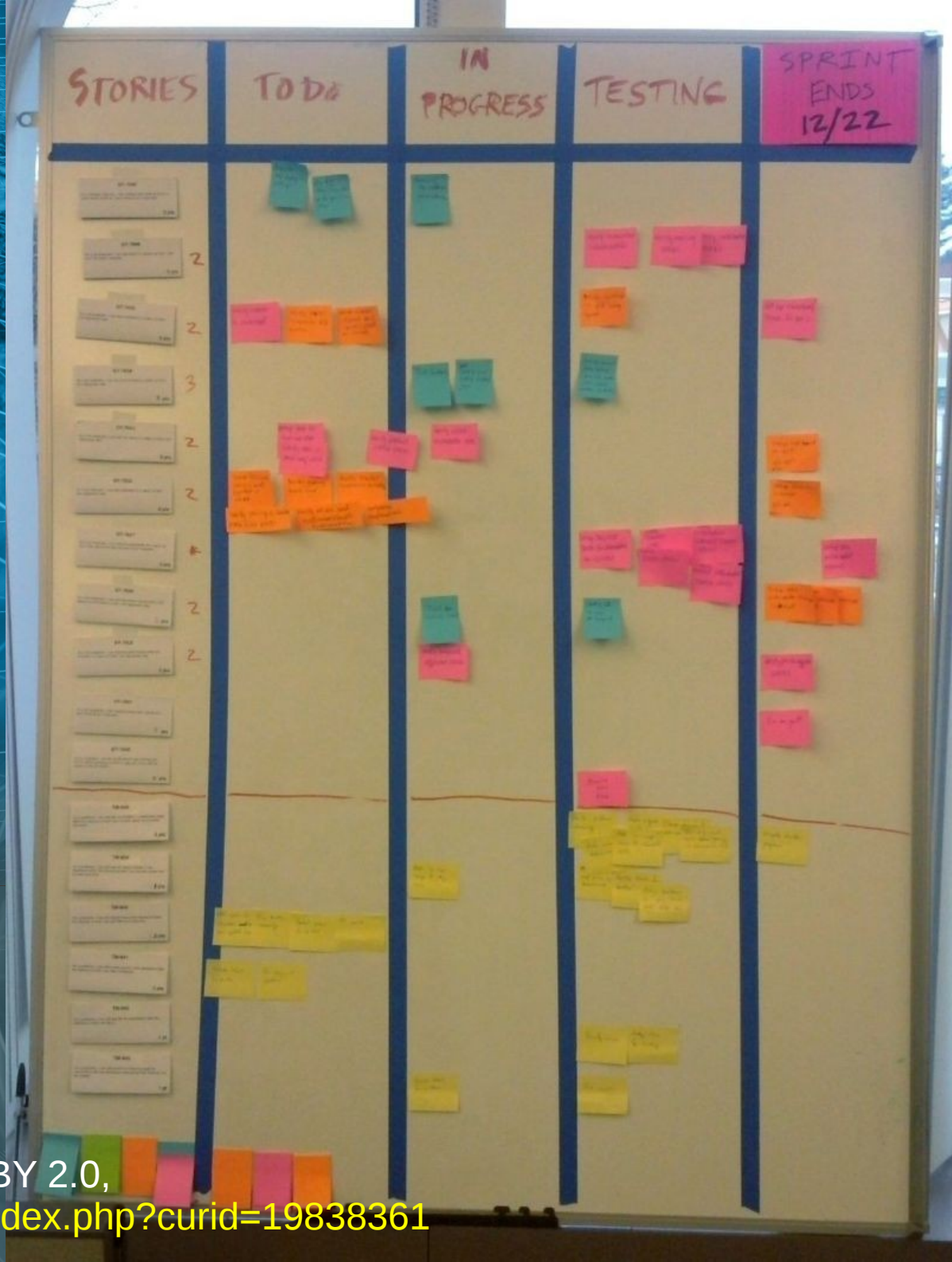
Task ID	Feature ID	Category	Type	Assigned To	Description	Status	Notes

More Complex Sprint Burn Chart



By Pablo Straub - Own work, Public Domain
https://commons.wikimedia.org/w/index.php?curid=7132232

Sample Physical Taskboard



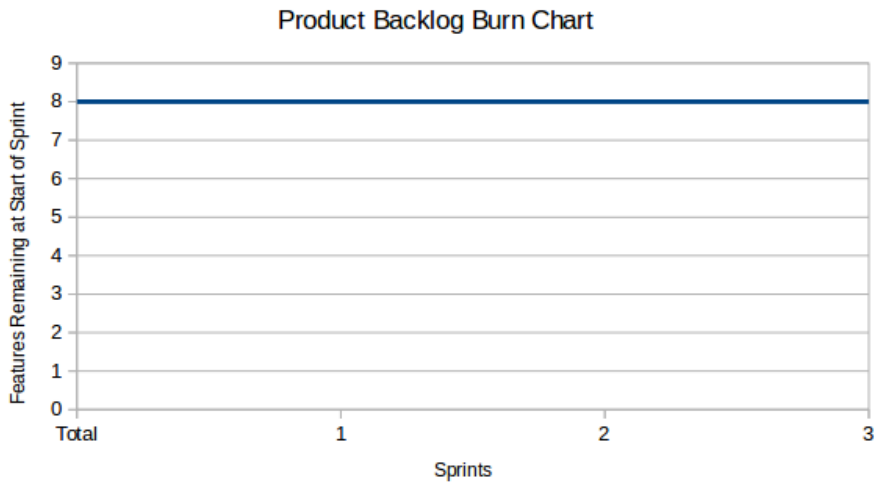
Homework Scrum Spreadsheet
Product Backlog

Product Name: **C1325 Library Management System**
Team ID:

Complete Fields in Green!!!

Name:
Initials:
Student ID:

Teaming is not available for this assignment!
Optional teaming may be available in future assignments



	Remaining	Completed (this sprint)
Total Features	8	
Sprint 1 Left	8	0
Sprint 2 Left	8	0
Sprint 3 Left	8	0

Note: Priority of unfinished Features is subject to change at the end of each sprint at the whim of the Product Owner
Additional features may be proposed by the student but must be approved by the Product Owner in writing

Feature ID	Prio	Status	Planned	Sprints Status	As a...	I want to...	So that...	Notes
AP	1	NS	1		LIB	Add a publication	I can keep track of what is in the library	
LP	2	NS	1		LIB	List all publications	I can find out what is in the library	This will also be needed for the next two features
CO	3	NS	1		LIB	Check out a publication	I can track who has borrowed each publication	Each publication will need to know if it is checked out or not
CI	4	NS	1		LIB	Check in a publication	I can tell when a publication is returned	
HE	5	NS	1		LIB	Get help	I can learn how to use the system	
PA	6	NS	1		LIB	Add a patron	I can conveniently keep track of all patrons	This is the Bonus level
SP	7	NS	1		LIB	Select a patron when checking out a pub	I don't have to rekey recurring patron info	This is the Bonus level
CU	8	NS	1		LIB	Add custom info for each publication type	I can keep better track of library assets	This is the Extreme Bonus level

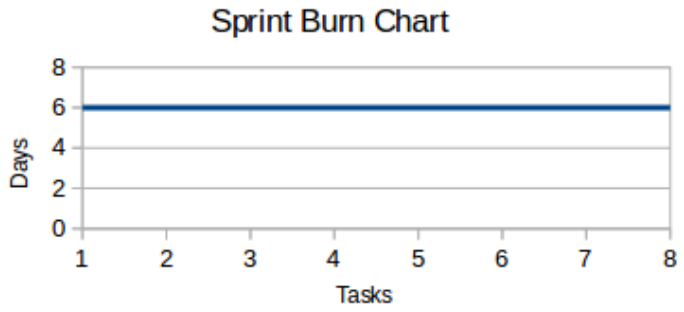
Homework Scrum Spreadsheet
Sprint Backlog

Sprint #	1
Start on	Sep 21
End on	Sep 28
Demo on	TBD

Complete Fields in Green!!!

***** Create a new Sprint Backlog for EVERY SPRINT *****

	Remaining	Completed (this day)
Total Tasks	6	
Day 1 Left	6	0
Day 2 Left	6	0
Day 3 Left	6	0
Day 4 Left	6	0
Day 5 Left	6	0
Day 6 Left	6	0
Day 7 Left	6	0



Feature ID	Description	Status	Notes
AP	Create a Publication class with data and a constructor		Included a WRITTEN test with each constructor and method!
AP	Add the to_string method to Publication		
AP	Add the is_checked_out method to Publication		
AP	Add the check_out method to Publication		
AP	Add the check_in method to Publication		
LP	--> Add additional tasks to complete the homework		
CO			
CI			
HE			
PA			
SP			
CU			

Multi-Week Projects

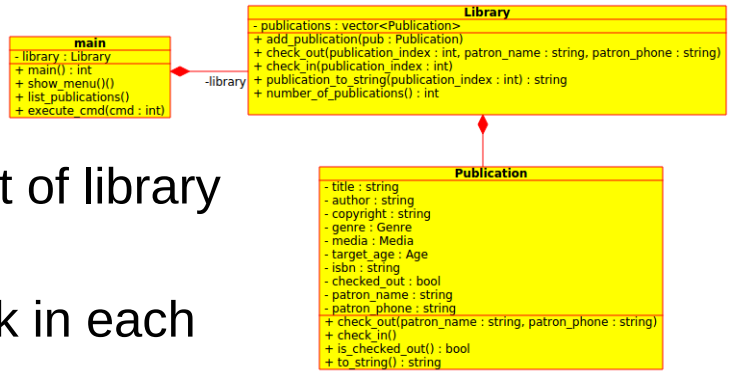
- Work individually*
- Work will be managed by super-simplified Scrum
  - Use git version management as always
  - A Product Backlog and Sprint Backlog will be maintained with automatically generated burn charts (delivered after each sprint)
  - Your sprint backlog will be pre-defined but is negotiable (especially on later sprints)
  - A *possible* demo to the class after the last sprint
    - To be scheduled as time permits

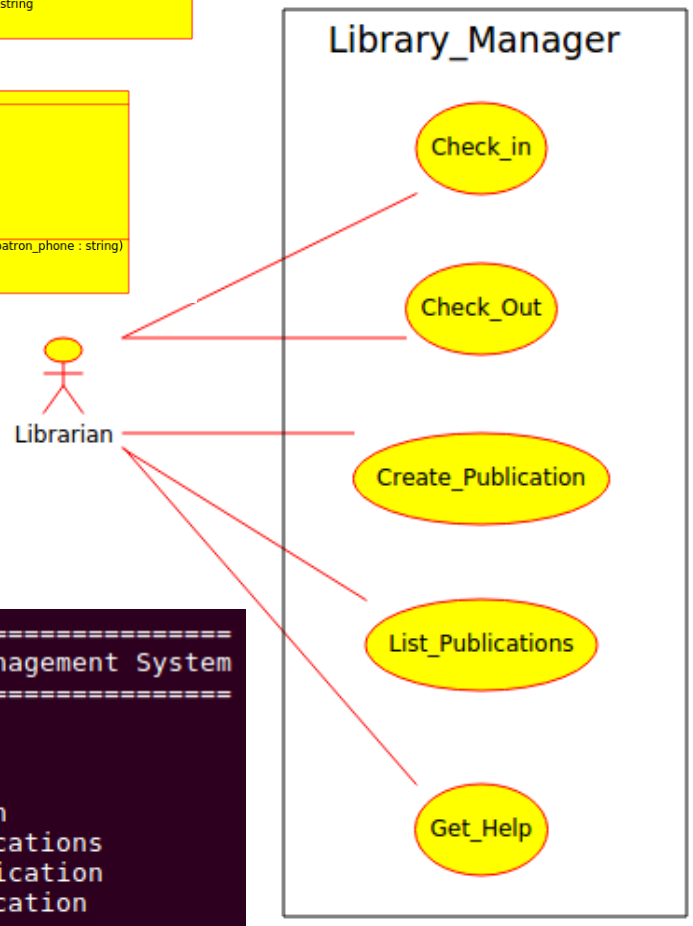
* Team development *may* become available later, **NOT for Homework #5**

Homework #5

Sprint 1 of 3

- Build a simple Library Management System
  - Add publications to the list of library assets
  - Check out and check back in each publication
  - Provide basic help
- As always, use git for version management
- Manage the sprint with the Scrum spreadsheet
- Details are on Blackboard
- **Due March 1 at 8 am**





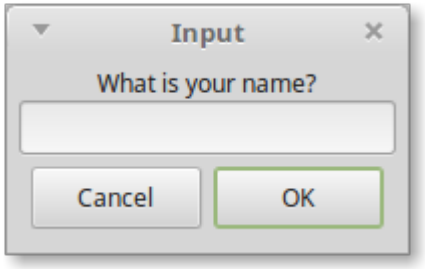

```
==================================
CSE1325 Library Management System
==================================

Publications
------------
(1) Add publication
(2) List all publications
(3) Check out publication
(4) Check in publication

Utility
-------
(9) Help
(0) Exit

Command?
```

Sprints

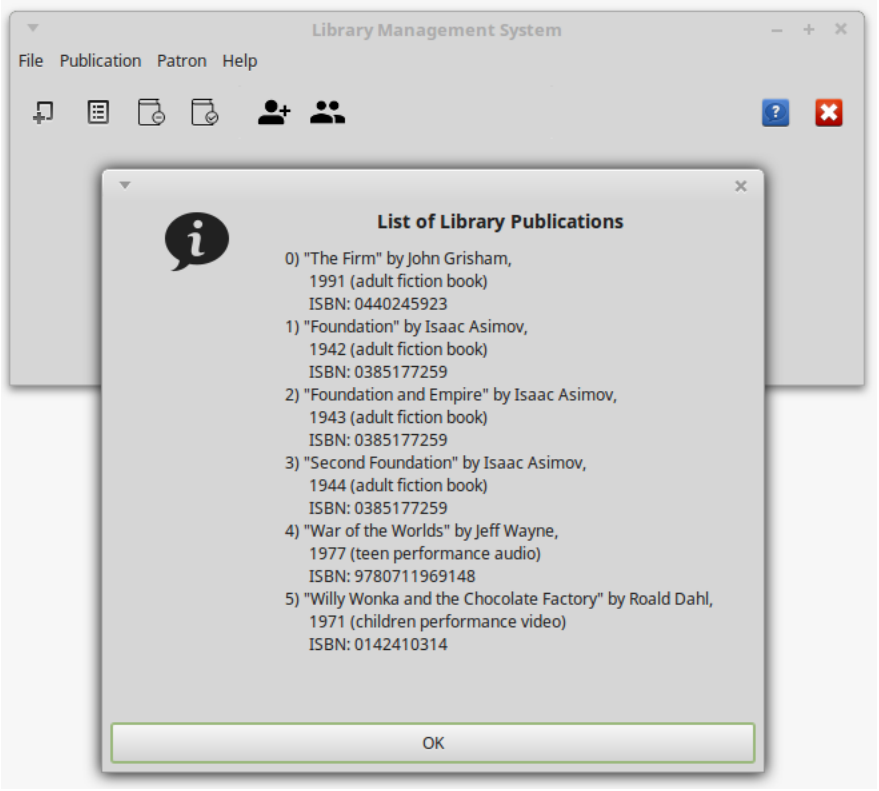
Feature ID	Priority	Planned	Status	As a...	I want to...
AP	1	1		Librarian	Add a publication
LP	2	1		Librarian	List all publications
CO	3	1		Librarian	Check out a publication
CI	4	1		Librarian	Check in a publication
HE	5	1		Librarian	Get help

Previews of Sprints #2 and #3
(Also known as Homework #6 and #7)

- Sprint #2
  - Add *dialogs*



- Sprint #3
  - Add a *main window*



For Next Class

- We begin GUIs in gtkmm!
  - You'll need the gtkmm libraries in your environment (they are pre-installed in the VM!)