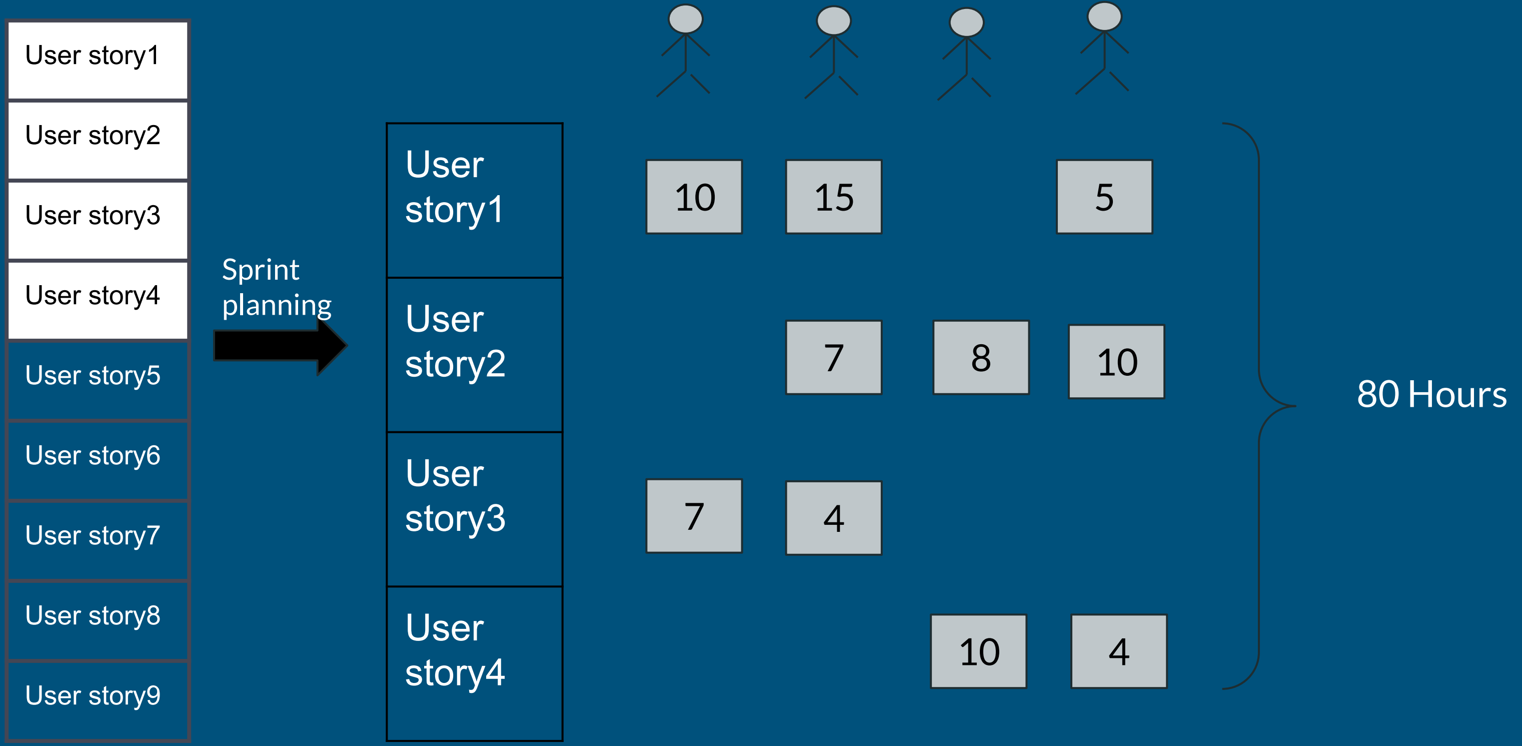
Burndown charts & Testing

Manas Jyoti Das, PhD
Computer Science

Burndown charts

- It is a visual tool commonly used in Agile and Scrum project management to track the progress of work in a sprint or over a project's timeline
- Representation of how work is progressing and whether the project is on track to meet its goals
- The chart can represent any one of the following:
  - Task burndown
  - Release burndown
  - Feature burndown
  - Sprint burndown
- Sprint burndown: It represents remaining amount of work for any specific sprint

Burndown chart

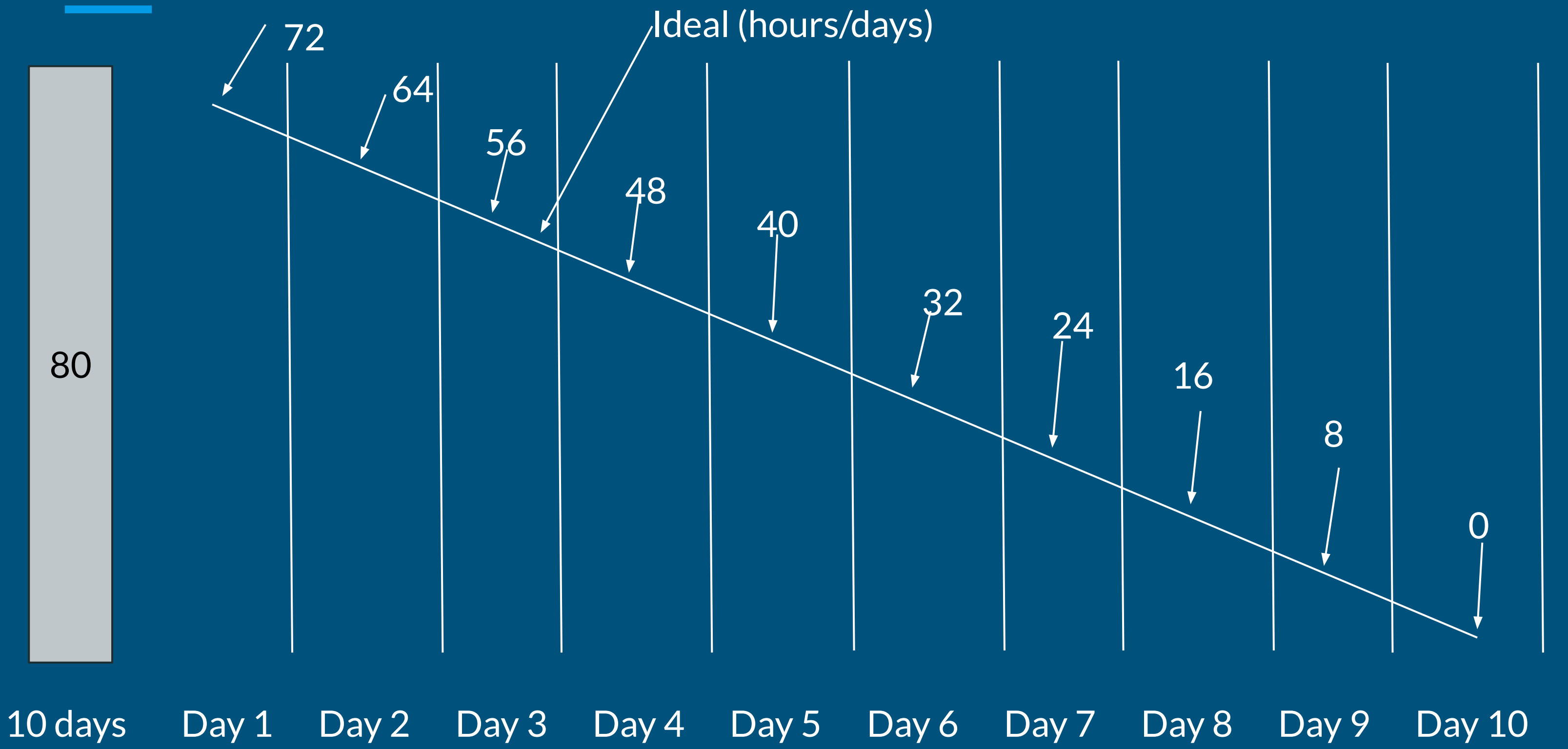


product backlogs

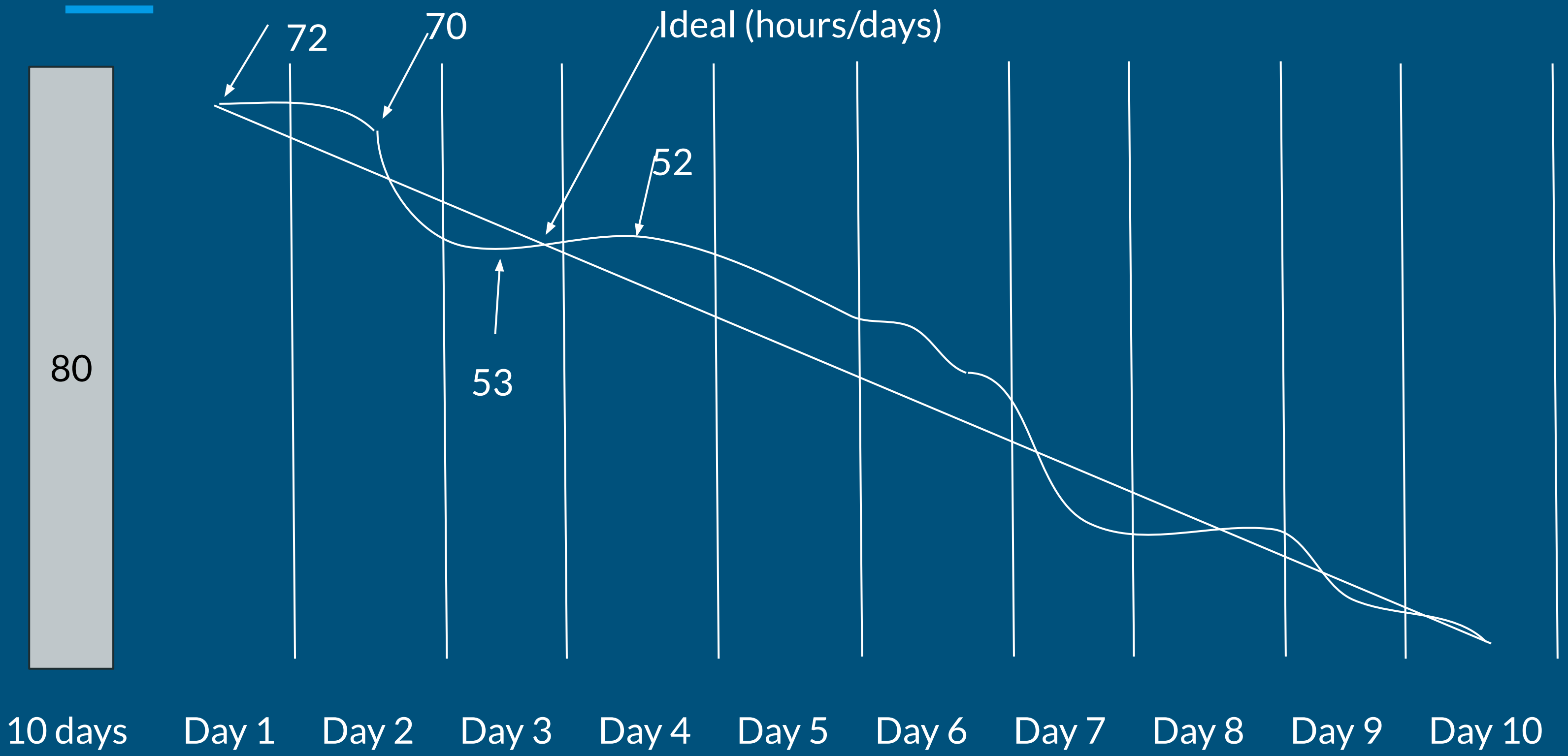
Burndown charts

- Assume sprint is for 10 days
- Ideal scenario: hours/days (80/10=8)
- Next : 80-8 = 72
- Next: 72-8 = 64

Burndown charts



Burndown charts



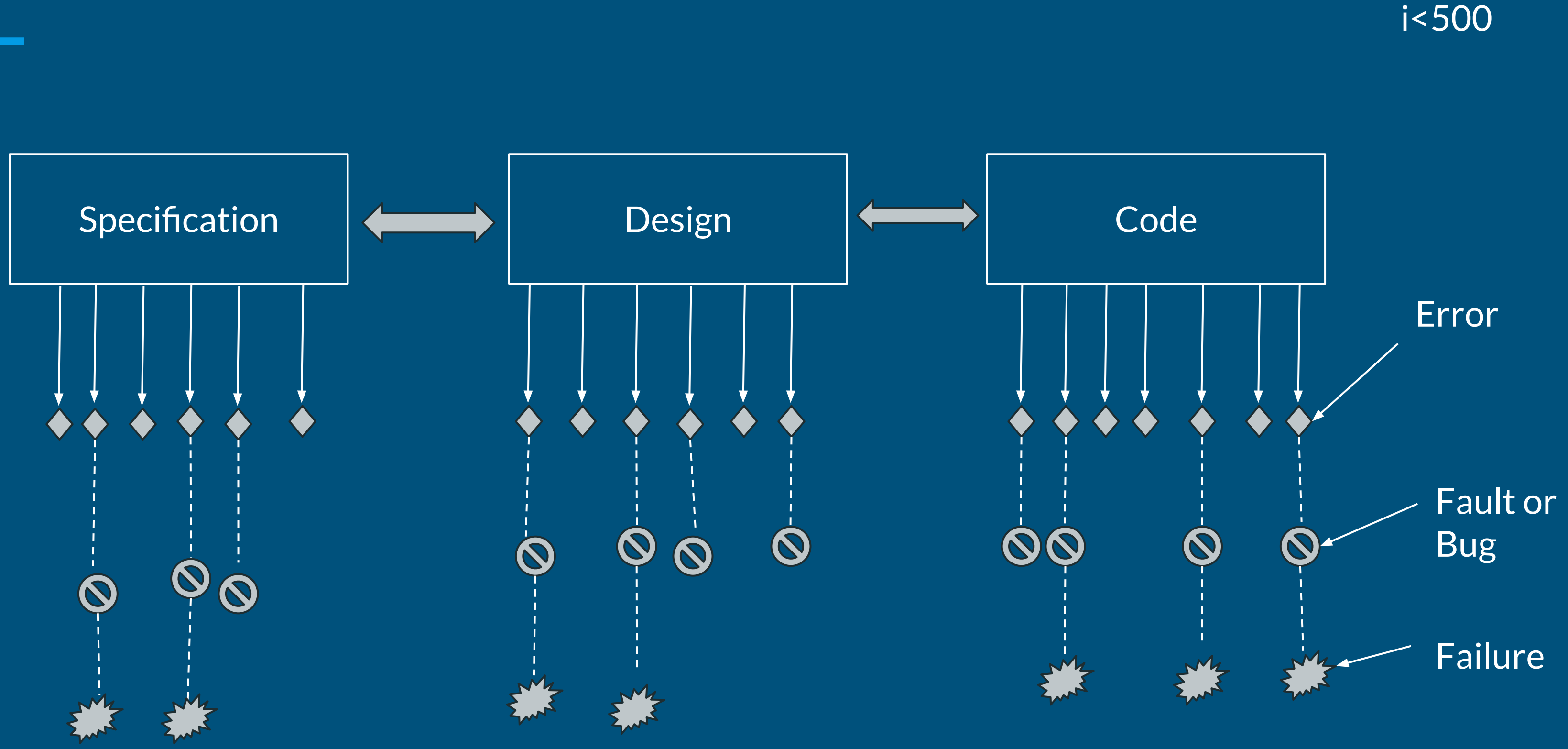
Benefits of Burndown charts

- Visibility: They provide a clear visual representation of progress, allowing everyone involved in the project to stay informed.
- Proactive Management: Deviations from the ideal line can indicate potential roadblocks and enable adjustments to be made early on.
- Team Communication: Burndown charts can facilitate discussions within the team about workload and identify areas where collaboration or additional resources might be needed

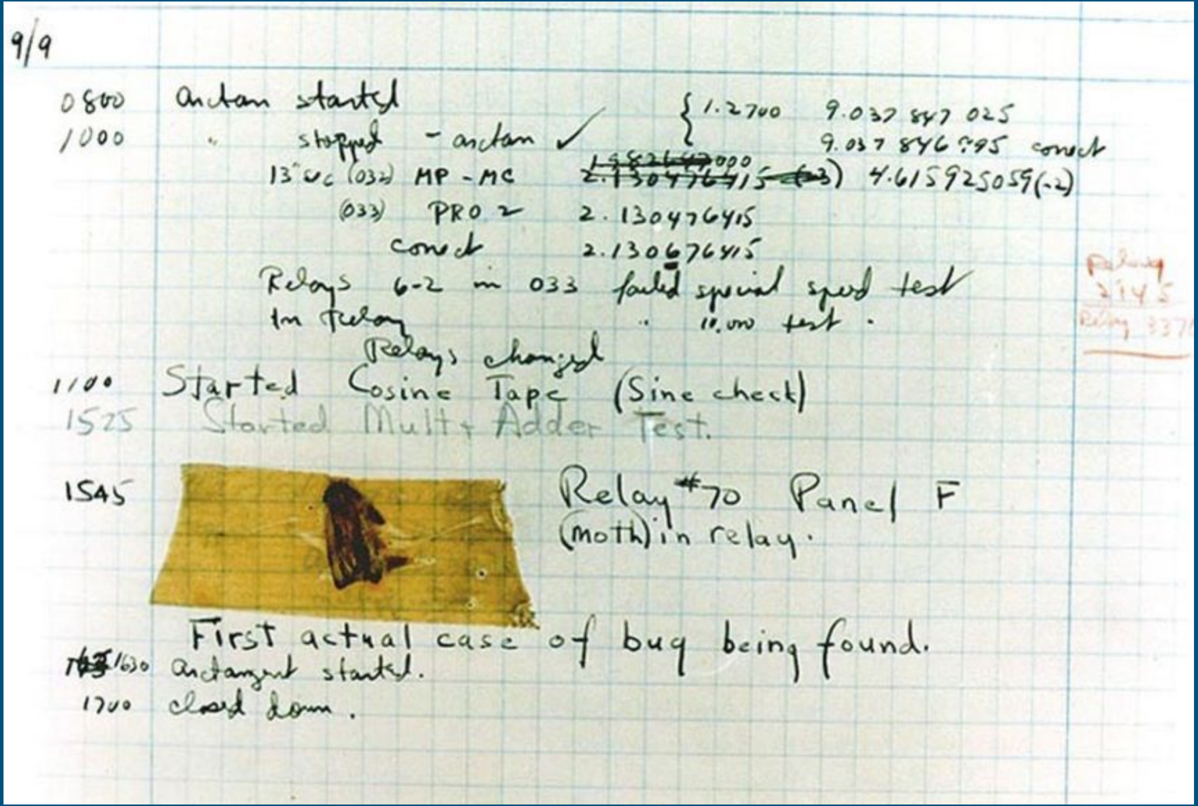
What to test?

- If there is any error or fault in the system.
- Difference between error and fault?
- IEEE 1044:
  - Errors: These are found in the code
  - Faults: these happens due to errors in the code that emerges into fault or failure of the system

Errors, faults and bugs



A bug's life



9/9

0800 Arctan started

1000 " stopped - arctan ✓ {1.2700 9.037 847 025 / 9.037 846 995 correct

13" uc (032) MP - MC ~~1.982147000~~ ~~2.130476415 (-3)~~ 4.615925059(-2)

(033) PRO 2 2.130476415

correct 2.130676415

Relays 6-2 in 033 failed special speed test

In relay " 10,000 test.

Relays changed

Relay 2145

Relay 3376

1100 Started Cosine Tape (Sine check)

1525 Started Mult+ Adder Test.

1545 Relay #70 Panel F (moth) in relay.

First actual case of bug being found.

~~1525~~ 1630 Arctangent started.

1700 closed down.

How to reduce bug?

- Two most important ways to reduce the number of bugs
- Code review
- Testing

Testing

- Monkey testing: Giving random numbers as an input. Now a days not much of use.
    - Dumb Monkey: Test with random values and random workflow
    - Smart Monkey: Knows what the system is doing
- The problem with this type of approach is that
    - Many program parts may not get tested
    - Risky areas of a program may not get tested
    - The tester may not be able to reproduce the failure
- Now a days testing activity is spread over entire life cycle

Verification and validation

Verification

- Review
- Simulation
- Unit testing: Done on a single component/module of the system
- Integration testing: Done on a few components/module of the system
- System testing : Done on fully developed system

Validation

Verification and validation

Verification	Validation
Are you building it right?	Have you built the right thing?
Done by developers	Done by testers
Static and dynamic activities: reviews, unit testing	Dynamic testing: Execute software and check against requirements

Testing is done in four levels

- Unit testing : each module is tested as a standalone module.
- Integration testing: Some of the modules are integrated. Not integrated randomly, there is an integration plan
- System testing: All the modules are integrated
- Regression testing: Testing done at the time of maintenance

- Why do unit testing? Why not integrate every module and do the testing?
    - It would be difficult to determine which module has error

Levels of testing

What user really need	Acceptance testing
Requirement	System testing
Design	Integration testing
Code	Unit testing
Maintenance	Regression testing

System testing

Thank You

See you all in the next class