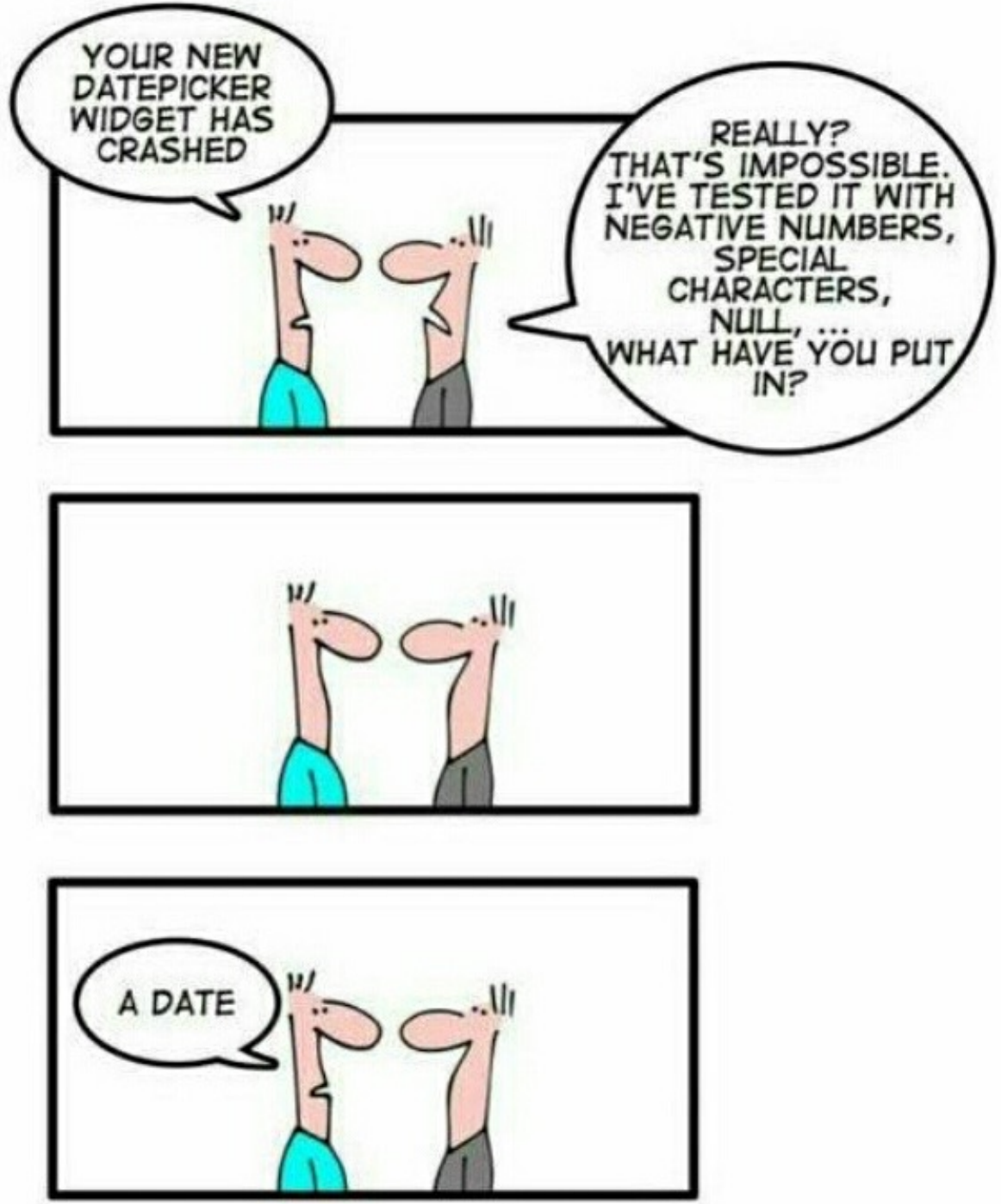

YOUR NEW DATEPICKER WIDGET HAS CRASHED
REALLY? THAT'S IMPOSSIBLE. I'VE TESTED IT WITH NEGATIVE NUMBERS, SPECIAL CHARACTERS, NULL, ... WHAT HAVE YOU PUT IN?
A DATE

Welcome to Software Development

CSCI 2134: Software Development

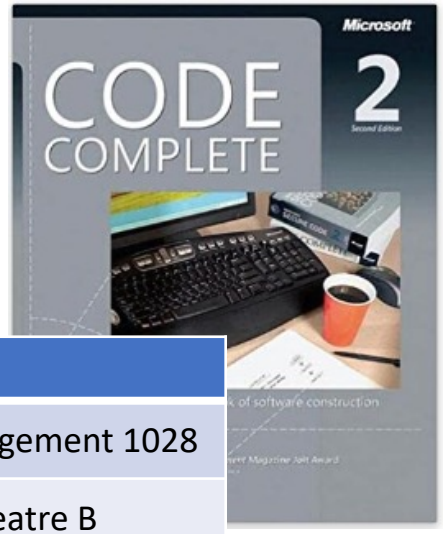
Agenda

- Announcements
  - Check your Dal email and course team regularly
  - Check your Brightspace account
  - Complete the Brightspace course orientation module "Welcome to the Course".
- Lecture Contents
  - Administrivia
  - The Process of Software Development
- Readings:
  - This Lecture: Chapter 1
  - Next Lecture: Chapter 30

Course Description

- Overview and the Software Development Life Cycle
- Tools of the Trade
- Code Quality
- Testing
- Debugging and Fixing
- Defensive Programming
- Software Specification
- Software Design
- Software Architecture

Administrivia



Who/Where/When

- Instructors: Alex Brandt
- Pre-reqs: CSCI 1101 or CSCI 1110

Contacts:

- Email:
  - Csci2134@dal.ca
- Website: http://dal.brightspace.com
- Office hours: TBA (see Brightspace)

Course Resources:

- Text: S. McConnell, “Code Complete 2”, Microsoft Press, 2004.

Lectures	Room
TR 16:05-17:25	Studley ROWE Management 1028
MW 14:35-16:55	Carleton TUPPER Theatre B

Labs	Room
B01: W 10:05-11:25	Studley MONA CAMPBELL 1201
B02: F 10:05-11:25	Studley MONA CAMPBELL 1201
B03: T 08:35-09:55	Studley GOLDBERG 143
B04: M 10:05-11:25	Studley MONA CAMPBELL 1201
B05: R 11:30-12:55	Studley GOLDBERG 134
B06: M 11:35-12:55	Studley GOLDBERG 143
B07: W 08:35-09:55	Studley MONA CAMPBELL 1201

What to Expect

- Please follow public health guidelines
- Lectures and labs will be in-person, depending on public health guidelines
- You are expected to attend your scheduled lab section
- Pre-recorded lectures from a previous term may be posted on Brightspace. These are intended for your reference, in case of short term illness, or for students arriving a few weeks late to campus, but do not replace in person instruction.

Course Assessment

- Participation and Quizzes (10%) via Brightspace
- Assignments (20%): worth 2%, 6%, 6%, and 6%
- Weekly Labs (20%): 8 in total. (Lab 0 is not worth marks)
- Two midterms: (10% each). Lowest replaced by final exam if higher
  - When: Oct 16/17 and Nov 20/21 during scheduled class times
- Final exam: (30% or 40%)
  - Final Exam: Scheduled during weeks of Dec 8-19

We will choose the marking scheme that best benefits each student.

Microsoft Teams

- We will be using Microsoft Teams for optional online discussion or questions
- You may :
  - Download Microsoft Teams
  - Join the course team with the join code **`ydv1i85`**
  - Say hi in the General channel
- See https://dalu.sharepoint.com/sites/its/SitePages/teams.aspx if you need more information about how to download and use Teams

Visit the
Dalhousie University Bookstore
for your Computer Science
course materials

bookstore.dal.ca/cs

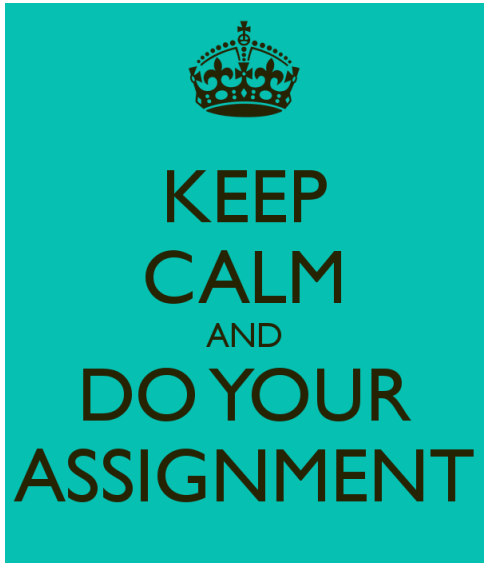
DALHOUSIE
BOOKSTORE
SUPPLIES & STUFF

Labs

Labs	Room
B01: W 10:05-11:25	Studley MONA CAMPBELL 1201
B02: F 10:05-11:25	Studley MONA CAMPBELL 1201
B03: T 08:35-09:55	Studley GOLDBERG 143
B04: M 10:05-11:25	Studley MONA CAMPBELL 1201
B05: R 11:30-12:55	Studley GOLDBERG 134
B06: M 11:35-12:55	Studley GOLDBERG 143
B07: W 08:35-09:55	Studley MONA CAMPBELL 1201

- Labs take place every week, starting next week
  - Lab 0 (Sept 11-15) is to help with setting up your laptop
- Lab instructions will be available 1 week before
- **Students are expected to download and run software during labs on their computer**
- Students should prepare for the lab by reading it over and completing all preparation steps
- **Students should arrive at the lab prepared**
- Some labs may be done in groups. If you cannot attend your scheduled lab then please post a comment in the Teams channel for labs asking for lab partners.
- All submissions will be done via Git.
- Each Lab is due **the first Sunday (11:59pm) after the lab**
- **No late submissions will be accepted** (except by SDA; read the syllabus!)

Assignments



- Due at 11:59pm on the following (tentative) due dates:

Date Due	Task	Value
September 24	Setting up your dev. tools	2%
October 15	Writing test cases	6%
November 12	Debugging, fixing, and testing	6%
December 3	Extending a piece of software	6%

- Assignments posted ~2 weeks before due date.
- **No late submissions accepted**
- Implemented using the Java programming language
- Submitted via Git

Quizzes

- There is one quiz for every lecture.
- Each quiz is due one week after the associated lecture at 23:59.
  - Ex: For a lecture on Wednesday, September 21, it is due Wednesday, September 28, at 23:59.
- Holidays and weekends do **not** extend the deadline.
- Remember there are two lecture sections! That means two different deadlines for the same quiz! When in doubt, whatever Brightspace says as the due **is the due date**.

What is Expected of Me?

- Learn in a way that works for you
- Make use of course resources
- Do the labs
- Start early on assignments
- **Ask for help** if you get stuck
- **Come to class**; take notes in lectures

This course is as much about doing as understanding

- **This is not intended to be a "hard" course.**
- **If you do the work, you will do well.**

Succeeding in the course

- A few suggestions for getting the most out of this course:
  - Labs should only take ~1-1.5 hours. Do them during your scheduled lab time or set aside a 1.5 hour block of time at the same time every week.
  - Think of the assignments like small projects. Expect to work on Assignments 2-4 for an hour or two each week. Start early and ask questions in class!
  - Complete each quiz immediately after class (or at least on the same day)
  - Seek help early if you are struggling:
    - Specific questions: ask on the course's MS Team
    - General help: attend TA or instructor office hours
    - Lab help: attend your lab or ask in the dedicated lab discussion channel
    - Assignment help: ask in the dedicated assignment discussion channel
    - Private help, unrelated to specific course content: email the instructor at csci2134@dal.ca

What Should I Know?

- The Java programming language
- Problem solving and problem decomposition
- Some experience with using an IDE

What Do I Need to Do Now?

- Join the course Team [Recommended]
- Get course textbook. [Recommended]
- Install the required tools on your laptop. [Lab 0/Assignment 1]
- Check Brightspace regularly. [Required]
- **Check your Dal email every day. [Required]**
- **Check Teams regularly and ask any questions there**
- Start Assignment 1 [Strongly Recommended]
- Review Chapter 30 of the text. [Recommended]

Course Representative

Our Course Representative is TBA.

The Course Representative is a point of contact to facilitate and provide more timely feedback mechanisms to instructors and to the Faculty of Computer Science.

Additionally, Course Representatives can assist peers in navigating to the most appropriate support mechanism on campus. You can think of a CR as ‘the middle person’; a neutral point of contact for students to use when they don’t feel comfortable addressing an issue with the professor directly.

Academic Integrity

- Academic integrity means being honest in the fulfillment of your academic responsibilities thus establishing mutual trust.
- Violations of intellectual honesty are offensive to the entire academic community, not just to the individual faculty member and students in whose class an offense occurs.
  - E.g., cheating on tests, plagiarism, falsification of experimental data, etc.
- All cases of academic misconduct are automatically referred to the Faculty Academic Integrity Officer.

Moss: Software Similarity Detection Software
https://theory.stanford.edu/~aiken/moss/

All submitted code will be passed through Moss which performs a pair-wise comparison of similarities

Moss Results

Tue Sep 8 23:29:31 PDT 2015

Options -l python -d -m 10

[How to Read the Results | Tips | FAQ | Contact | Submission Scripts | Credits]

File 1	File 2	Lines Matched
/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (99%)	/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (99%)	86
/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (76%)	/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (66%)	91
/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (81%)	/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (82%)	69
/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (70%)	/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (61%)	70
/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (69%)	/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (40%)	71
/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (56%)	/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (50%)	43
/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (62%)	/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (55%)	67
/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (55%)	/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (48%)	40
/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (54%)	/home/ubuntu/Projects/work/2015/uct-csc1010h/tutorials/6/raw/[redacted]/ (55%)	40

If a student does not wish their assignments to be submitted to Moss, they should contact the instructor.

Never …

- Never email/send/copy your work to another student before the due date
- **Never use homework sites such as Chegg or Course Hero**
  - If you are using the site, chances are another student is also …
- Never copy and paste someone else's code into your own assignment
- Never type code from another source into your own assignment
- Never write code while discussing the problem with other students

It’s OK to …

- Discuss an assignment with another student or TA
- Ask for help to debug your code (but don’t expect someone else to do the work)
- Explain to someone else how to solve a problem
- Use a whiteboard/blackboard/scrap paper to work out the problem, as long as you erase/destroy the board/paper after the discussion
- Look at other people’s code, BUT take a brief break before writing your own

Culture of Respect

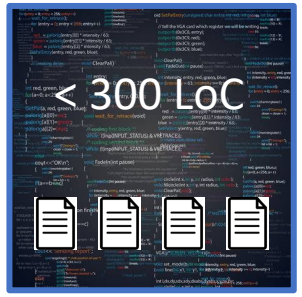
- Every person has a right to respect and safety.
- Inclusiveness is fundamental to education and learning.
- Misogyny and other disrespectful behaviour in our classrooms, on our campus, on social media, and in our community is unacceptable.

Why Are We Here?

- To become better software developers
- To understand
  - What tasks we need to do
  - What tools we need to use
  - What processes we need to follow
  - What milestones we need to meet
  - What deliverables we need to deliver

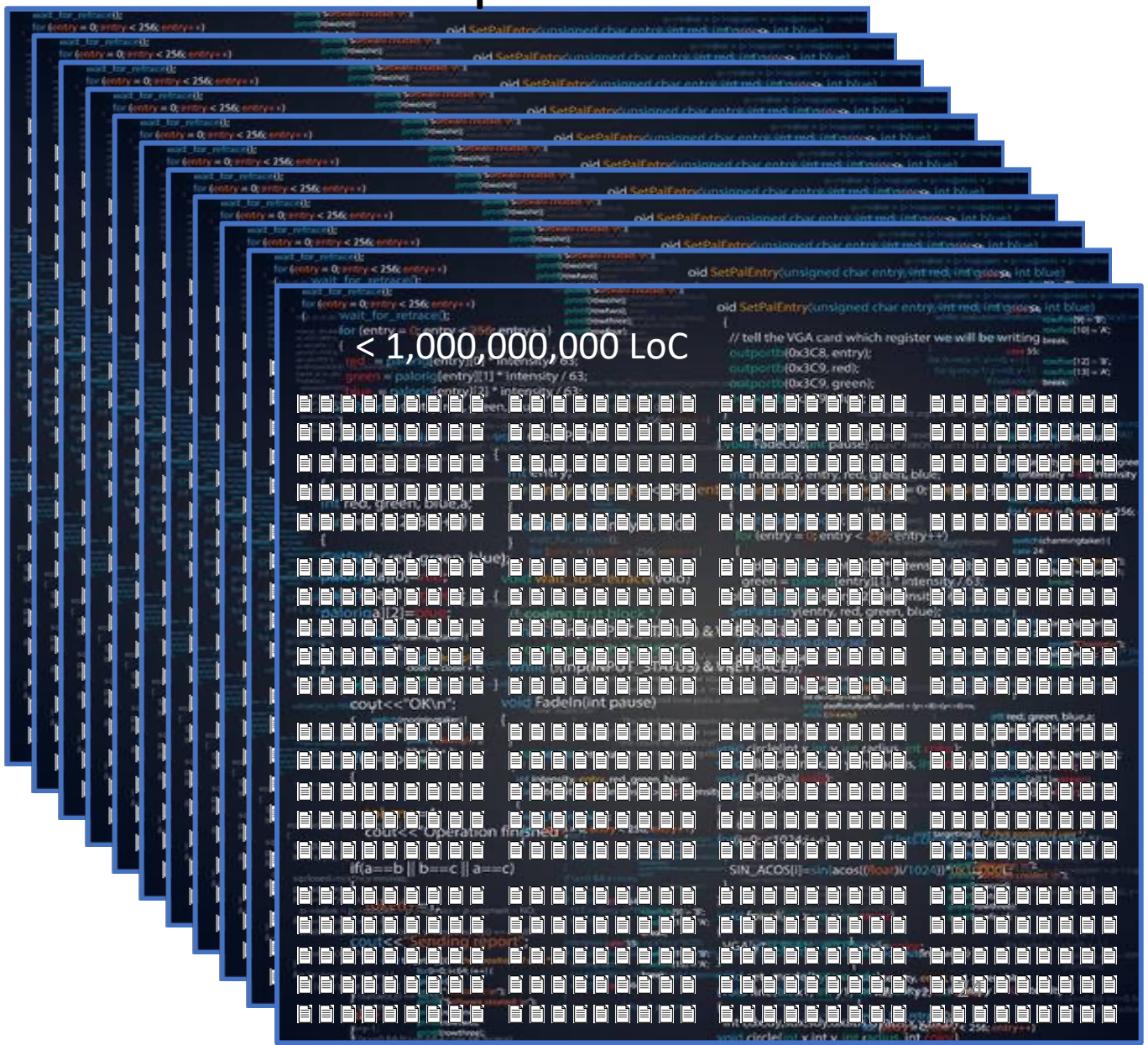
Programming != Software Development

Small Programs



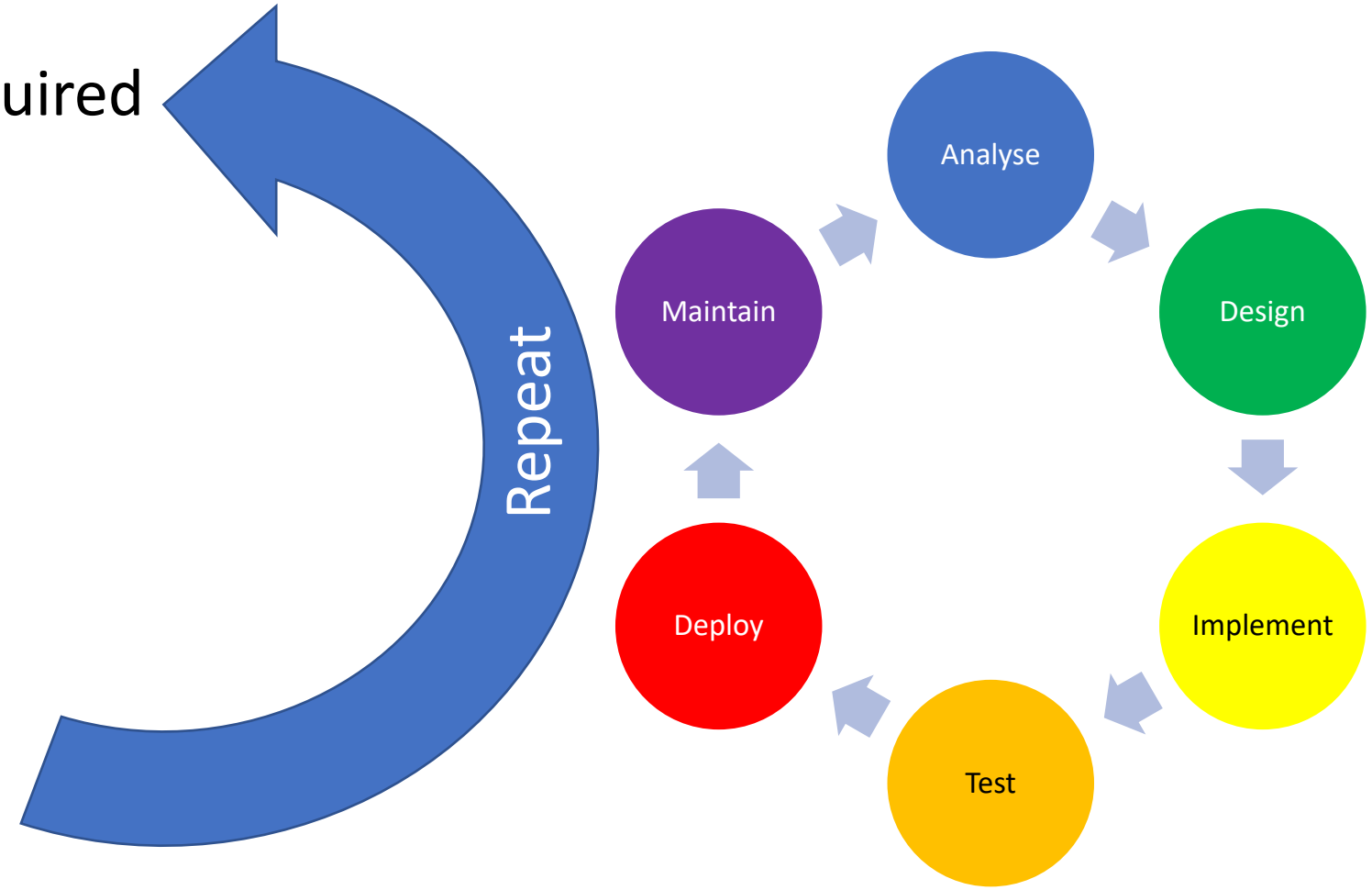
Industry Software Projects

Individual Software Projects



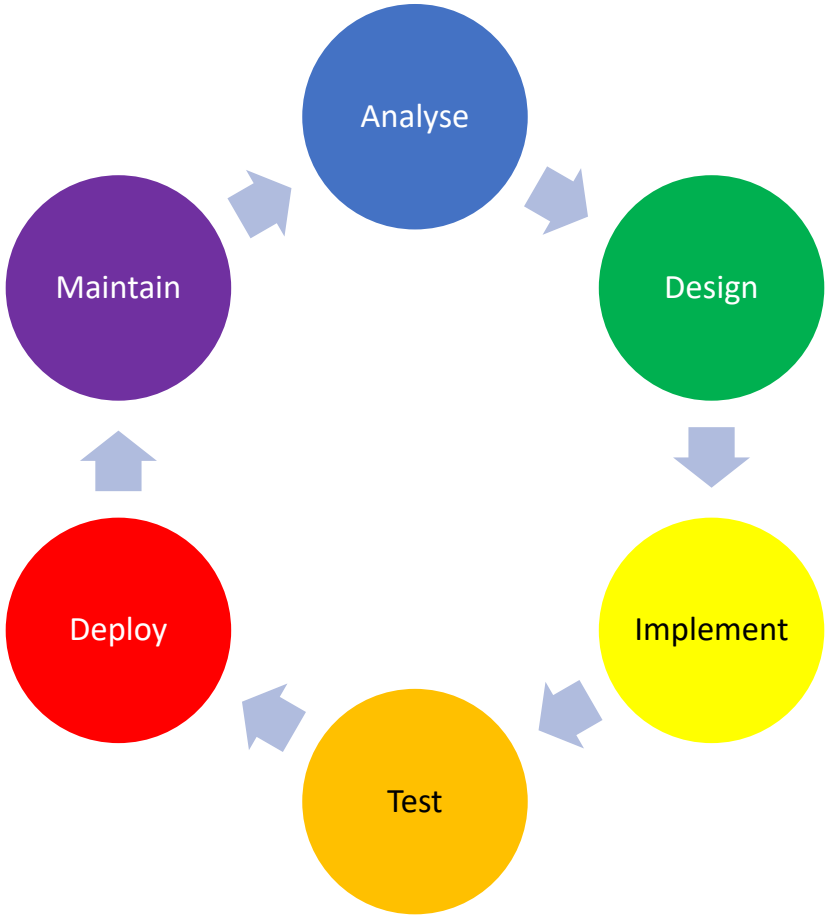
What do you need to do to develop software?

- Determine what is required
- Design the software
- Write the software
- Test the software
- Debug the software
- Deploy the software
- Maintain the software
- Upgrade the software



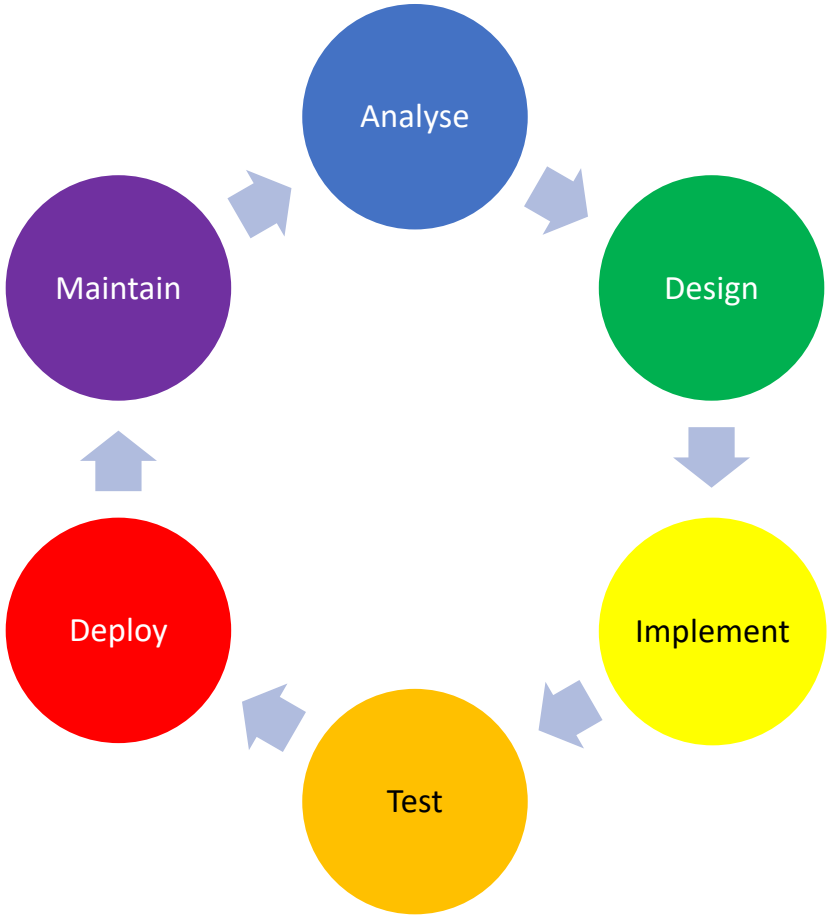
The Software Development Life Cycle

- Requirements gathering
- Design
- Implementation
- Quality assurance
- Deployment
- Maintenance



So far …

- We have asked you to write small programs
  - A little design
  - Mostly implementation
  - Very little testing
- No analysis, deployment, or maintenance



A Different Focus in Software Development

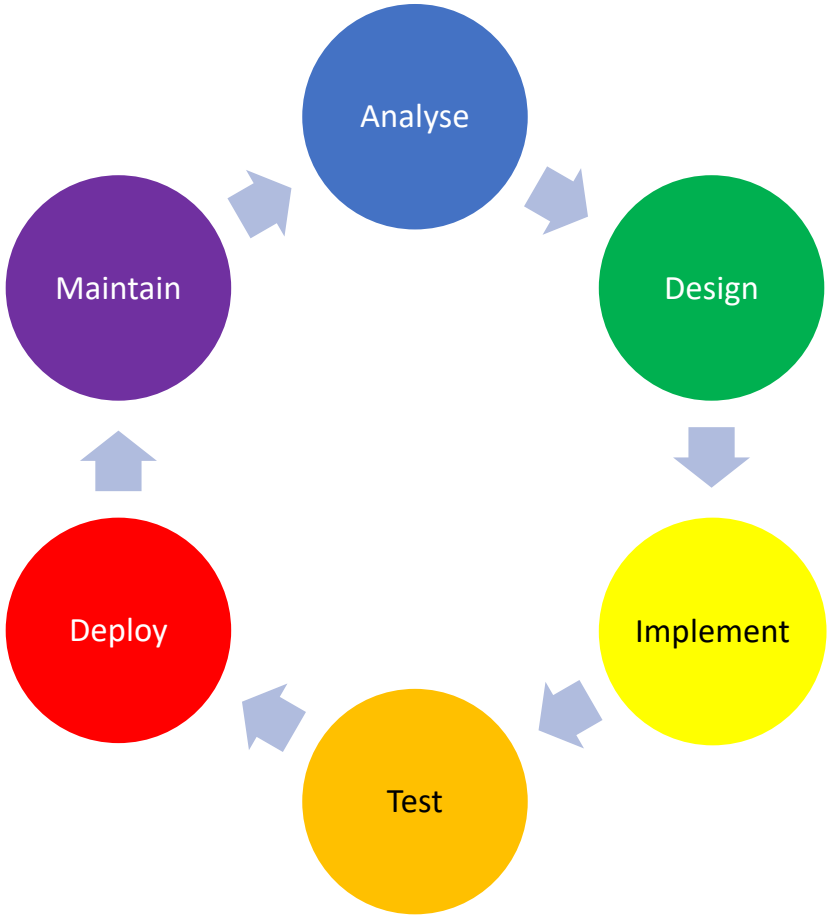
Primary Focus

- Software Quality
- Testing
- Debugging
- Defensive Programming
- Analysis
- Design

Tangentially

- Deployment (as part of Integration)
- Maintenance (as part of Code Quality)

Why do we care?



We Want Quality Software!

- What makes up quality software?
  - Here is a start of a list:
- This course is about how we get there.

User Perspective	Developer Perspective
Fulfils its purpose	Understandable, clear logic
Usable, consistent	Portable
Efficient	Well documented
Secure	Adaptable, extensible, easily modifiable
Robust	Modular
Correct, tested, reliable	Maintainable
Scalable	Reusable
	Testable

After You Finish This Course

Do

- Write clear and consistent code
- Use Git to manage a code repository
- Review another person's code
- Write unit tests for a piece of code
- Use a symbolic debugger to debug a piece of code
- Refactor procedural code
- Apply defensive programming practices, including assertions and exceptions

Understand

- Differentiate between well written and poorly written code
- Describe what tests are needed for a piece of code
- Determine why a program is not working properly
- Explain why a piece of code needs to be refactored
- Suggests ways to make a piece of code more robust
- Identify what questions need to be addressed for a given project
- Analyze the quality of a design based on the SOLID principles
- Explain what heuristics to apply at each stage of the design process
- Suggest an appropriate software architecture for a given problem

Key 

Points

- **Read the Syllabus!**
- **Download Microsoft Teams and join the class Team**
- Be sure your Dal email is working.
- Be sure your Brightspace account is working.
- **Start on Assignment 1!**
- Lab 0 is the week of September 11-15 (check your lab section) to help you with set-up for Assignment 1 (**bring your laptop**).
- The goal for this course is to
  - To give you a toolbox of skills to make you a productive software developer
  - Understand that software development is much more than just programming

Image References

Retrieved November 17, 2017, or later

- http://clipart-library.com/expectations-cliparts.html
- https://www.google.com/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&ved=0ahUKEwjX38K138PXAhUmj1QKHXbJBQ4QjRwIBw&url=https%3A%2F%2Fwww.tes.com%2Flessons%2FeJaBi5gFlgQeFw%2Fexercise-health-and-lifestyle-unit-31-session-1&psig=AOvVaw0PHa9BrLoSKGfwlSwB40Pb&ust=1510944063771131
- https://www.canstockphoto.com/skills-knowledge-abilities-6235368.html
- http://mzayat.com/clipart/to-do-list-clip-art.html
- https://us.123rf.com/450wm/ylivdesign/ylivdesign1703/ylivdesign170304532/75107211-stock-vector-joint-pipe-in-form-y-letter-icon-simple-style.jpg?ver=6
- https://leifdavidsen.wordpress.com/2016/01/05/a-message-awakens-what-is-ibm-mq-and-why-do-you-need-it/
- https://mriet.wordpress.com/2012/07/01/merge-bubbles-are-bad/
- http://blog.wiser.com/rule_based_pricing_vs_algorithmic/
- http://pengetouristboard.co.uk/vote-best-takeaway-se20/