Introduction to the Course

Individual Software Process

Description in Course Catalog

กระบวนกำรพัฒนำซอฟต์แวร์สมัยใหม่ กำรพัฒนำแบบ วนรอบและแบบค่อยเป็นค่อยไป กำรวำงแผนและประมำณ

โครงกำรเดี่ยว กำรจัดกำรเวลำ กำรติดตำมเวลำ คุณภำพรหัส

โปรแกรม กำรปรับปรุงรหัสโปรแกรม กำรตรวจสอบรหัส

โปรแกรม กำรควบคุมรุ่นของรหัสโปรแกรม กำรทดสอบ

ซอฟต์แวร์เบื้องต้น กำรพัฒนำซอฟต์แวร์ภำยใต้กรอบงำน

Modern software development process, iterative and incremental development, individual project planning and estimation, time management, tracking time, code quality, code refactoring, code review, source code version control, introduction to software testing, software development under a modern framework.

Purpose of This Course

Developers work on projects in teams.

They apply a process to their projects.

Individual Software Process - skills, knowledge, and habits to be an **effective developer** alone or on a team.

Goal of the Course

Understand and be able to apply software development skills used by individuals & teams

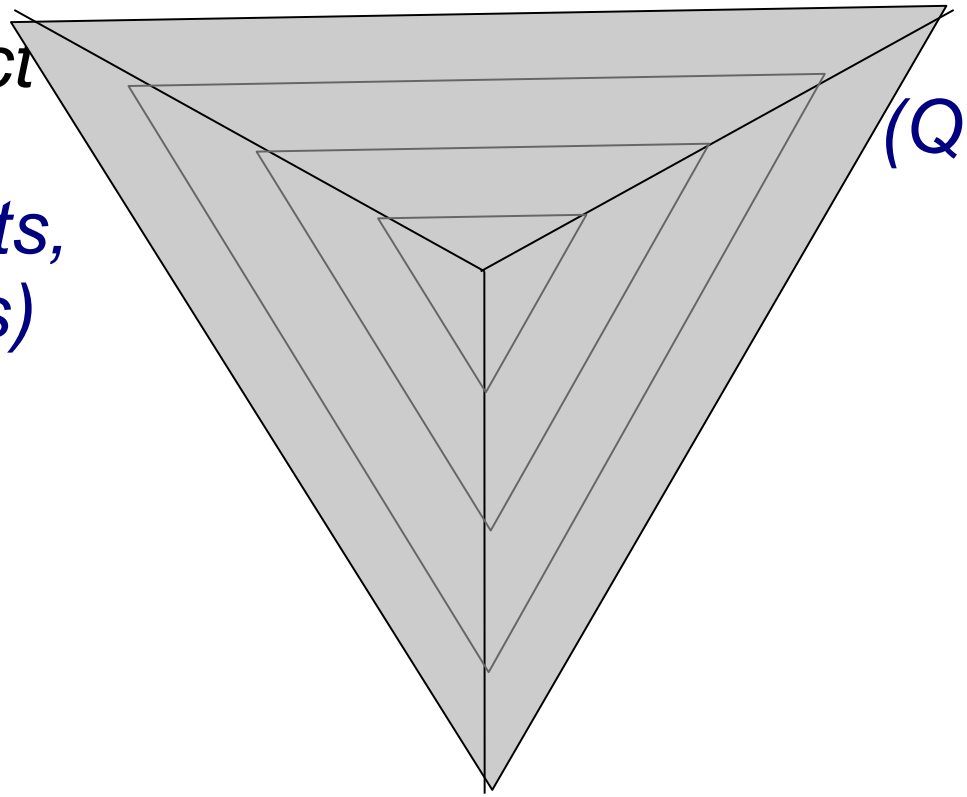
Improve your ability to write good quality code that is testable and maintainable

Topics

Conceptual Knowledge	Skills	Technology	Habits
Software processes Process areas and practices Iterative & Incremental dev, Agile concepts Waterfall	Estimation Planning Tracking Work Testing Review design & code Build Management Refactoring Retrospective	Git Python unittest Persistence Task boards Issue tracking Automation, CI Build tools	Clean Code Quality Focus Attention to detail Self-learning Communication skill Time Mgmt.

Dimensions of a Software Project

Build the right product (Features, Requirements, User Needs)

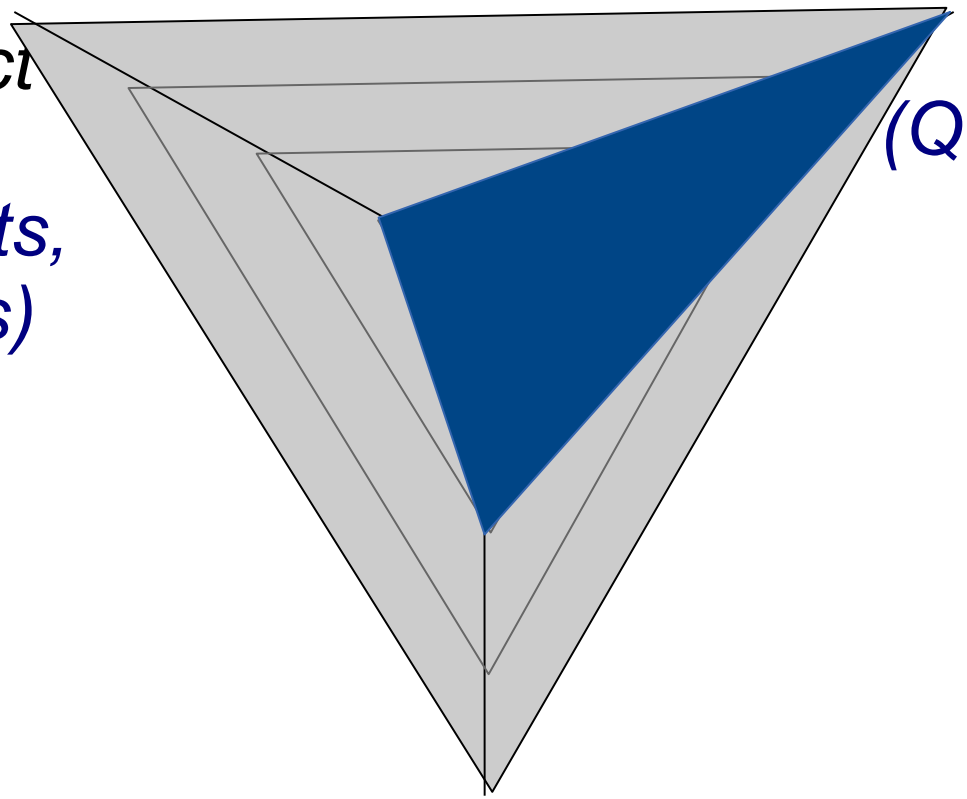


Build the product right (Quality, Process)

Build it fast (Time & Cost)

Focus of this course

Build the right product (Features, Requirements, User Needs)



Build the product right (Quality, Process)

Build it fast (Time & Cost)

Prerequisites

1. Can write O-O style code at the level of **Programming 2**.

2. **Git basics**: create & clone a repo, update files, push changes, view changes to files.

3. How to use **command line** to navigate the file system, manipulate files, and enter commands.

4. How to use Github and Github Classroom.

See: **"Git" topics on https://skeoop.github.io/**

Programming 2 *Really* is Needed

Everyone should *at least* have completed Prog 2 for basic O-O and programming skills.

If you have not, this course will be too difficult -- and a waste of your time.

Pass Programming 1 and Programming 2 **first**.

Then take ISP.

You will learn more.

Not a PowerPoint Course

"Slides" are an aid to presentation, but do not contain much detail or depth.

For real in-depth learning you must read the assigned material and do the work.

Studying from "slides" is not enough to pass the course (or get a job).

Work and Grading

1. Weekly assignments - in lab and homework
2. Quizzes
3. Written Exams
4. Programming Exams
5. Small team project - a web application

Grading

Your grade is based on your understanding of the material and ability to apply it,

as demonstrated on exams, quizzes, class participation, and assignments.

Minimum Requirement to Pass

An average exam score >= 50% on both written exams and programming exams.

Why?

You must understand concepts and how to use them.

You must be able to write and test code.

Approximate Grading Scale

A	85% and above
B	75% - 85%
C	65% - 75%
D	55% - 65%
F	less than 55% overall
	or exam average < 50%

The Rules

1. No copying
2. Do assigned reading & work
3. Submit work on time
4. Write good quality code
5. Use the coding standard
6. Participate in class

Write Good Quality Code

1. Write code that is easy to read.
2. Write code that is testable.
3. Consistently use a naming & coding style standard
4. Write meaningful comments.
   Include Pyton docstring or Java Javadoc comments.

No Comments -> No Credit

Bad Coding Style -> No Credit

Two Things We Won't Tolerate

1. Copying

Copy anything → Fail (F)

Including Homework.

No second chance.

2. Laziness

Signs of laziness:

- Not doing assigned reading
- Wait until last day to do homework
- Not participating
- Submitting sloppy or buggy code
- Copying

Online Course Resources

Google Classroom. **`https://classroom.google.com`**

- Assignments, announcements, feedback, discussion

Github Classroom: programming work

- **`https://github.com/orgs/ISP2022`**

Course Material: **`https://cpske.github.io/ISP`**

- Organized by topic, not sequential order

Discord: for meetings & monitoring quiz/exams