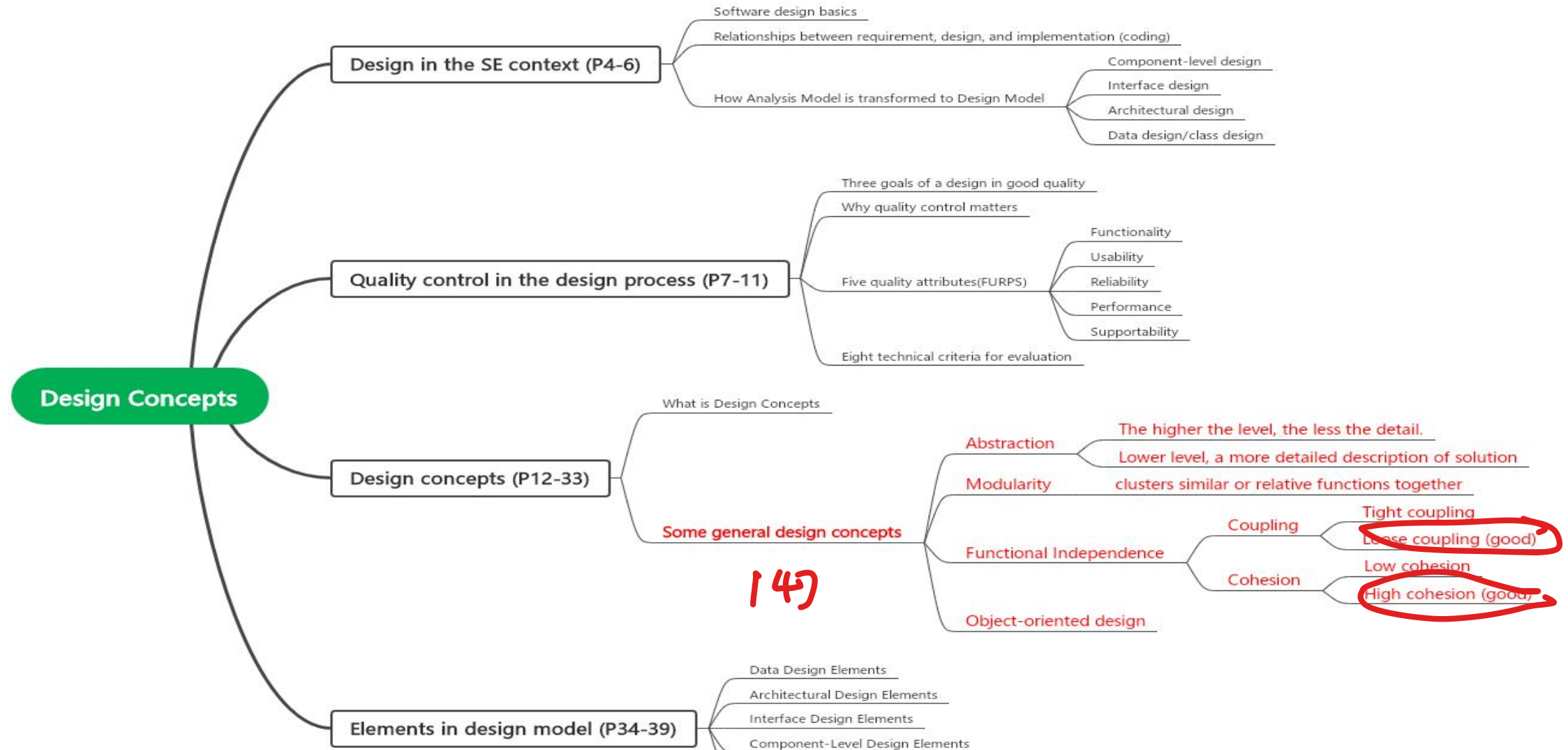


Revision Week 8-12

CPT 203 Software Engineering

AY 24/25

Week 8 Structure

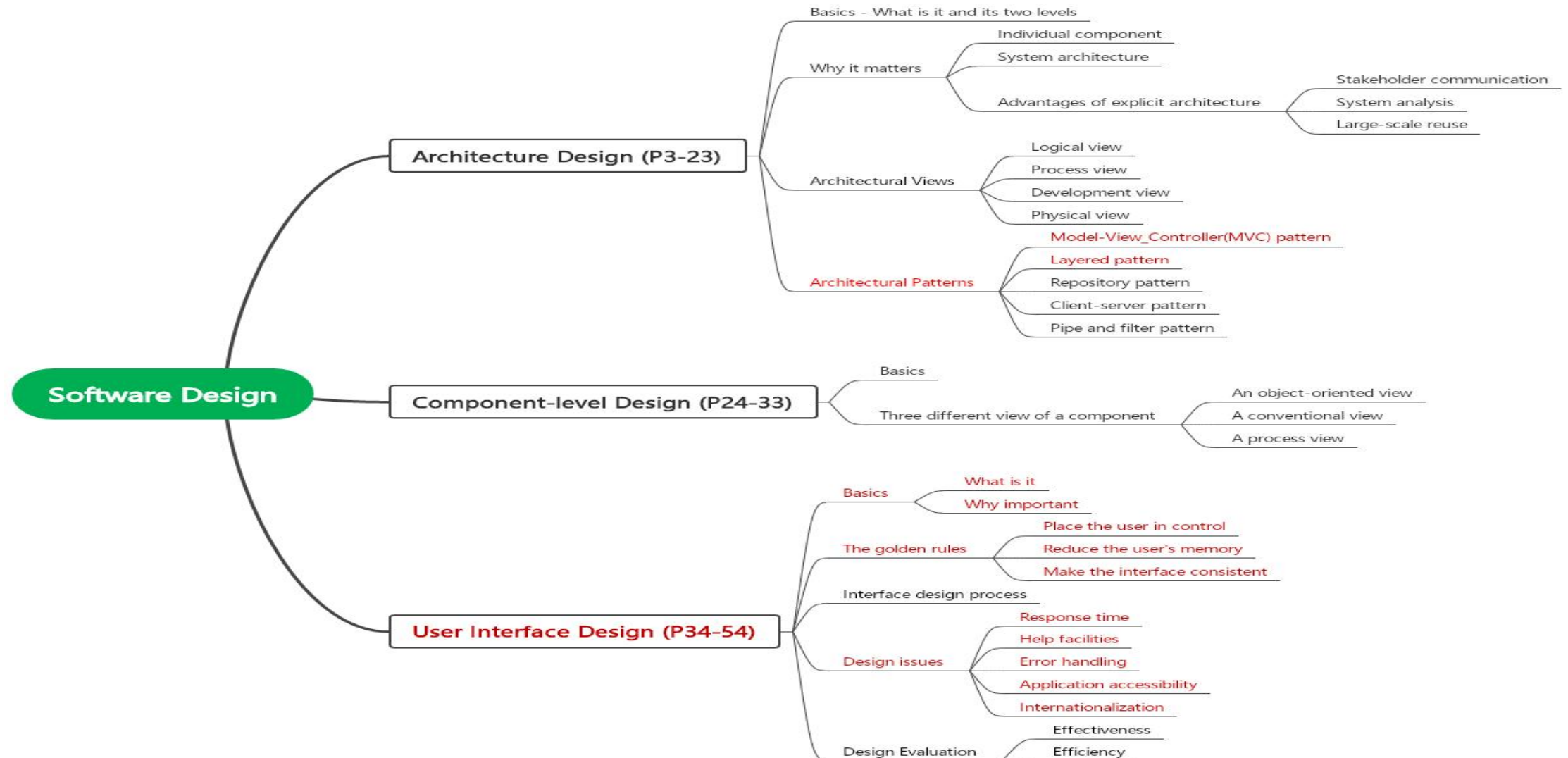


147

Week 8 Try-to-do

- Understand the definition of the each design concepts
- Understand the examples provided in the lecture
- Practice all the Week 8 TTL questions again and CW2 Q2
- Understand the importance of each design concepts (why we shall use them, what advantages, etc. You may rely more on the textbook --
- *Software Engineering A Practitioners Approach 8the Edition, chapter 12*)

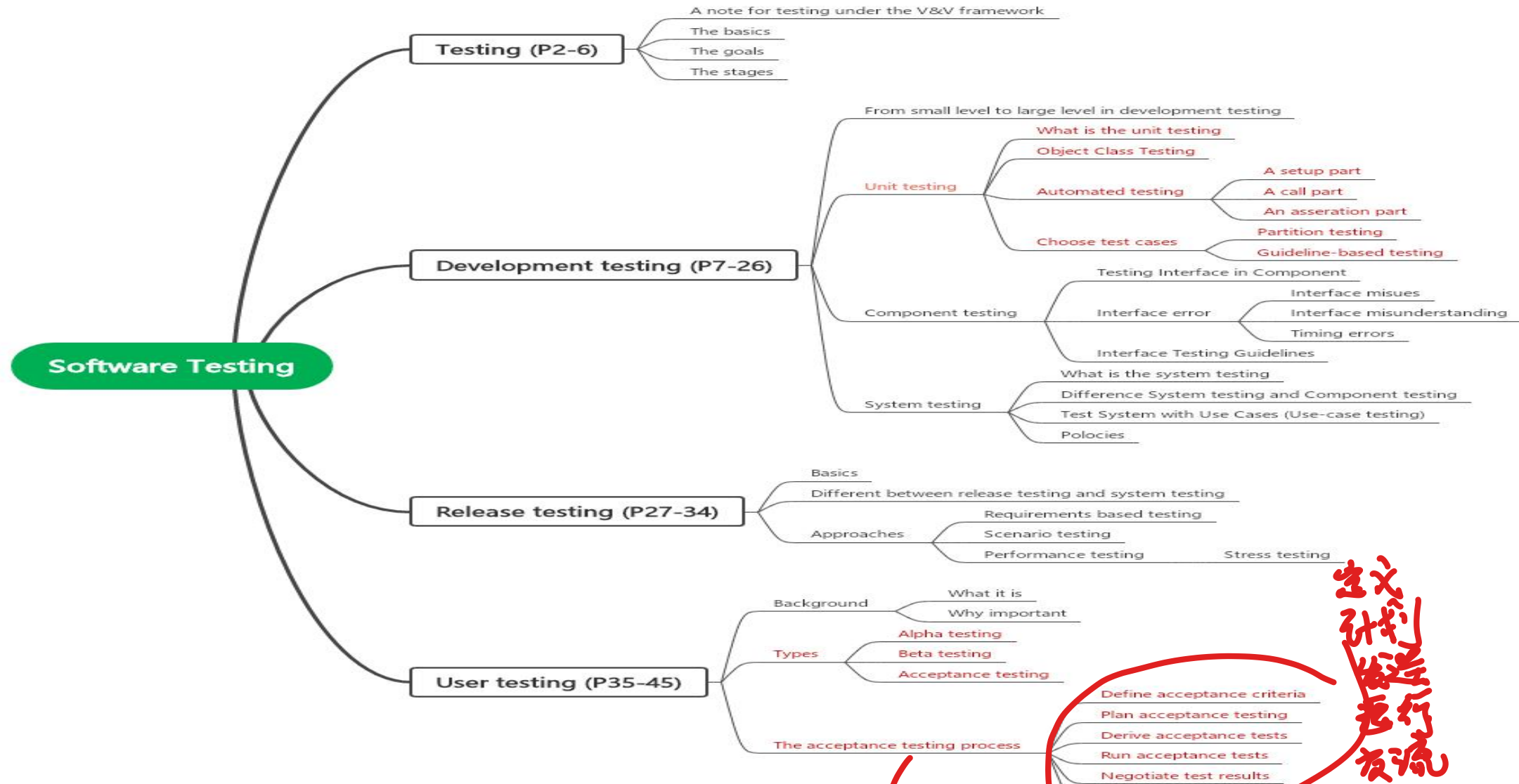
Week 9 Structure



Week 9 Try-to-do

- Understand the architecture design patterns covered in the lecture (MVC, layered pattern) and tutorial (the rest 3) - how they work, suitable to which type of software, etc
- Understand 3 general UI design rules (principles) and the related sub-rules involved
- Understand the definition of the issues involved in the UI design
- Practice Week 9 TTL Q3, CW2 Q4
- No figures needed, but should be able to describe the rules and why you use them (benefits)

Week 10 Structure

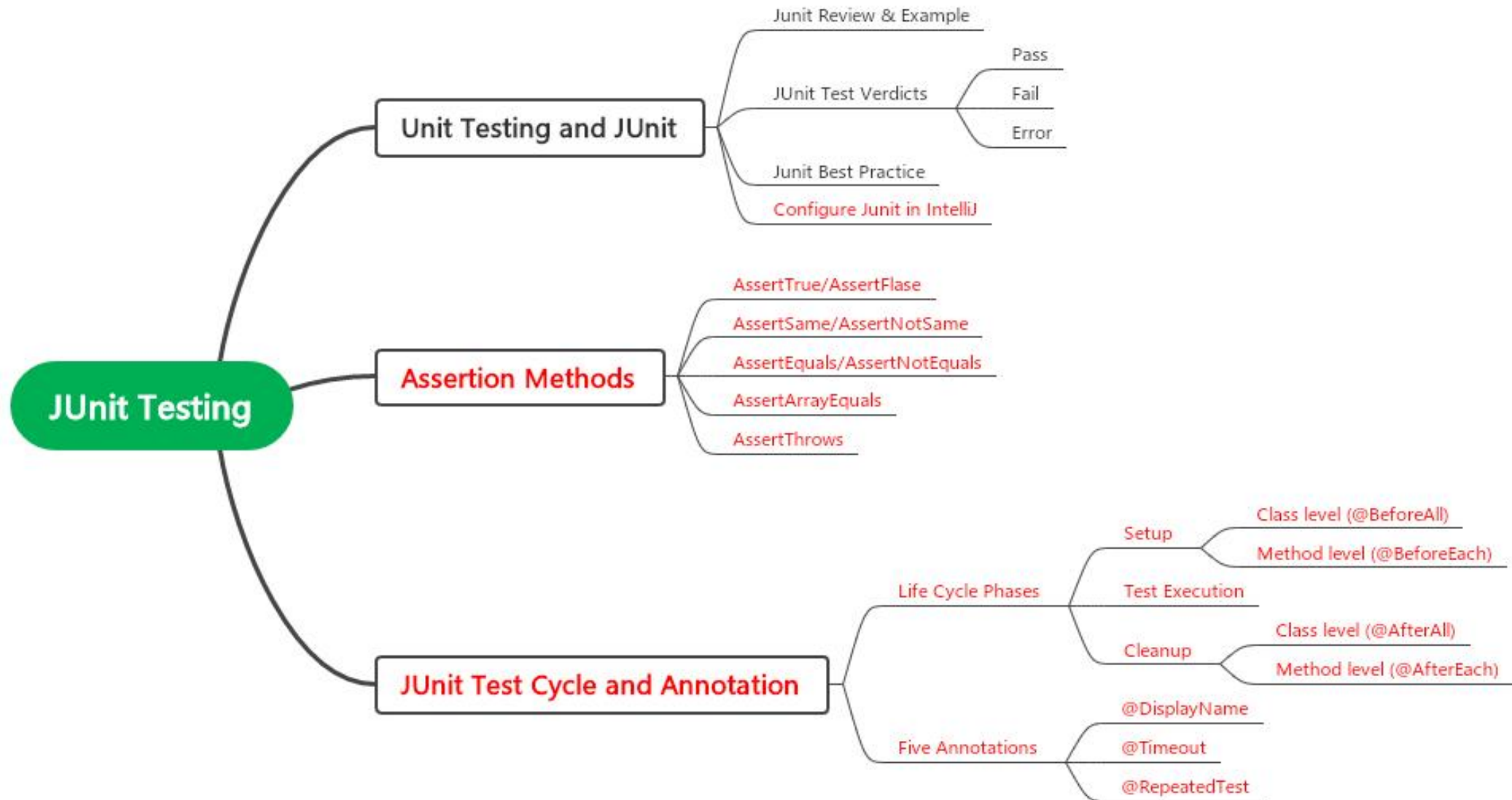


定义
计划
执行
验证
反馈
R/A

Week 10 Try-to-do

- Understand the testing stages (i.e., development, release, user testing) and the involved sub-stages.
- Prioritize Unit testing, focusing on the process we do for an object class (testing attributes, methods, and states) and the strategies we can use (Partitioning)
- Understand the user testing stages
- Practice Week 10 TTL Q1 Q4, CW2 Q7 (partially)

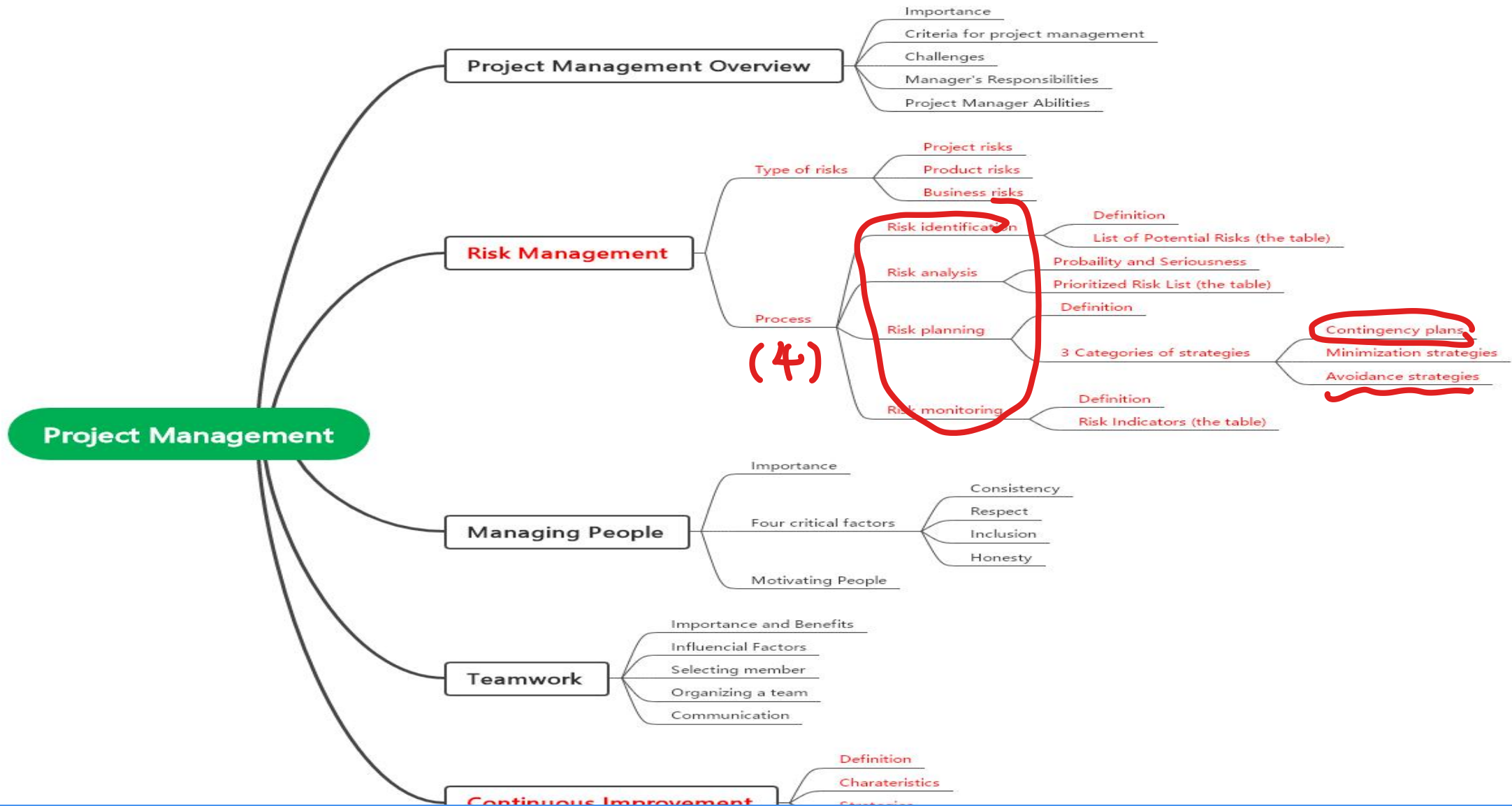
Week 11 Structure



Week 11 Try-to-do

- Do try the codes uploaded on LMO
- Understand all the assertion method covered in the lecture
- Understand the annotations, especially the basic ones (e.g., @BeforeAll, @BeforeEach, @Test)
 - @Test -> 2 marks
 - @test/@TEST/@Tests -> 0 mark
- Practice Week 11 TTL Q2 Q2, CW2 Q6 (the assertThrows())

Week 12 Structure



Week 12 Try-to-do

- Understand all stages involved in risk management
- Understand the definition, characteristics, strategies, and metrics involved in continuous improvement
- Practices Week 12 TTL Q1, CW2 Q7
- A term: Quality management systems (QMS)
 - Refers to a wide range of coordinated policies, processes, and practices that are designed to ensure the software products meet a certain level of quality and consistently satisfy customer requirements
 - It covers the quality control process throughout the entire software engineering process of a project, from requirement collection to project future iteration
 - For this class, you should at least understand the quality management system from 4 perspectives:
 - Why software design is important in software quality
 - Why software testing is important in software quality
 - Why risk management is important in software quality
 - Why continuous improvement is important in software quality

Finding the Past Exam Paper

- <https://lib.xjtlu.edu.cn/>

The screenshot shows the XJTU Library homepage with six service tiles. The 'Exam Papers & Theses' tile is highlighted with a red border.

- Discover**
All-in-one Library searches for print book records, e-books, journal articles, and a wide range of digital content.
- Catalogue**
Explore print book records, manage renewals and reservations, and request inter-branch delivery services.
- E-Journals**
Access the electronic journals and magazines provided by the Library by entering the title in the search field.
- Databases**
Alphabetically lists all Library databases to enable efficient browsing and sorting through a user-friendly interface.
- Exam Papers & Theses**
Access past exam papers, as well as previous Final Year Projects (FYPs), dissertations, and theses from XJTU graduates.
- Research Repository**
Preserve and/or search for XJTU-affiliated scholarly output; records are searchable by keywords or phrases.

The screenshot shows the XJTU Library search interface. The search bar is highlighted with a red border. The search results table is shown below.

Advanced Search

Paper Code Contains cpt203

Sort by: Paper Title Search

Paper Title	Paper Code	Department	Year
Software Engineering I	CPT203	Department of Computing	2021-2022
Software Engineering I	CPT203	Department of Computing	2021-2022
Software Engineering I	CPT203	Department of Computing	2022-2023
Software Engineering I	CPT203	Department of Computing	2022-2023
Software Engineering I	CPT203	Department of Computing	2023-2024
Software Engineering I	CPT203	Department of Computing	2023-2024

Suggestion 1

- ii. Two software engineers are arguing on how to modularize their software design. One engineer decomposes the design into too few modules while the other engineer decomposes the design into too many modules. What factors must the engineers be considering when decomposing a software design? **(5 marks)**

Try to structure your answer based on the marks

- Just answer the two correct factors - cost of effort (per module) and number of modules - not full marks
- So you do:
 - ~~Name the correct factors~~
 - Explain that designers should balance the cost of each module and the number of modules based on the intersection point of the two curves (*point M, W8, page18*)

Suggestion 2

Q4. For each of the UI page, illustrate what interface design principles are used, how they are applied with specific examples, and how they improve the interaction between the system and users (e.g., equality, diversity and inclusion). (20 marks)

Try to redo CW2 questions on your own

- Not only for this Q4 but also other questions
- Review these questions together with the teammembers might be a good idea

Suggestion 3

- Use 72 as the test case
- Test a case that fails the function

```
public class Grade {  
    public static char getLetterGrade(int mark) {  
        if (mark >= 75) {  
            return 'A';  
        } else if (mark >= 60) {  
            return 'B';  
        } else if (mark > 50) {  
            return 'C';  
        } else {  
            return 'F';  
        }  
    }  
}
```

Below is the code for a simple `Calculator` class that provides methods to perform basic arithmetic operations.

```
public class Calculator {  
    public int add(int a, int b) {  
        return a + b;  
    }  
    public int subtract(int a, int b) {  
        return a - b;  
    }  
    public int multiply(int a, int b) {  
        return a * b;  
    }  
    public int divide(int a, int b) {  
        if (b == 0) {  
            throw new ArithmeticException("Division by zero");  
        }  
        return a / b;  
    }  
}
```

Do try the codes uploaded on LMO

- The Junit question typically has two cases for you to test (i.e., the working and non-working), do not miss any
- When you see Exception, you think `assertThrows()`
- Annotation matters, not only `@Test`

Suggestion 4

Question A4 (12 marks)

List and explain the FOUR steps of Project Risk Management. (12 Marks)

Question A9

Can you please define Risk Management and provide explanations along with examples for the three categories of risks? (12 marks)

Question A9

Please list and explain the three categories of strategies involved in the Risk Planning phase, and provide related examples to each of them. (12 marks)

Do read the question and tell the difference between the concepts in question

- 4 risk management steps, 3 risk categories (types), 3 strategies
- Make sure you know what are they in details, not only the names
- Quality management system -> a comprehensive process -> why design, testing, risk management and continuous improvement are important in project quality

Overall

- When reviewing W8-12, the highlighted parts here can be your **priority** (but not necessarily the only)
- Try the **TTL questions, CW2 questions, and Junit codes**
- Try **the past exams**, though solution to the past exam is not supposed to be provided, you can try the questions and come to SD423 during office hour for suggestion