CS309

OBJECT-ORIENTED ANALYSIS AND DESIGN

Yuqun Zhang （张煜群）
Department of Computer Science and Engineering
Southern University of Science and Technology

Who and Where Am I?

- Dr. Yuqun Zhang （张煜群）
- Research Interests: Software Analysis and Testing (Fuzz Testing, Taint analysis, Defect Prediction, Code Summarization, etc.)
- Email: zhangyq@sustech.edu.cn
- Office: Room 610, Engineering Building
- Office Hours: 4-6pm, Thursday, or appointment by email

A LITTLE SOMETHING ABOUT ME…

My Styles and Rules

- Casual
- Interaction
- Mutual Respect
- NO CHEATING!!!!!
  - You may work together in this class, as specified on each specific assignment. Do **NOT** use any resource without citation.

Instructor and Teaching Assistant

- Yueming Zhu (朱悦铭)
  - Email: zhuym@sustech.edu.cn
- Chuan Jiang (江川)
- Jiahong Xiang (香佳宏)
- Shangxuan Wu (武尚煊)
- Chaozu Zhang（张超祖）
- Jihao Shang (商际豪)
- Xinyu Xu (徐新语)
- Lan Lu(卢斓)
- Weibao Fu (傅伟堡)
- Jialin Li (李佳霖)
- Zhiyue Wang (王志越)

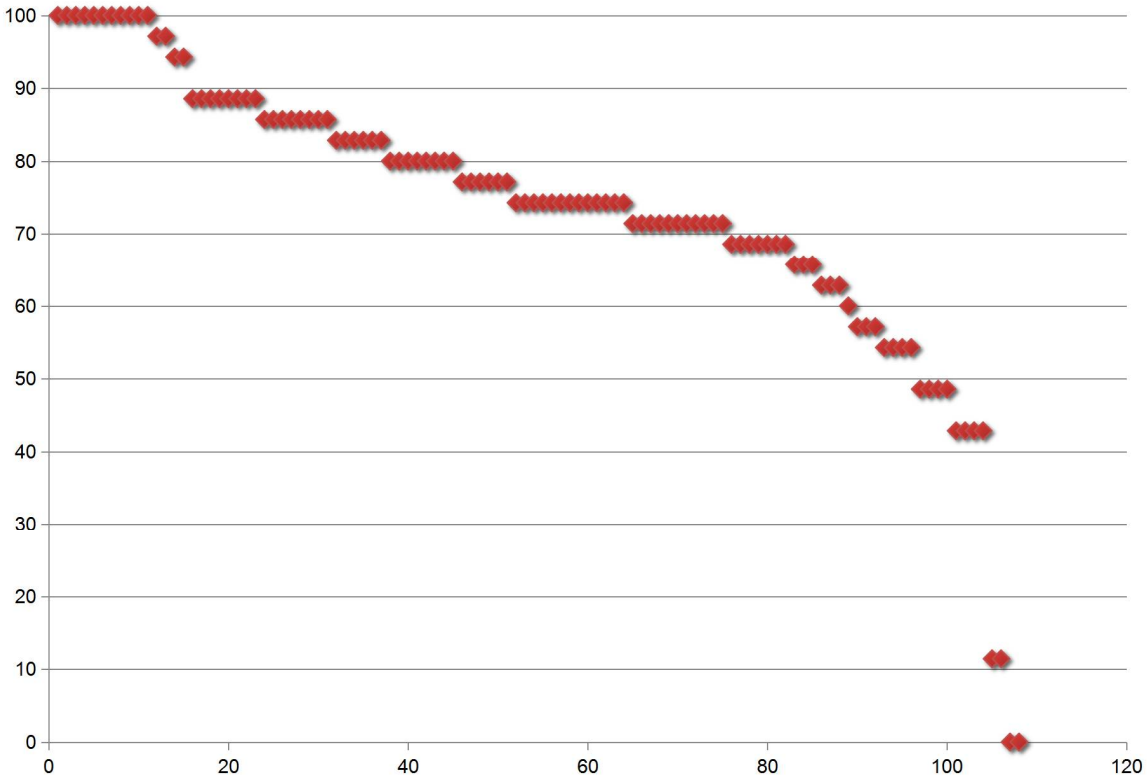
Textbooks

- Freeman et al., *Head First Design Patterns*
- *Martin Fowler, Refactoring*
- Block, *Effective Java*
- Zeller and Krinke, *Essential Open Source Toolset: Programming with Eclipse, JUnit, CVS, Bugzilla, Ant, Tcl/TX and More*
- McConnell, *Code Complete: A Practical Handbook of Software Construction*
- Pilone, *UML 2.0 Pocket Reference*

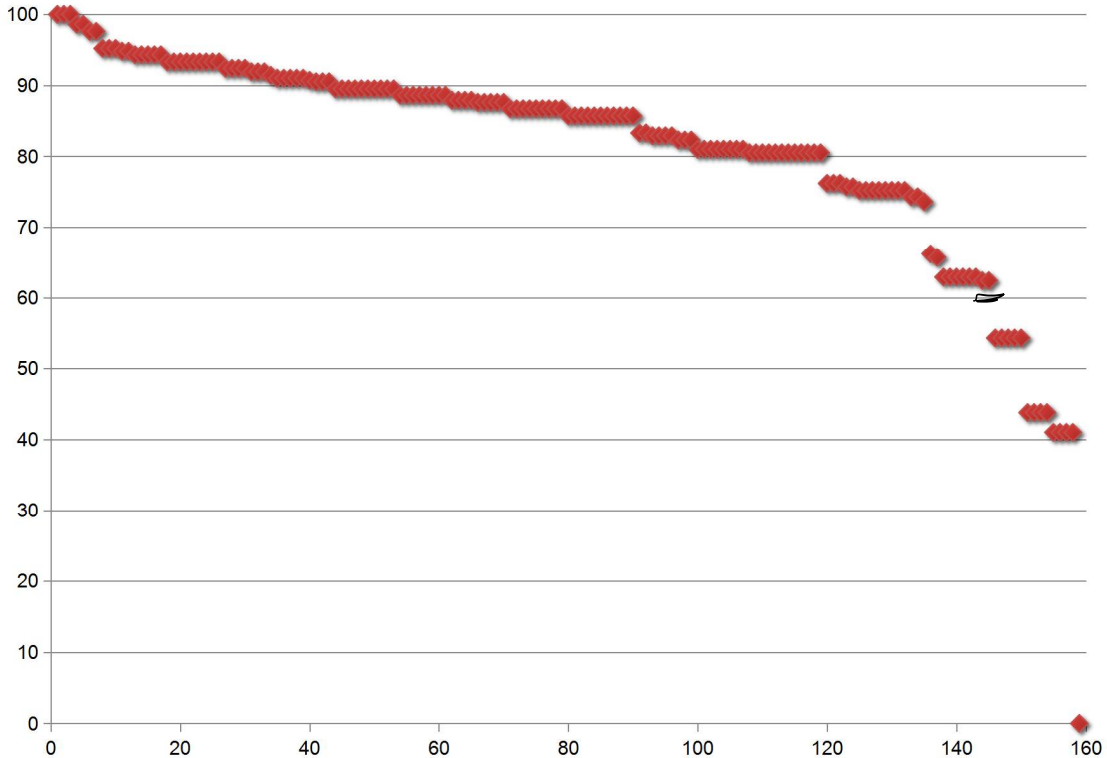
Evaluation and Grading

- Weekly Lab Tutorials– 25%
  - ~5 times
- Project – 35%
  - Web applications/Games
  - Group of 4 to 5 (before the end of next week)
  - 3 presentations (proposal, progress, final)
  - 1 written report 国庆 { requirement / features
- Exams – 35%
  - Final:
    - What's on an exam? Anything from any aspect of class, including lab sections.
    - No hints (重点)
- In-Class Exercises/Attendance – 5%
  - Spontaneous (That means in general I do not call the roll. But I have my own moves  )

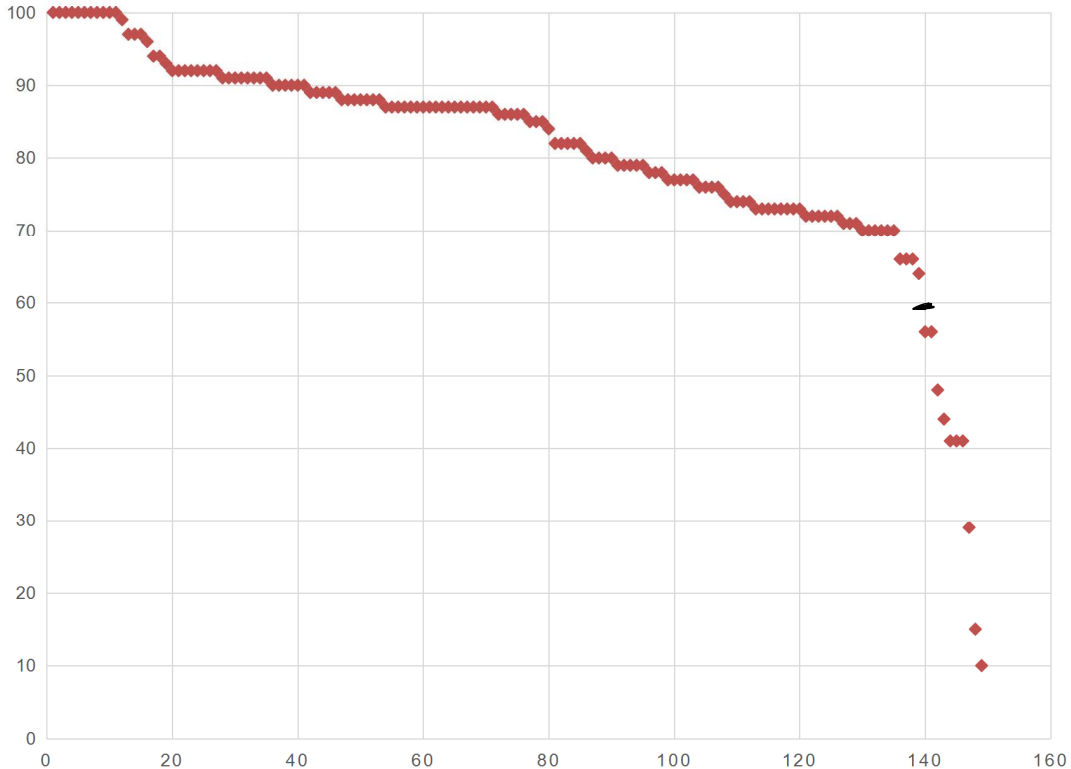
Project in 2017 Fall

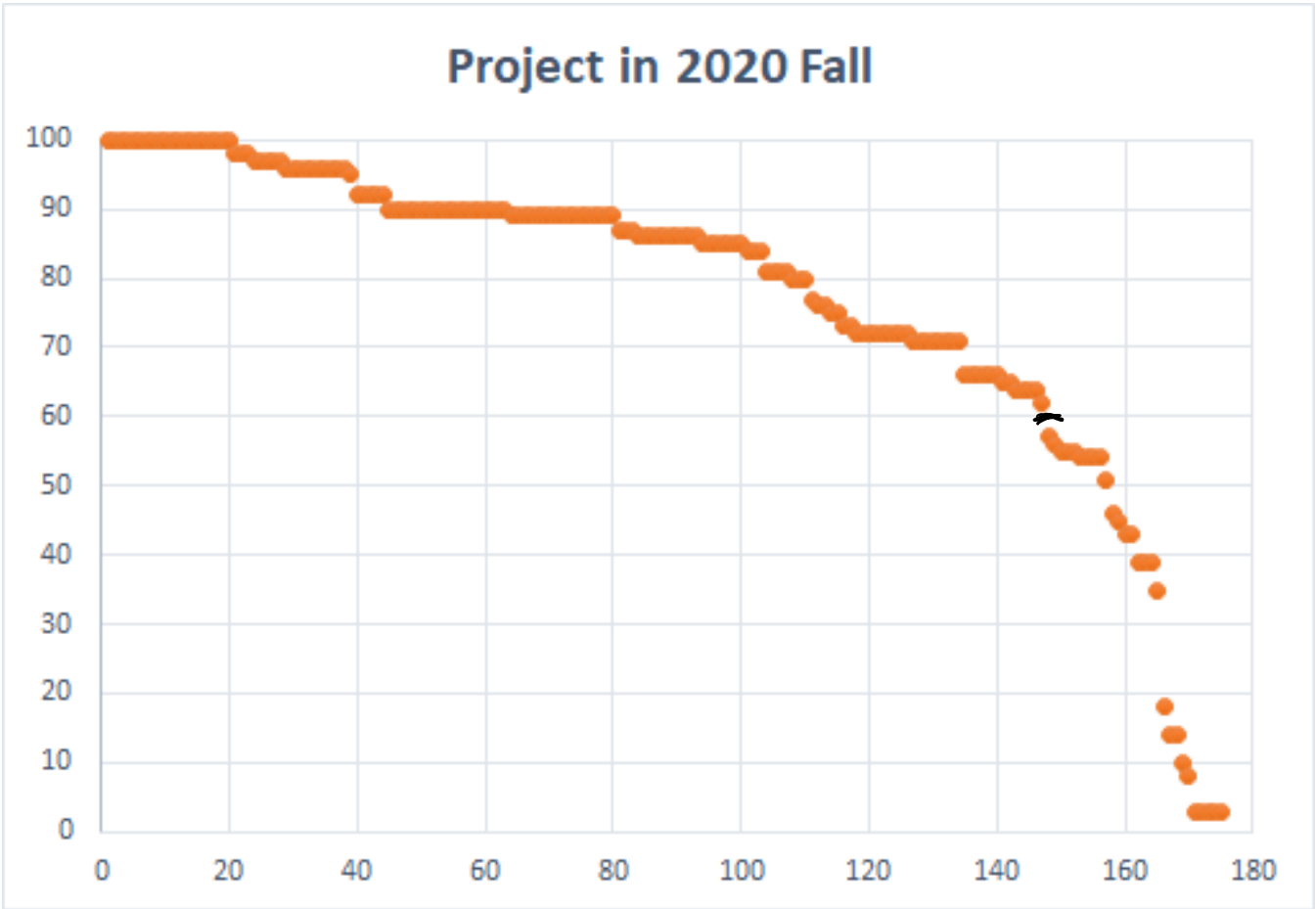


Project in 2018 Fall



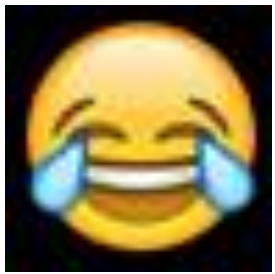
Project in 2019 Fall




Project in 2020 Fall
100
90
80
70
60
50
40
30
20
10
0
0
20
40
60
80
100
120
140
160
180

You could say this is a “breathing” class (not a “氵” class).

I just want you to be happy in this semester.



ALRIGHT, LET'S GET REAL

Expectations

- You're going to have to "own" your education in this class
  - I have a feeling this is going to be an awesome semester…
- But…
  - Expect that I may not be able to give you an immediate answer (I'm alright if my response to your question is "I don't know," so you're going to have to be alright with that, too)
  - I (or the TAs) WILL always try to help find you the answers you need in a timely fashion. Be patient.

Tips of Handling Problems

- Once you encounter problems (theory or practice), you are expected to
  - first, try your real best to solve them by yourself

Tips of Handling Problems

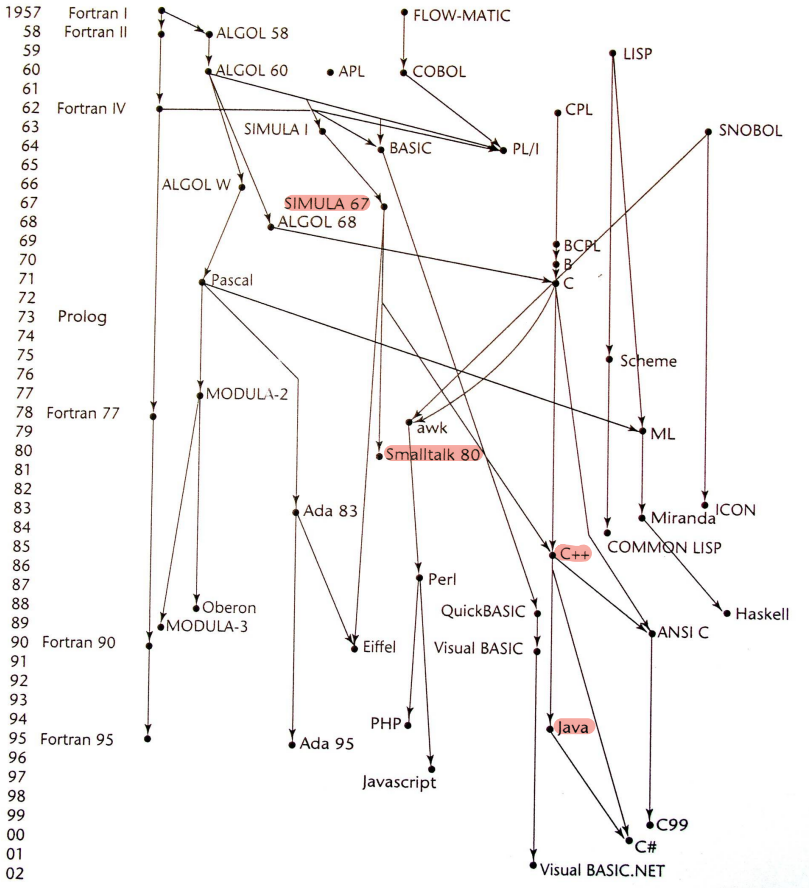
- Once you encounter problems (theory or practice), you are expected to
  - first, try your real best to solve them by yourself
  - if not working, try to talk with your cohorts.

Tips of Handling Problems

- Once you encounter problems (theory or practice), you are expected to
  - first, try your real best to solve them by yourself
  - if not working, try to talk with your cohorts.
  - if not working, then ask us

Tips of Handling Problems

- Once you encounter problems (theory or practice), you are expected to
  - first, try your real best to solve them by yourself
  - if not working, try to talk with your cohorts.
  - if not working, then ask us
- If we find that you are not paying effort by yourself, we would be reluctant to help you at later time.


1957
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
00
01
02
Fortran I
Fortran II
Fortran IV
Prolog
Fortran 77
Fortran 90
Fortran 95
ALGOL 58
ALGOL 60
APL
FLOW-MATIC
COBOL
LISP
CPL
SNOBOL
SIMULA I
BASIC
PL/I
ALGOL W
SIMULA 67
ALGOL 68
BCPL
B
C
Pascal
Scheme
MODULA-2
awk
ML
Smalltalk 80
Ada 83
ICON
Miranda
C++
COMMON LISP
Perl
QuickBASIC
Haskell
Oberon
MODULA-3
ANSI C
Eiffel
Visual BASIC
PHP
Java
Ada 95
Javascript
C99
C#
Visual BASIC.NET

What You Would Learn

- Of course the object-oriented design and analysis
- Typically, you are going to learn something about
  - requirement engineering (UML)

UML Examples

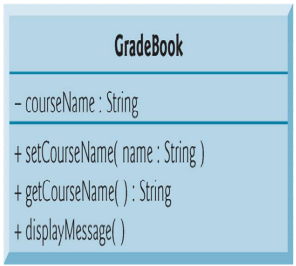


Fig. 7.3 | UML class diagram for class GradeBook.

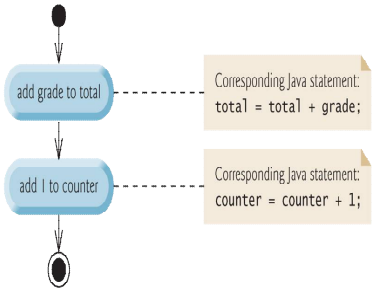


Fig. 3.1 | Sequence structure activity diagram.

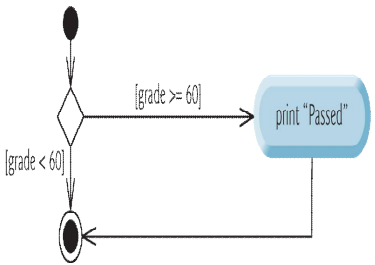


Fig. 3.2 | if single-selection statement UML activity diagram.

What You Would Learn

- Of course the object-oriented design and analysis
- Typically, you are going to learn something about
  - requirement engineering (UML)
  - design pattern (including information hiding, design principles,etc)
  - refactoring
  - web app frameworks? testing?

Your Projects

- Each group picks one problem from a pool
- We created 5 projects for you. Yet you can work on your own if you want to.
- Come talk to me if you want to come up with your own ideas. DO EXPECT THAT YOUR IDEAS MIGHT BE ASSIGNED WITH A LOWER STARTING SCORE.
- One contact person is needed for each group (This year, almost all the project requirements are proposed by student assistants. It is pretty necessary to contact them during your progress. They are the boss!).

The Project List

- Project Task Manager: Build a website for assisting teachers and students to manage course projects
- Sustech Store: Build a website to provide services for trading used goods for Sustechers.
- Interactive OJ: Build an online judge system that can be specifically adopted for class quizes.
- SUSTech Pac-Man: Build a Pac-Man game using SUSTech as background
- SUSTech Pokemon: Build a Pokemon game using SUSTech as background

Project Task Manager:

- Basic requirements:
  - Creating tasks with some attributes, such as status, assignees, priority, and deadline.
  - Tasks may have subtasks.
  - Users can create a team and add tasks to the team. The team dashboard can view the tasks owned by the team, team members, etc.
  - The user dashboard can view the tasks assigned to the user.
  - Providing statistics, e.g., how many tasks a user has done, how many tasks a team has done, the user's contribution to its team, etc.

Project Task Manger:

- Advanced requirements
  - Authority management, e.g., task authority, user authority.
  - Convenient UI design, such as adapting webpages for phone screens, draging to set subtasks, displaying tasks in the tree diagram, identifying the datetime of text input, etc.
  - Third-party software integration, such as email reminders after tasks are scheduled, displaying tasks on the calendar, system notifier, etc.
  - Offline access (synchronizing data when online)
- Stakeholders for Requirements:
  - Chuan Jiang
  - Chaozu Zhang

Sustech Store:

- Basic requirements,
  - Users can publish selling and buying information, including price, categories, labels, etc.
  - Users can publish sell and buy goods according to selling and buying information published.
  - Users can search goods based on their information
  - Integrating chatting system for users to communicate when trading.
  - Integraing notification system to deliver transaction information, e.g., email, QQ, etc.
  - Supporting payment options, such as using virtual currency and integrating third-party payment platform.

Sustech Store:

- Advanced requirements,
  - Integrating campus life properly
  - Supporting users' authorization to prevent masquerading
  - Designing a credit system with standardization management which can restrict users' transaction behaviors.
  - Supporting purchasing agent service, e.g., bringing lunch or documents
- Stakeholders for Requirements:
  - Jihao Shang
  - Xinyu Xu

Interactive OJ

- Basic requirements:
  - Two roles: administrator (teacher) and user (student)
  - Students:
    - Submitting code and viewing submission results including execution results, execution time, and error information.
    - Viewing historical submission records and statistics of submissions
  - Administrator (Teacher):
    - Assigning/modifying questions, including adding question descriptions, code templates (refer to demo) and one or more test cases.
    - Dividing students into different Labs and setting the start time, end time and visibility of a quiz in a certain lab.
    - Viewing student/lab submission status and real-time statistics. Providing filtering functions, such as filtering according to question completions.
    - Feedback of the correct reference code to the students/labs

Interactive OJ

- Advanced requirements:
  - Supporting both java and python
  - Three types of roles: teacher, teaching assistant, student (refering to sakai)
  - Supporting running user-defined test samples (refering to LeetCode)
  - Importing and exporting related information of students/labs/quizzes by csv or excel
  - Preventing malicious attacks
  - Concurrency design: supporting up to 200 concurrent requests with performance optimization.
  - Fast deployment on Linux server. (https://github.com/halo-dev/halo)
- Stakeholders for Requirements:
  - Jiahong Xiang
  - Shangxuan Wu

SUSTech Pac-Man

SUSTech Pac-Man

- Basic requirements:
  - Map generation: A map can be generated automatically or based on the script uploaded by the user to get a personally designed one.
  - Movement control: A player should control the movement of a Pac-Man through keyboard inputs while a AI Pac-Man (role for the machine) and ghosts should move according to the designed script (Notice that we do not require high-level AI algorithms in this part and you are not required to make effort on it).
  - Various game elements: beans, pac-mans, ghosts and other tools
  - Battle mode: human vs machine and machine vs machine

SUSTech Pac-Man

- Advanced requirements:
  - Well-designed game interfaces
  - Elements related with SUSTech
  - A switch of visual angle from 2D to 3D and vice versa
  - Fog mode: A player has visual field on only a certain area around
  - Prop mode: A player can carry and use props (For example, a magnet which can absorb beans in a certain area around)
  - Online mode: Support game of human vs human
  - Other ideas from yourself and please enjoy the process of designing a game
- Stakeholder for requirements:
  - Lan Lu
  - Weibao Fu

SUSTech Pokemon

SUSTech Pokemon

- Basic requirements:
  - Setting at least 6 different pokemons (must include: Pikachu皮卡丘, Charmander小火龙, Squirtle 杰尼龟, Bulbasaur妙蛙种子, Pidgeotto比比鸟 and Muk臭臭泥), with different attributes.
  - Pokemons should own several attributes including race (water, fire, earth, grass...race determines restraint relationship), attack, defense, health, speed (speed determines evasion), and at least three skills and at least four logics of playing cards (inclusive but not restricted)
  - Completing the battle scene, battle process and battle settlements:
    - Each battle includes 1v1 pokemon battle
    - Each player can only use up to 3 pokemons in a game. Player can use another pokemon after the last one loses his ability to fight. Pokemons who have failed one competition cannot join in the game again. Finally, the player who still has available pokemons wins.
    - Battle scenes: field background, players and pokemons information board, etc
    - Battle process: It can be described in words (similar to "Pikachu launched - 100000 volts - against the Charmander!", "Excellent result!"). Pokemon attributes and skills determine the battle results.
    - Battle results: the settlement interface gives game results and rewards, etc.
  - Completing the PvE mode (no considering the AI level), and PvP mode on single PC

SUSTech Pokemon

- Advanced requirements:
  - Enriching pokemon contents: more pokemon attributes (add special attack and special defense, etc.), additional pokemon growth content (level, evolution, character, breakthrough, skill learning, etc.), and more number of pokemons
  - Enriching battle scenes: including more attack and skill animation, battle special effects, etc
  - Adding other game elements: store, pokemon illustrated handbook, home, etc
  - Adding game scenes: level training scenes, adventure scenes (fight and capture pokemons in an adventure), etc
  - PvP mode on multiple PCs based on the internet
  - Art, creativity, UI, 3D and other contents
- Stakeholder for requirements
  - Jialin Li
  - Zhiyue Wang

Tips for your projects

- Frequently contact your stakeholders. They manage the requirements and have written detailed descriptions.
- Launch your projects ASAP. You don't want to start off just two weeks before the final ddl.
  - Whoever accomplish the projects and present them on mid-term presentation can be awarded with a bonus of 10% of your final project score.
- If you want to be better graded, you should go for as many bonus points as possible.
- You need to run your deliverables with test cases.
- Don't simply rely on the technical leader (大腿) in your team. We would grade you based on your individual contributions to the team in an even more rigorous manner than before.

QUESTIONS?