901 10110

# Basic Computer Concepts

## *Term Project*

王凡

farn@ce.ee.ntu.edu.tw

Dept. of Electrical Engineering

National Taiwan University

# Term Project

***Do whatever you like!***

- Please identify a topic that you like to do.
- Explain why it is worth doing.
- Describe the design.
- Describe how you plan to test it.
- Implement it.
- Submit it to Microsoft Windows App Store.

http://windows.microsoft.com/en-US/windows-8/apps

# Term Project
## team

- 2 students a team
  - a programmer and a tester
- They define the specification together.
- Then they work in parallel respectively for the program and testplan implementation.
- ***Note that changes to the specification in later stages of the project will cause pains of the team members. Thus such changes should be avoided.***

# Term Project
## *team, continued*

- programmer-specific job
  - design the program according to the spec.
  - implement the program according to the design.
- tester-specific job
  - design the test plan according to the spec.
  - implement the test plan according to the test design.
  - The tester is not responsible for testing any property different from the spec.

# Term project

## *checkpoint 1, team registration (3/6)*

*Submission to the TA via email the following team member information:*

- Name of the team
- Name of the programmer,
- Name of the tester, and
- Student ID numbers of the two members

# Term project

## *checkpoint 2, proposal (3/20)*

- 5 mins presentation for each team.
- *Submit to the TA via email:*
  - *the powerpoint document and*
  - *design framework, including*
    - *sequence diagrams,*
    - *use cases,*
- Explain why it is worth doing.

# Term project

## *checkpoint 2, proposal (3/20), continued*

- Specify your program
  - explain why it is reasonable.
  - Note that once the specification is fixed, it is not supposed to change. Spec. change will create the difficulties for the tester in constructing test plans.
- For the *programmer*, describe your design
  - classes, methods, control flows, …
- For the *tester*, describe your test plan.
  - **explain why it is sufficient.**

# Term project

## *checkpoint 3, midterm report (5/1)*

- Submit the midterm report, program code, test code, and test report via emails to the TA.
- For the *programmer*,
  - Explain what part of the specification and design that you have implemented.
  - Show the execution result.
- For the *tester*,
  - Explain the test plan parts that you have executed.
  - **Show the test report.**

# Term project

## *Checkpoint 4, final presentation (6/26)*

- 8 mins presentation for each student.
- Submit to the TA via email:
  - the powerpoint document and
  - the programs
- Explain your change to the specification.

# Term project

## *Checkpoint 4, final presentation (6/26)*

continued,

- For the *programmer*, describe your implementation.
  - classes, methods, control flows, …
  - ***Please do not ask the audience to trace your code in the presentation!***
- For the *tester*, show how you validate your implementation through testing.
  - instrumentation, bug reports, coverage, …

# Term Project

- The two students in a team will be graded independently.
  - *for design*: class diagrams, sequence diagrams, use cases, and state charts;
  - *for testing*: unit test template generation
  - If the program is not finished but the test plan is good, the tester can still get good score.

# Grading policy (1/3)

- Novelty of the program
- Specification quality
  + complete use cases
  + complete scenarios
- Teamwork
  + frequent discussion between the members
  + frequent delivery and testing of intermediate code.
  + frequent documentation of the projects

# Grading policy (2/3)

- Programmer-specifics

  + good design
    - clear class interface definition,
    - easy-to-maintain flow-charts

  + good implementation
    - good algorithms
    - good coding style

  + good presentation

  - change of specification

# Grading policy (3/3)

- Tester-specifics
  + Soundness
    - why we have to test this ?
  + Sufficiency
    - why this is enough ?
  + Efficiency
    - How many bugs are found with the test plan ?
    - How fast the bugs are found ?
    - What is the bug distribution chart over time ?
  + Bug report quality