COMPSCI 230 Tutorial
Extreme Programming

Q1: What is the purpose of Extreme Programming (XP)?

To “create quality programs in short time frames.”

Q2: What are the four basic concepts of XP according to Myers?

1. Listening – To the customer and other programmers
2. Collaborating – With the customer to develop the specification and test cases
3. Pair programming
4. Testing

Q3: What are the strengths of XP according to Myers?

- Works well for
  - Small to medium team size
  - In environments with frequent specification change
  - Where near instant communication is possible
- Avoids issue with customer and programming team meet to design every detail before coding begins.
- Avoids coding unneeded functionality.

Q4: The planning phase in XP differs from traditional models of software development. Traditional models typically combine requirement gathering and application design. What is the focus of the XP planning phase?

Identifying customer’s application requirements and designing use case stories that meet them.

Q5: Is there any benefit to involving the customer in the application development process?

Yes, they gain ownership and confidence in the application by being involved at every stage of development.

Q6: XP relies on unit and acceptance testing of modules where every incremental change must best tested. What is the purpose of this?

It ensures that the code still meets the specifications.

Q7: According to Myers, Extreme Unit Testing (XUT) has two ‘rules’. What are they? Underline the significant difference between standard unit testing process and XUT.

1. All code modules must have unit tests **before** coding begins.
2. All code modules must pass unit tests before being released into production.

Q8: Why are software updates to released software considered ‘hazardous’ if they change any of the software’s features?

It may cause issues with stakeholders who are familiar with the buggy/original behavior. They are likely to be ‘confused, annoyed, or even angered’ when it is ‘fixed’.

Q9: If software must be updated, what are the 2 steps that should be taken in order to reduce negative responses from stakeholders?

1. User-visible behavior should remain constant.
2. Feature change should be minimized.

Q10: The Ariane 5 rocket failure was caused by an error where a single component failure caused the entire system to fail triggering the flight termination system. What is the general rule regarding critical systems in order to prevent this sort of failure?

Critical systems should be designed to avoid a single point of failure.

Q11 Sommerville called this a ‘critical and elementary error’ in the design of the software controlling Ariane 5. Do you think this is justified? Feel free to read up on the accident: http://en.wikipedia.org/wiki/Cluster_(spacecraft)#Launch_failure

Free question.