1. Read *Software Error Costs*. Identify two main points of the paper.

·       *The “true cost of a software bug”, how it affects companies and make them lose millions of dollars.*

·       *The importance of testing a software on early stages, before it gets to a point where it can seriously impact its users.*

1. Read the *V\_V\_definitions* handout.

1. What’s the difference between verification and validation?

*Validation means developing the right system. (Follows requirements)*

*Verification means developing the system right. (Satisfies requirements)*

1. Why is specification important?

*Because it’s the basis for the entire software and it lays the framework that every developer will follow.*

1. What is the difference between “coding” and “programming”?

*Coding is a subset of programming which is basically the translation of one programing language to another one. Programming is the process of developing a system with no errors and a correct behavior.*

1. What is a “bug”?

*An error in a system that produces an incorrect result.*

1. What is the difference between testing and debugging?

*Testing is when you are making sure the system works correctly and debugging is the process of fixing the bugs found when testing.*

1. What is the domain of a program that takes two 16-bit integers and returns the maximum of them? What is the range?

*The domain is the input of the program which would be the two 16-bit integers. The range is the output which is the maximum of them.*

1. The definitions include three levels of test: unit, integration, and system. What are the difference among these? Are there other levels? What are they, and what purpose do they serve?

*Unit testing is when individual units/components are being tested, Integration testing is when combination of individual units/components are being tested and System testing is when the whole system is being tested.*

*Yes, there are other levels of testing: Acceptance, Configuration, Conformance, Correctness, Exhaustive, Installation, Integration, Performance, Regression, and Usability testing and all of them have the same purpose which is to identify any problems that would affect the system in different areas.*

1. Describe to a 3331 student the difference between static quality attributes and dynamic quality attributes.

*Dynamic attributes are the ones reflecting the system’s behavior during execution while static attributes are the ones reflecting the system’s structure.*

1. What is an operational profile? How might you establish one?

*It’s a quantitative representation of how the system will be used. You can establish one by documenting all user inputs and their occurrence probabilities in such a profile. The most used functions by the inputs should be tested the most.*

1. What are the domain and range of the following program?

Write a program that reads a sequence of integers and outputs the integers in this sequence sorted in either ascending or descending order. The order of the output sequence is determined by an input request character which should be ``A'' when an ascending sequence is desired, and ``D'' otherwise. While providing input to the program, the request character is input first followed by the sequence of integers to be sorted; the sequence is terminated with a period.

*Domain: Inputs such as the “sequence of integers” and the “input request character” which is either “A” or “D”.*

*Range: The output which in this case is the sequence sorted in either ascending or descending order.*

13. Translate the following into English:  (Ù is AND; Ø is NOT; ® is IMPLIES; " is FORALL; $ is THERE EXISTS; Î is ELEMENT OF). In the following, P is a program, S is a specification, d is an element of the domain, r is an element of the range, T is a test set, t=(d,r) is a test.  I’ve given you the first description.

* ($ d,r | S(d,r) Ù Ø P (d,r))

*There exists some element of the domain d and some element of the range r such that the relationship between d and r is in the specification, but not in the program.*

* ($ d,r | S (d,r))

*There exists some element of the domain d and some element of the range r such that the relationship between d and r is in the specification.*

* " tÎT, t=(d,r) Ù S (d,r) ® P (d,r)

*For all the tests t that are in the test set T, t is in the specification that is implied in the program.*

* T is *ideal* if

    ($ d,r | S (d,r) Ù Ø(P(d,r))) ®

                  ($ tÎT | t=(d’,r’) Ù S (d’,r’) Ù ØP(d’,r’))

*There exists some element of the domain d and some element of the range r such that the relationship between d and r is in the specification, but not in the program. Which implies that there exists an element of T such that a test that has no relationship between d and r is in the specification but not in the program.*

Page Break

14. What’s the difference between Dynamic and Static techniques?

*Dynamic techniques are used during the verification stage while Static techniques are used during the validation stage.*

15. Why do we validate software?

*Because we want to make sure that the software meets the requirements and standards set by the customer.*

16. Why do we verify software?

*Because we want to make sure that we built the right software and confirm that it fulfills the plans of the customer.*

17. What is a good test?

*A test that help us find out if a certain area met the specified requirements and if there are any problems such as bugs or faults.*

18. What is a good test set?

*A set of tests that helps identify all the defects and requirements met on the system in the most areas we can test.*

19. What are the steps needed to develop software? (Think of the SLDC.)

*Planning, Analysis, Design, Development & Implementation, Testing and Maintenance.*

20. For each step, list types and causes of faults: what can go wrong in each step of the development process?

**Planning**:

·       *Lack of communication between developers and customers.*

·       *Lack of flexibility (E.g. Assumptions being done by the developers without consulting the customers).*

**Analysis**:

·       *Excessive overhead (a lot of unnecessary documentation).*

**Design**:

·       *Lack of feedback from customers can result with having a poor design.*

**Development & Implementation:**

·       *Integration issues such as the use of APIs or additional software.*

**Testing**:

·       *Not enough testing can result on major bugs later on.*

·       *Quality issues, since a lot of rework can be required.*

**Maintenance**:

·       *New bugs introduced due to the lack of maintenance or features added.*