Honor Pledge: I pledge my honor that I have abided by the Stevens Honor System.

Midterm Exam 1: 100 Points **Name:** Alex Johnson

1. A Software Development Process pattern can provide a consistent method for problem solving the software process. This process can be described in a set of stages which get further refined as the project continues. These stages can get defined into more detailed task patterns.
2. It is extremely important for computer software to be able to evolve with time. One of the reasons for this is that technology is constantly changing, and the software needs to be able to meet all the specifications of changing environments. Another reason software needs to be ready to evolve is because over time the requirements might change. Certain customers who use the product may find they need it to do more or adapt differently to their changing environment, therefore the software needs to adapt and change. Finally, software eventually most likely will need to be fixed or refurbished. Therefore, it is good to make sure the code is updated and ready to adapt so it is easier to fix and does not need to be completely re-written.

3. It is extremely important for processes to be agile. Agility is the act of accomplishing things very quickly with full completeness. This also means being ready to accept any challenges presented to your team. Agile software means to be growth oriented and build on a project incrementally. Another piece of agility is keeping things cost effective. All in all, agility is building a project in the most cost effective and timely manner which makes it extremely important in designing a product.

4. Customers and end-users are extremely important for an agile process team. These users provide feedback on what the use cases are for a product. They also can provide information on how well the product is working after each incremental delivery. While the product should be well tested before delivery, the customer can usually find bugs in earlier developments of the product allowing the dev team to go back in and polish information.
5. Requirements engineering should be an iterative process because each step builds off another. For example, when you create your use cases in the written requirements section, that is then used to develop your Use Case Diagram and the sequence diagrams. Each part of requirements engineering needs to be completed before moving onto the next part.
6. The first quality a software engineer should poses is creativity. Being creative is a big part of problem solving and being able to think outside the box can help to find better solutions. The next quality is good teamworking skills. Almost all software engineers are going to need to work in teams, so it is extremely important to be able to communicate well. The third essential quality is being diligent. A big part of the software engineering process is delivering the product in many steps. Therefore, a software engineer needs to be diligent and able to get the work in on time. The fourth quality is technical skills. Knowing a lot about computer science and the different aspects of it can help to provide insight to possible solutions. Finally, the last characteristic is a good business background. Having a good business background is essential to being able to communicate with the costumer and know how to make deals or describe the product in a way they can understand.
7. Nonfunctional requirements are important because they define a set of standards to be followed in the software system. This ensures the product is usable and effective. Essentially nonfunctional requirements ensure the product is created to the highest standard.

8. The most significant result of the written requirements engineering process is the requirements section and more specifically the use cases. These cases depict all of the functions of the program and allow for an easier development process.
9. The UML diagram that is most useful in scenario-based modeling is the use-case diagram. The use case diagram depicts all the actions a user can perform in the software. This shows how the software is supposed to be used from the user's point of view. The use case diagram depicts all the scenarios possible in the program. Another diagram is the Activity Diagram. This depicts the activities throughout using the program.
10. The most important use of Use Cases going from written requirements to analysis and modeling is for the sequence diagrams. Sequence diagrams break down each individual use case and depict how it functions in the software. These diagrams are extremely important to designing each component of the software and satisfying all the requirements.