NAME: JESSE ABIGAIL VICTOR

MATRIC: HCSF/23/0043

COURSE: INTRODUCTION TO SOFTWARE ENGINEERING

CODE: COM 324

Level: HND 1

**QUESTION 1:**

WHAT IS SOFTWARE ENGINEERING

SOFTWARE ENGINEERING is the branch of computer science that deals with the **design, development, testing, and maintenance of software applications**. Software engineers apply engineering principles and knowledge of programming languages to build software solutions for end users.

Software engineers design and develop computer games, business applications, operating systems, network control systems, and middleware—to name just a few of the many career paths available.

KEY ASPECT OF SOFTWARE ENGINEERING:

1. **REQUIREMENT ANALYSIS**: understanding and documenting what the software is supposed to do by gathering requirement and stakeholders.
2. **DESIGN**: planning the structure and components of the software system, including the architecture, interfaces, and data models.
3. **IMPLEMENTATION**: writing the actual code according to the design specifications.
4. **Testing**: systematically checking the software for errors, bugs and inconsistencies to ensure its meets its requirements.
5. **MAINTENANCE**: updating and refining the software after it is deployed, fixing bugs, and making improvement based on user feedbacks.
6. **PROJECT MANAGEMENT**: planning, scheduling, and managing resources to ensure the software project is completed on time and within budget.
7. **QUALITY ASSURANCE**: ensuring that the software meets quality standards and performs as expected.
8. **DOCUMENTATION:** creating detailed documentation for the code, design and the usage of the software to facilitate maintenance and future development.

Software engineering aims to build b software that is reliable, maintainable, and scalable often using established methodologies like agile, waterfall or DevOps to guide the software.

**QUESTION 2.**

LIST THE NEEDS FOR SOFTWARE ENGINEERING.

1. SYSTEMATIC APPROACH: to manage the complicity of software project through structured methodologies and best practices.
2. CLEAR REQUIREMENT: to understand what the software need to do, which help in setting objectives and guiding development efforts.
3. DESIGN AND ARCHITECTURE: to create a blue print for the software, ensuring that this modular, scalable and maintainable.
4. EFFICIENT CODING PRACTICES: to provide high quality code that is readable, and adheres to standards.
5. TESTING AND VALIDATION: to identify and fix bugs, verify that the software meets its requirement, and ensure its performs well under different conditions.
6. PROJECT MANAGEMENT: to plan, track and manage development process, resources, and timeline to deliver the project successfully.
7. QUALITY ASSURANCE: to ensure the software meet quality standards and it’s free from critical defects.
8. DOCUMENTATION: to provide clear and comprehensive information about the software design, code and usage with aids in future maintenance and development.
9. MAINTENANCE AND SUPPORT: to handles issues that arises after deployment, make updates, and adapt the software to changing needs.
10. User training and support: to help users understand and effectively use the software, ensuring that it’s meet their needs and expectations.

**QUESTION 3**

EXPLAIN THE 5 MAJOR STEPS IN SOFTWARE DEVELOPMENT LIFE CYCLE

1. **Requirement Analysis**

* Purpose; understand and define what the software needs to do.
* Activities: gather requirement from stakeholders, document functionality and system expectations.
* Outcome: a clear requirements specification.

2. **Design**

* **Purpose:** Plan the architecture and layout of the system.
* **Activities:** Create system architecture, UI/UX design, and data model
* **Outcome:** A detailed design blueprint guiding the development

3**. Implementation (coding)**

* **Purpose:** Translate the design into a working software product
* **Activities:** Write and compile code, develop software components
* **Outcome:** Functional software based on design

4**. Testing**

* **Purpose:** Verify that the software is free of defects and meets requirement
* **Activities:** Performs various tests (unit, integration, system), identify and fix bugs.
* **Outcome:** A thoroughly tested product ready for deployment

5. **Deployment and Maintenance**

* **Purpose:** Release the software and ensure its working properly long-term
* **Activities:** Deploy to production, monitor, provide update, and handle bug fixes
* **Outcome:** A live, maintained software product

**QUESTION 4**

List 5 tools needed at each step in question 3

1. **Requirement Analysis:**

* Jira,
* Lucid chart,
* Confluence,
* Visio,
* IBM DOORS.

1. **Design:**

* Enterprise Architect,
* StarUML,
* Balsamiq,
* Adobe XD
* Lucid chart.

1. **Implantation:**

* VS Code
* Git/GitHub
* Intelij IDEA
* Docker
* Jenkins.

1. **Testing:**

* Selenium
* Junit
* TestRail
* Postman
* JIRA.

1. **Deployment and, Maintenance**:

* Jenkins,
* Kubernetes,
* Ansible,
* New Relic,
* Aws Cloud formation.

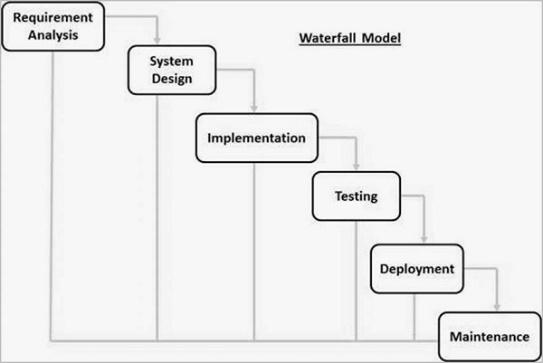
**QUESTION 5**

List 5 Software Development Methodologies

1. Waterfall
2. Lean development
3. Agile
4. Scrum
5. DevOPs

**QUESTION 6**

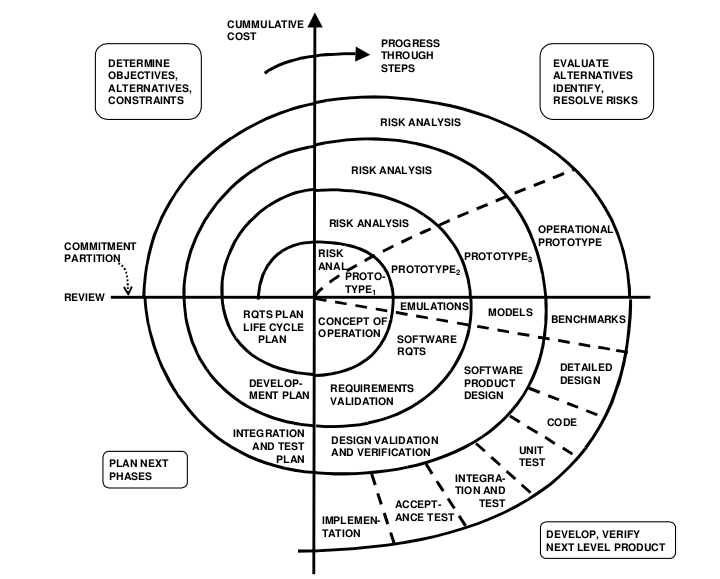
Draw And Explain The Waterfall Model, V Model And Spiral Model



The Waterfall Model is a linear application development model that uses rigid phases: When one phases ends, the next begins. Steps occur in sequence, and, if unmodified, the model does not allow developers to go back to previous steps (hence “waterfall”: Once water falls down, it cannot go back up).



The V-model is a graphical representation of a systems development lifecycle. It is used to produce rigorous development lifecycle models and project management models. The V-model falls into three broad categories, the German V-Modell, a general testing model, and the US government standard.



The spiral model is a systems development lifecycle (SDLC) method used for risk management that combines the iterative development process model with element of the waterfall model. The spiral model is used by software engineers and is favored for large expensive and complicated project

**QUESTION 7**

Explain Functional and Non Functional Requirements of a Software, Give Example

FUNCTIONAL REQUIREMENTS

Functional requirement define how a system must work, what the system should do, specifying tasks and behaviors.

* **Examples:** user authentication, payment processing, search functionality, report generation and notifications.

NON FUNCTIONAL REQUIREMENT

Nonfunctional requirement describe how the system performs, detailed how it should performs, focusing on qualities like performance, security and usability.

* **Example**: Common types or categories of non-functional requirements (NFRs) include security, capacity, compatibility, reliability, and more

**QUESTION 8**

Explain 3 types of software testing

1. **UNIT TESTING**

* Purpose: test individual component or functions of the software to ensure the work as expected.
* Focus: isolated unit of code, typically functions or method
* Examples: testing a function that calculate the total price of items in a shopping cart.

2. **INTEGRATION TESTING**

* Purpose: ensure that different components or modules of the software work together correctly.
* Focus: interaction between integrated components
* Examples: testing the interaction between a login module and a user profile module.

3. **SYSTEM TESTING**

* Purpose: test the complete, integrated system to ensure its meets all specific requirements.
* Focus: the entire system validating both functional and nonfunctional aspect
* Examples: testing the web application to ensure all features (login, search, and checkout) work as a whole.

Question 9

What is software case tools?

A computer-aided software engineering (CASE) tool is a software package that provides support for the design and implementation of information systems. It can document a database design and provide invaluable help in maintaining the consistency of a design.

**Examples are:**

1. Design and modeling
2. Coding and development
3. Testing and quality assurance
4. Project management
5. Integration and build

**Question 10**

Mention 5 types of software case tools

1. Diagramming tools: used for creating system models and flowcharts (e.g. Microsoft Visio).

2. Code generational tools: automatically generate code from design models (e.g.., Altona UModel).

3. Project management tools: Assist in planning, tracking and managing project (e.g., JIRA)

4. Testing tools: help automate software testing processes (e.g., selenium).

5> version control tools manage and track changes in code (e.g., Git)