1. Избройте поне три линейни последователни модели на процеса на разработка на софтуер?  
- Водопаден модел

- V- Model

- CleanRoom Engineering

2. Каква е разликата между линейните последователни и непоследователните модели на процеса на разработка на софтуер?  
- Последователни модели (sequential life-cycle models) - при тези модели се спазва една сравнително строга последователност на дейностите (фазите) при разработване на софтуер

- Инеративни модели (nonsequential life-cycle models) - тези модели (наричат се още циклични модели) допускат връщане (feedbacks) към предишни дейности (фази).

3. Каква е разликата между инкременталния развой и гъвкавите модели за разработка на софтуер?  
- Инкременталният развой е непоследователен тип.

- При гъвкавите модели се следват принципи. Доставят се работещи версии на продукта на няколко седмици или месец; Клиентите и разработчиците работят заедно по време на целия проект;

4. Колко и кои са нивата на зрелост в CMM?  
- Ниво 1 – начално

- Ниво 2- повторяемо

- Ниво 3- определено

- Ниво 4- управляемо

- Ниво 5- оптимизиращо

5. Кои са двете характеристики, с които се определя софтуерния риск?  
- несигурност –няма 100% вероятен риск

- загуба –ако риска стане реален ще се появят нежелани последствия и загуби.

6. Опишете концепцията за работа със софтуерния риск?  
- Избягване на риска- Разработва се стратегия за намаляване на реорганизацията в персонала.

- Наблюдение на риска - Риска трябва да се наблюдава и да се прецени каква е в даден момент неговата вероятност.

- Управление на риска и планиране на евентуални случаи

7. Избройте видовете валидационни тестове?  
Alpha тестове и Beta тестове

8. Какви тестове се провеждат при тестване на най-малките компоненти на системата (unit testing)?  
- Interface

- Local data structures

- Boundary conditions

- Independent paths

- Error handling paths

9. Опишете структурата на модела за качество на типичния йерархичен модел?  
- Ниво 0 – Качество

- Ниво 1 - фактори, представящи даден аспект на качеството на софтуера от гледна точка на потребителя.

- Ниво 2 - критерии – представят характеристики на програмния продукт.

- Ниво 3 - метрики –детайли на даден критерии. Те се определят от оценъчните елементи.

- Ниво 4 – оценъчни елементи – елементарни характеристики на най-ниско ниво, коитоподлежат на количествена оценка.

10. Кои са SOLID принципите избройте ги и ги обяснете?  
- Single Responsibility Principle (SRP). „Класът трябва да има една единствена отговорност.“

- Open-Closed Principle (OCP). “Модулът (компонента) трябва да бъде отворен за разширения, но затворен за модификации.“

- Liskov Substitution Principle (LSP). “Подкласовете трябва да бъдат заменими с техните базови класове.“

- Interface Segregation Principle (ISP).“Много специфични клиентски интерфейси са по-добре от един с обща цел.“

- Dependency Inversion Principle (DIP). “Зависимост от абстракции. Не зависимост от конкретики.”

11. Кои са стъпките в процеса на валидация на UI?  
- Дефиниране на интерфейсните обекти и действия (операции).

- Дефиниране на събития (действия на потребителя)

- Описание на всяко състояние на интерфейса

- Определете как потребителя интерпретира състоянието на системата от информацията предоставена през интерфейса.

12. Избройте някои от аспектите на добрия стил на програмиране (поне 4 от общо 9)?  
- коментари

- структорно програмиране

- производителност

- цялостна документация

- форматиране

- странични ефекти

- сигурни програми

- преносими програми

- избор на имена

13. Кои са критериите за качество на една софтуерна архитектура?  
- Структурираност

- Простота

- Нагледност

14. Как от ООА модел преминаваме към ООП модел?  
- Като се разшири ООА с класове от потребителския

интерфейс.

15. Има ли възможност в SCRUM да се добави проектирането като стъпка – кога и как? - Да

16.Какви модели се създават по време на ООА?  
Базов, статичен и динамичен модел.

17.Какво включва извличането на изискванията при ООА?  
Демек?

18.Клас диаграмата към кой модел на ООА принадлежи?  
- Статичния

19.Какво представлява CRC модела?  
Class-responsibility-collaborator моделирането предоставя прост начин за идентифициране и организиране на класовете, които съответстват на системата или изискванията към нея.  
CRC модел реално е колекция от стандартни индекс карти, които представят класовете. Картите са разделени на три секции. В най-горната секция записвате името на класа. В тялото на картата изброявате отговорностите на класа в лявата страна и сътрудничества в дясната.

20.Каква е разликата между Sequence диаграми и Collaboration диаграми?  
Sequence диаграмата позволява моделиране на времето на живот на обекта, а Collabroation диаграмата позволява моделиране на структурните връзки между обектите по време на тяхното взаимодействие.  
Разликата между Sequance и collaboration диаграма е в това, че в Sequance по вертикала се изобразява протичане на времето.

21.Какви са двата принципа за разработка на interaction диаграмите?  
Sequance и Collaboration

22.Какви диаграми се използват за представяне на жизнения цикъл на обектите от един клас?  
- Крайни автомати

23.Един от документите на фазата „Анализ и дефиниция“ е речник. Какво е неговото предназначение?  
Речникът дефинира общата терминология, бива използван за потребителския интерфейс, help и ръководствтвото за потребителя.

24.Защо фазата на „Анализ и дефиниция“ се разделя на две подфази?  
Разделянето на две фази е нужно, защото може да се стигне до неразбирателство с клиента още след първата подфаза - планирането(проучването), и съответно да се прекрати проекта. Ако не се раздели на две части и се окаже, че клиента не е съгласен, или че разработчиците не са съгласни и съответно проекта пропадне, това би било много изгубено време.

25.На какви принципи трябва да отговарят основните концепции?  
WHAT DAFUQ?

26.Кой е основният недостатък на COCOMO (Constructive Cost Model)?  
Основният недостатък на COCOMO е, че се базира на редове първичен код. Поради това то не е подходящ за Обектно-ориентирани проекти.  
Трудната му ранна определеност и липсата на единно виждане за ред първичен код.  
Броя редове код.  
Няма ясна структурираност, съсредоточава се върху качеството на кода, а не върху целия продукт; тясна експериментална база.; не е подходящ за ОО.

27.Кои са двата критерия, които влияят при определяне на цената на софтуера?  
Функционалност и време

28.Кои са основните концепции на функционалния изглед на системата?  
Функционални дървета, Диаграми на потока от данни и Use case диаграми (бизнес процес)

29.Какви видове връзки има в ERD?  
Семантични връзки между множества отидентичности.

30.4.Кое от изброените не е принцип на доброто писане на код:

А) Създаване на unit тестове преди писане на кода;

Б) Създаване на визуално представяне на кода, за да бъде разбран;

В) Прилагане на refactoring след написване на кода;

Г) Всички тестове трябва да бъдат проследими до клиентските изисквания

Фазата „Анализ и дефиниция“ цели да анализира проблема и да определи изискванията към софтуера.

Фазата се разделя на две части:  
1. Фаза на планиране и 2. Фаза на дефиницията.  
Разделянето на две фази е нужно, защото може да се стигне до неразбирателство с клиента още след първата подфаза - планирането(проучването), и съответно да се прекрати проекта. Ако не се раздели на две части и се окаже, че клиента не е съгласен, или че разработчиците не са съгласни и съответно проекта пропадне, това би било много изгубено време.

**От двете подфази трябва да получим по 4 документа (4+4 = 8 документа):**  
**От фаза на планиране**: 1.Речник, 2.Предварителна спецификация на изискванията, 3.Определяне на цената, 4.План на проекта.  
**От фаза на дефиниция**: 1.Спецификация на изискванията (описателна), 2.Модел на продукта (основни методи, формализирано описание), 3.Потребителски интерфейс (концептия и/или прототип), 4.Ръководство за потребителя (предварително)

***Фаза планиране:*  
Речника и предварителната спецификация на изискванията** са отговорност на **приложния специалист**, а клиентът и ръководителят сътрудничат.  
**Определянето на цената и планът на проекта** са отговорност на **ръководителя на проекта**, а клиентът и приложният специалист сътрудничат.  
Речникът дефинира общата терминология, използва се за потребителския интерфейст, HELP? И ръководството за потребителя.

***Фаза дефиниция:***  
**Спецификацията на изискванията, потребителският интерфейс и ръководството за потребителя** са отговорност на **приложния специалист**, а клиентът, ръководителят и системният анализатор сътрудничат.  
**Моделът на продукта** е отговорност на **системния анализатор**, а клиентът, ръководителят и приложният специалист сътрудничат.

**Спецификацията на изискванията често се променя поради:**  
 **\***Грешки, неточности, недоразумения и пр.  
 **\***Промяна на изискванията по време на проекта  
 **\***Необходимост от различна структура на документа

\*Основа: софтуерните документи (напр. Спецификацията на изискванията) трябва да бъдат прецизни както програмата.  
\*Метод (преглед): проверка на документите от група оценители, чрез внимателно четене на документите (бележка: 2-5 оценителя + автора на документа)  
\*Процес: 1.Определяне на участниците, 2.Среща за преглед на гру..., 3.Разработване на протокол на прегледа

***Предварителна спецификация на изискванията:***  
1.основни функции на продукта, 2.основни данни, 3.обща производителност, 4.важни аспекти на потребителския интерфейс, 5.важни критерии за качество  
Този документ е ориентиран към потребителя и трябва да бъде четен и разбран от него, т.е. той трябва да бъде написан в терминологията на потребителя (не в тези на разработчика).

***Съдържание на спецификацията на изискванията*** – словесно описание на изискванията към продукта:  
1.**Функционални изисквания**: функционалност, данни (логически изглед), потребителски интерфейс  
2.**Изисквания на средата на приложението:** например потребителски профил  
3.**Технически изисквания**: език за разработка, операционна система, хардуер  
4.**Изисквания за производителност**: производителност, количество данни  
5.**Изисквания за валидност**: разработване на тестове, тестови случаи  
6.**Изисквания за качество**: удобен за използване интерфейст, надеждност...  
7.**Изисквания за реализация**: модел на процеса, документация, срокове, цена

---------------------------------------------------------------------------------------------------------------  
Съдържание на спецификацията на изискванията:  
1.Въведение  
 1.1 Цел  
 1.2 Област  
 1.3 Дефиниции, акроними и съкращения  
 1.4 Референции  
 1.5 Общ преглед  
2.Общо описание  
 2.1 Изглед на продукта  
 2.2 Функции на продукта  
 2.3 Потребителски характеристики  
 2.4 Ограничения  
 2.5 Допускания и зависимости  
3. Специфични изисквания  
 \*Има няколко различни организации (подхода) в тази секция, зависещи от приложението.  
 \*Независимо от избраната организация, тази секция трябва да съдържа следната информация:  
 - изисквания за външния интерфейс  
 - функционални изисквания  
 - изисквания за производителност  
 - проектни ограничения  
 - критерии за качество  
 - други изисквания  
---------------------------------------------------------------------------------------------------------------

Общо представяне на продукта – диаграма на бизнес процес (use-case диаграма):  
\* Именуване на основните функции и \*Дефиниране на правата на достъп на актьорите

7 изисквания за качество:  
Функционалност – удобство, прецизност, възможност за интеграция, съгласуваност, сигурност  
Надеждност – зрелост, допустими грешки, възстановимост  
Използваемост – разбираемост, разучаемост, оперируемост  
Ефективност – поведение към време, поведение към ресурси  
Възможност за поддръжка – възможност за анализ, възможност за промяна, стабилност, възможност за тестване  
Преносимост

-----------------------------------

**Речник**- специфициране по-точно на понятията, описани в началните изисквания; изясняват кой ще ползва разработеното приложение

**Модел на продукта**- съответства на типа на анализ- структурен или ОО.

**Цена на софтуера:**   
**КОКОМО- цели**: формулира множество от критерии, които следва да удовлетворява даден модел за установяване цената на разработване на софтуерни продукти. Основната му цел е за всеки планиран софтуерен проект да се оцени цената и срокът на разработване. Основополагащата му идея е използването на броя редове първичен код.;  
**Критерии на Боем**- определеност, точност, обективност, детайлност, устойчивост, областна приложение, конструктивност, простота на прилагане, предсказуемост, икономичност.   
**Недостатъци**- трудната му ранна определеност и липсата на единно виждане за ред първичен код

**ФМП**- усилията за разработката на даден софтуер се определят от неговата функционалност;

**Структурен анализ**- обозначава един клас от методи, подходящи за структурно програмиранe. Декомпозиране до мини спецификации

**Контекстни диаграми**- описват интерфейсите на моделираната система с околната среда. ; идея- да се моделира трансформацията на цялостното поведение на разработваната система, като:

• Системата се представя като черна кутия

• Явно се представя входното/изходното поведение

• Представя се трансформацията на интерфейсната информация.

**Мини спецификации**- псевдо код, правила, таблици и дървета на решенията