**1. Яка різниця між bug, failure, error?**

*Bug (defect)* – помилка програміста (або дизайнера або ще кого, хто бере участь у розробці), тобто коли в програмі щось йде не так як планувалося і програма виходить з-під контролю. Наприклад, коли відсутня валідація полів і в результаті неправильні дані викликають краші або інші збої у роботі програми. Або всередині програма побудована так, що від самого початку не відповідає тому, що від неї очікується.

*Failure* – збій (причому не обов'язково апаратний) у роботі компонента, всієї програми чи системи. Збій у роботі програми може бути індикатором наявності дефекту.

*Error* – помилка користувача, який намагається використовувати програму іншим способом. Наприклад, вводить літери в поля, де потрібно вводити цифри (вік, кількість товару тощо). У якісній програмі передбачені такі ситуації та видаються повідомлення про помилку.

**2. Яка різниця між поняттями QA та QC?**

*QA (Quality Assurance)* – забезпечення якості продукту, що включає такі процеси:

* перевірка технічних характеристик і вимог до ПЗ;
* оцінка ризиків;
* планування завдань для поліпшення якості продукції;
* підготовка документації, тестового оточення і даних;
* тестування;
* аналіз результатів тестування, а також складання звітів та інших документів.

*QC (Quality Control)* – контроль якості продукту, тобто:

* перевірка готовності ПЗ до релізу;
* перевірка відповідності вимог і якості даного проєкту.

**3. Що таке функціональне та нефункціональне тестування.**

Функціональне тестування – тестування всіх функцій системи для підтвердження, що кожна функція програми працює відповідно до документації.

У рамках функціонального тестування ми відповідаємо на питання «Чи працює система?», нефункціональне відповідає на питання: «Як добре працює система?». Нефункціональне тестування направлено на перевірку тих аспектів ПЗ, які можуть бути описані в документації, але не відносяться до функцій програмних продуктів.

Нефункціональне тестування складається з підвидів:

* тестування стабільності;
* юзабіліті тестування;
* приймальне тестування;
* тестування документації;
* тестування витривалості системи;
* тестування навантаження;
* тестування продуктивності;
* тестування сумісності;
* тестування безпеки;
* об'ємне тестування;
* стрес тестування;
* тестування швидкості відновлення;
* тестування локалізації, інтернаціоналізація та ін.

Також окремо виділяють структурне тестування та тестування змін (регресійне та повторне тестування).

**4. Різниця між Severity та Priority?**

Серйозність (Severity) – це атрибут, який охарактеризовує рівень впливу бага на загальну функціональність продукту, що тестується.

Пріоритет (Priority) – це атрибут, який визначає швидкість усунення бага.

*Високий пріоритет і низька серйозність*

Будь-які незначні дефекти серйозності, які можуть безпосередньо впливати на досвід користувача, автоматично переносяться в цю категорію. Сюди ж потрапляють баги, які мали бути виправлені, але які не впливають на додаток.

*Висока серйозність і низький пріоритет*

Дефект, який виникає у функціональності додатку (для якого немає обхідного шляху) і не дозволяє користувачу використовувати систему, але рідко використовується кінцевим користувачем.

**5. Суть і різниця між валідацією і верифікацією.**

Верифікація (Verification) – підтвердження того, що певні вимоги були виконані. При верифікації ми задаємо питання “Чи правильно створюється і тестується продукт?”

Валідація (Validation) – перевірка того, що продукт відповідає очікуванням і потребам користувачів. При валідації – “Чи створюємо ми правильний продукт?”

**6. В чому різниця та суть тестової документації: тест-плану, чекліста та тест-кейсу?**

*Тест-план (Test Plan)* – документ, що описує цілі, підходи, ресурси і графік запланованих тестових активностей. Він визначає об'єкти тестування, властивості для тестування, завдання, відповідальних за завдання, ступінь незалежності кожного тестувальника, тестове оточення, метод проєктування тестів, визначає використовувані критерії входу і критерії виходу і причини їх вибору, а також будь-які ризики, що вимагають планування на випадок надзвичайних обставин.

*Чекліст (Check-list)* – контрольний список, який містить ряд необхідних перевірок для тестування.

*Тест-кейс (Test Case)* – це сукупність кроків, конкретних умов та параметрів, необхідних для перевірки реалізації тестованої функції або її частини.

**7. Основні атрибути баг-репорту та тест-кейсу?**

**Баг-репорт (Bug Report)** – це технічний документ, що описує ситуацію або послідовність дій, що призвела до некоректної роботи об’єкта тестування, з вказанням причин і очікуваного результату.

*Summary (Тема)* – це короткий опис бага.

*Description (Детальний опис)*– більш широкий опис суті бага, якщо є така необхідність.

*Steps To Reproduce (Кроки для відтворення)* – описані всі кроки, щоб можна було у точності відтворити помилку.

*Actual result (Фактичний результат)* – вказується, що працює не так, в якому місці продукту і за яких умов.

*Expected result (Очікуваний результат)* – вказується, як саме має працювати система на думку тестувальника.

*Attachments (Вкладення)*. Це може бути, наприклад, скріншот, відео або лог-файл.

*Priority (Пріорітет дефекта).*

*Severity (Серйозність дефекта).*

*Status (Статус)* – основний атрибут, що визначає поточний стан бага.

Оточення дефекта:*Component (Компонент)* або *Environment (Оточення)* – це атрибут дефекту, який вказує на якій платформі цей дефект відтворюється (iOS, Android, Windows, Mac і їх версії, назви і версії браузерів, в яких відтворюється дефект).

**Тест-кейс (Test Case)** – це сукупність кроків, конкретних умов і параметрів, необхідних для перевірки реалізації функції, що тестується або її частини.

*Test Case ID (Порядковий номер).*

*Test Summary (Тема)* – ім’я, короткий зміст чи назва.

*Preconditions (Передумови)*– список усіх необхідних підготовчих дій (налаштування програми, середовища тестування) для виконання даного тест-кейса.

*Test Steps (Кроки до відтворення)* – кроки, необхідні для проходження тесту.

*Expected result (Очікуваний результат)* – очікувана поведінка системи на вказані у кроках дії.

**8. Назвіть характеристики хорошого тест-кейсу?**

Хороший тест-кейс покриває як позитивні, так і негативні сценарії і виконує тільки одну дію за один раз та не перетинається з іншими.

Характеристики хорошого тесту-кейсу:

* набір тестів не повинен бути надмірним – повинен бути мінімальним і достатнім; рекомендовано використовувати розбиття на класи еквівалентності;
* тест-кейс не повинен бути надто простим чи, навпаки, складним;
* тест-кейс повинен бути точно визначеним;
* тест-кейс повинен бути уніфікованим;
* тест-кейс повинен бути однозначним і зрозумілим.

**9. Що таке Use Case/User Stories?**

*Use Case (Сценарій користування)* – це перелік дій, сценарій, за яким користувач взаємодіє з додатком або програмою для виконання будь-якої дії та досягнення конкретної мети.

Детальніше в статті: «[Що таке use case та для чого вони потрібні](https://training.qatestlab.com/blog/technical-articles/what-is-a-use-case-and-what-are-they-for/)»

*User Story (Історія користувача)* – це неформальне загальне пояснення функцій програмного забезпечення, написане з точки зору кінцевого користувача. Її мета полягає в тому, щоб сформулювати, яку цінність для замовника несе функціонал програмного забезпечення.

**10. У чому різниця між повторним та регресійним тестуванням?**

*Регресійне тестування (Regression testing)* – проводиться з метою перевірки працездатності функціоналу, що існує, та перевірки на відсутність сторонніх помилок після оновлення білда (внесення правок або доповнень в систему). Виконується тільки при додаванні нової фічі (додаткова функціональність ПЗ) або істотній зміні функціоналу системи.

*Повторне тестування (Retesting)* – проводиться для підтвердження виправлення помилки та роботи даного функціоналу. Ретест виконується в тому ж оточенні й з тими ж даними, але на новому білді.

**11. Життєвий цикл дефекту**

Життєвий цикл дефекту (Defect Lifecycle) - це послідовність етапів, які проходить дефект на своєму шляху з моменту його створення до остаточного закриття. Для простоти сприйняття зображується у вигляді схеми з можливими статусами і діями, які призводять до зміни цих статусів.

*Новий (New)* – звіт про дефект заводиться в баг-трекінгову систему вперше.

*Призначено (Assigned)* – звіт про дефект призначається на відповідного розробника/програміста.

*Відкрито (Open)* – розробник бере звіт про дефект в роботу для аналізу і виправлення.

*Виправлений (Fixed)* – розробник зробив необхідні зміни в коді і перевірив ці зміни сам. Звіт про дефект з цим статусом повертається назад тестувальнику.

Повторне тестування в режимі очікування – після виправлення дефекту розробник надав конкретний код для повторного тестування тестувальником. Тестування знаходиться на розгляді у тестувальника.

*Повторне тестування (Re-testing)* – на цій стадії тестувальник виконує повторне тестування зміненого коду, який був наданий розробником, для перевірки, виправлений дефект чи ні.

*Перевірений (Verified)* – якщо дефект не відтворюється, тестувальник підтверджує, що цей дефект виправлений.

*Перевідкритий (Reopened)* – якщо дефект все ж відтворюється, навіть після його виправлення розробником, тестувальник перевідкриває його і призначає на розробника. Цей дефект проходить через життєвий цикл дефекту ще раз.

*Закрито (Closed)*– якщо тестувальник впевнений, що дефект більше не відтворюється, то він його закриває. Цей статус означає, що дефект виправлений, протестований і схвалений.

*Дублікат (Duplicate)* – якщо дефект повторюється двічі або є два бага, які є наслідком однієї причини, то одному з них присвоюється даний статус.

*Відхилений (Rejected)* – якщо розробник вважає, що цей дефект не є обгрунтованим або вагомим, і дефект не буде розглядатися для виправлення або реалізації, він його відхиляє.

*Відтермінований (Deferred)* – очікується, що дефект, якому привласнили такий статус, буде виправлений в наступних версіях. Причин для присвоєння цього статусу може бути кілька: пріоритет дефекту низький, нестача часу, даний дефект не спричинить великих збоїв в програмному продукті.

*Не баг (Not a bug)* – цей статус присвоюється, якщо в функціонал програми не буде внесено ніяких змін. Наприклад, якщо замовник просить змінити колір або розмір кнопок, або тексту – це не дефект, а просто зміни в дизайні додатка.

**12. Що таке позитивне і негативне тестування?**

*Позитивне тестування* – це тестування із застосуванням сценаріїв, які відповідають нормальній (штатній, очікуваній) поведінці системи. З його допомогою ми можемо визначити, що система робить саме те, для чого і була створена.

*Негативним* називають тестування, в рамках якого застосовуються сценарії, які відповідають позаштатній поведінці тестованої системи. Це можуть бути, наприклад, виняткові ситуації або невірні дані.

**13. Дайте визначення тестуванню білого і чорного ящика.**

*Тестування чорного ящика (Black box testing)* – спеціальний метод перевірки працездатності програмного забезпечення, у якому вся функціональність товару досліджується без аналізу вихідного коду. Тестувальники створюють логічно зрозумілі тест-кейси, спираючись виключно на вимоги специфікації на проєкті.

Наприклад, тестувальник проводить тестування веб-сайту, не знаючи особливостей його реалізації, використовуючи лише передбачені розробником поля введення та кнопки. Джерело очікуваного результату – специфікація.

*Тестування білого ящика (White box testing)* – особливий метод перевірки програмного забезпечення, який передбачає, що внутрішня структура та технічні особливості програмного забезпечення досконально відомі тестувальнику.

Наприклад, тестувальник, який, як правило, є програмістом, вивчає реалізацію коду поля введення на веб-сторінці, визначає всі передбачені (як правильні, так і неправильні) і не передбачені введення користувача, і порівнює фактичний результат виконання програми з очікуваним. При цьому очікуваний результат визначається тим, як повинен працювати код програми.

**14. Що таке SDLC, STLC?**

*Життєвий цикл програмного забезпечення (Software Development Life Cycle)* – період часу, який починається з моменту прийняття рішення про необхідність створення програмного продукту і закінчується в момент його повного вилучення з експлуатації.

Детальніше в статті: «[Популярні життєві цикли розробки ПЗ](https://training.qatestlab.com/blog/technical-articles/popular-software-development-life-cycles/)»

*Етапи тестування ПЗ (Software Testing Life Cycle)* – це процес, який складається з певної кількості окремих частин, має кінцеву мету і виконується протягом усього життєвого циклу розробки програмного забезпечення. До основних етапів тестування відносять: аналіз вимог, процес тест-дизайну, розробка, тестування та дебагінг, експлуатація та підтримка.

**15. Що таке тест-дизайн? Основні техніки тест-дизайну.**

*Тест-дизайн (Test Design)* – це один із початкових етапів процесу тестування ПЗ, на якому плануються і проєктуються тестові випадки (тест-кейси) відповідно до критеріїв якості, вимог до проєкту і цілей тестування. Головною метою тест-дизайну є покриття тестами всього функціоналу, використовуючи при цьому мінімальну кількість тестів. Для того, щоб досягти зазначеної мети, застосовують різні техніки тест-дизайну – загальні правила і рекомендації щодо створення тестів при проведенні тестування.

Техніки тест-дизайну: Аналіз граничних значень (Boundary Value Analysis), Вичерпне тестування (Exhaustive Testing), Еквівалентний розподіл (Equivalence Partitioning), Передбачення помилки (Error Guessing), Попарне тестування (Pairwise Testing), Причина/Наслідок (Cause/Effect), Таблиця прийняття рішень (Decision Table) та ін.

**16. Що таке кросбраузерне тестування?**

*Кросбраузерне тестування (Cross-browser testіng)* – це тестування з використанням різних браузерів з метою перевірки властивості сайту (веб-додатку) правильно відображатися та функціонувати в усіх браузерах. Це пов'язано з тим, що будь-який з браузерів має власні надбудови, плагіни, а також відмінності в десктопній і мобільній версіях.

**17. Які ви можете назвати особливості тестування додатків на мобільних пристроях?**

Основні моменти, на які необхідно звертати особливу увагу саме при тестуванні додатків на мобільних пристроях:

* розмір екрана та touch-інтерфейс,
* ресурси телефону,
* різні роздільні здатності екрану та версії ОС,
* реакція програми на зовнішні переривання;
* перевірка роботи оновлень,
* реклама в мобільному додатку,
* перевірка локалізації.

**18. Типи мобільних додатків. Поясніть різницю між Native, Web App, Hybrid App.**

*Нативні додатки (Native apps)* – мобільні додатки, написані мовами програмування, затвердженими розробниками програмного забезпечення під кожну конкретну платформу, а тому органічно вбудовуються в самі операційні системи. Додатки завантажуються через магазини програм (App Store, Google Play тощо) та відповідають вимогам цих магазинів. Нативні програми можуть повністю або частково працювати і за відсутності інтернет-з'єднання, тому користувачі менше залежать від якості зв'язку і можуть користуватися програмою там і тоді, коли їм це зручно.

*Мобільні веб-додатки (Web apps)* – це мобільна версія сайту лише з розширеним інтерактивом. Різниця між веб-додатком та адаптивною версткою сайту не велика, оскільки і там і там застосовуються стандартні веб-технології, а швидкість роботи обмежена якістю інтернет-з'єднання. При цьому веб-програми не розміщуються в спеціалізованих магазинах програм і зазвичай використовують браузер телефону для роботи.

*Гібридні додатки (Hybrid apps)* – Генератори мобільних додатків дозволяють створювати кросплатформні додатки, наближені за функціоналом та якістю до нативних додатків. Це щось середнє між нативними та веб-додатками. Такі програми встановлюються через офіційні магазини, мають обмежений доступ до апаратної частини смартфонів та планшетів, в них можна настроювати push-сповіщення. А також вони, як правило, дешевші за нативні програми.

**19. Що таке логи (log files)? Як зняти логи на мобільних пристроях?**

*Логи (лог-файли, log-files)* – це файли, які містять інформацію про роботу сервера або комп'ютера, в які записуються певні дії програми, а також всі дії користувача над нею.

Для зняття логів на Android-пристроях використовують такі інструменти, як Minimal ADB, Android Studio. Для зняття логів на iOS-пристроях використовують iTunes, iMazing, XCode (для MacOS).

**20. Перерахуйте відомі вам ігрові механіки та ігрові жанри.**

*Ігрова механіка* – це набір правил, який визначає характер взаємодії гравця та інтерфейсу. Наприклад: Досягнення, Механіка призначеної зустрічі, Уникнення, Поведінковий контраст, Поведінковий імпульс, Вдячна продуктивність, Механіка каскадної подачі інформації, Ланцюги подій, Товариські ігри, Обставини,  Відлік, Стримувальні фактори, Нескінченні ігри, Припинення, Механіка фіксованих проміжних винагород, Механіка фіксованих відносин винагород, Механіка проміжних винагород, Лотерея, Модифікатори, Конфіденційність, Прогресивна динаміка, Механіки співвідношення нагород, Реальний час і час з затримкою, Посилення, Розподіл фізичних благ, Статус, Механіка змінних проміжних винагород, Механіка змінних відносин винагород, Весело один раз, Весело завжди, Віртуальні предмети та ін.