1. **Покажіть розвиток інженерії і зокрема інженерії програмного забезпечення**

  
**2. Наведіть модель Code and fix та вкажіть недоліки**

Описати правила цього методу просто:

- отримуємо початкове розуміння потреб замовника;

- починаємо програмувати;

- коли щось буде готово, показуємо «це» замовнику;

- отримавши відгуки, виправляємо наш код;

- повторюємо цикл до повного задоволення замовника (або поки у нього не закінчаться гроші, терпіння ...)

Використовуючи Code-and-Fix, ми тільки пишемо код. Тут все дуже просто: немає необхідності що-небудь планувати, немає необхідності що-небудь документувати. Тому Code-and-Fix вимагає мінімальної кваліфікації розробників, відповідно, їм можна платити менше грошей.

Але, у всього є своя ціна. Як показав досвід, такий підхід дуже скоро приводить до коду, який неможливо підтримувати: виправлення однієї помилки призводить до появи декількох нових; внесення мінімальних змін в одну з частин програми, призводить до руйнування функцій, реалізованих іншими частинами.  
**3. Поясніть, в чому системо утворююча роль життєвого циклу программного забезпечення в інженерії**

**Життєвий цикл ПЗ** - це стадії, які проходить програмний продукт від появи ідеї до її реалізації в коді, імплементації в бізнес і подальшої підтримки. Моделі життєвого циклу багато в чому зумовлюють і методології розробки ПЗ.

**4. Наведіть Step wise модель, поясніть роль її в інженерії програмного  
забезпечення та вкажіть недоліки**

**Поступове вдосконалення** - це концептуальна основа для поступового розвитку артефакту. Відповідно до поетапного вдосконалення розвиток відбувається в трьох вимірах: від абстрактного до конкретного, від часткового до повного та від неструктурованого до структурованого. Таким чином, розвиток артефакту можна охарактеризувати як змішану послідовність доопрацювань, розширень та реструктуризацій артефакту.

**5. Основні складові фаз життєвого циклу, дайте визначення, наведіть приклади**

Кожна фаза містить три складових:

− процеси;

− продукти;

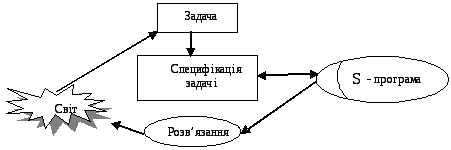
− ресурси.

На кожній фазі відбувається певна сукупність процесів, кожен з яких породжує певний продукт, використовуючи певні ресурси.

**6. Типи програм за Леманом, загальні визначення**

Існує підхід запропонований М. Леманом згідно з яким усі комп’ютерні програми можна поділити на три типи: S (Specification), P (Problem) і E (Еnvironment).

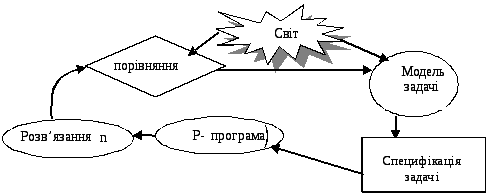
***S*-програма** – це така програма, функція якої відома й визначена однозначно специфікацією задачі. Для *S*-програм характерна повна визначеність вихідної задачі, вимог і значень, а тому S-програми після створення не змінюються. А якщо *S*-програма змінюється, то зміни не повинні порушити відповідності вхід/вихід, оскільки інакше вона розв’язуватиме іншу задачу, і це буде інша програма.



Місце S-програми в реальному світі

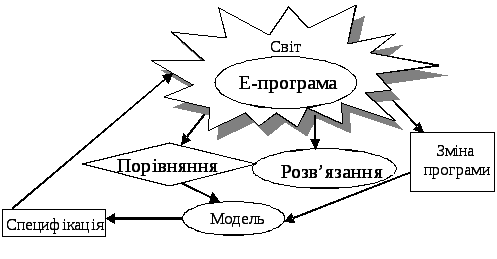
***Р*-програма** – це така програма яка розв’язує задачу, що не має точної постановки. Тому специфікація задачі та розв’язання наближені, уособлюючи абстрактну модель реальної ситуації, і після порівняння з вимогами реального світу уточнюватимуться через зміну програми. Проте це буде не нова, а стара програма.

Прикладом є програма що обчислює прогноз погоди. У такій програмі результати розрахунків порівнюються з реальними даними про погоду, а методи прогнозування вдосконалюються зі змінною самої програми.



Місце *Р*-програми в реальному світі

***Е*-програма** – це програма, яка розв’язує таку задачу, що потребує її присутності в контексті реального світу. У процесі використання *Е*-програми в реальному світі становлення до неї зазвичай змінюється і постає потреба змінити програму. При цьому змінена *Е*-програма, так само як і *Р*-програма, не буде новою програмою. Прикладом такої програми являється програма керування тренажером для реального об’єкту.



Місце *Е*-програми в реальному світі

Зазначені відмінності між типами програм є принциповими й можуть впливати на стосунки між замовником програми і виконавцем – розробником, вибір типу життєвого циклу або обсяг фінансування. Наприклад, формуючу стосунки із замовником розробник залежно від типу програми (у разі *Р*- і *Е*-програм) має передбачити у проекті неодмінні зміни програм.

Зазвичай Р і Е програми називають *програмами-застосуваннями*, або *комп’ютерними застосуваннями*, або *програмними системами.*

**7. Комп’ютерні програми як системи**

Програмна продукція, яка являє собою сукупність програм і (або) підсистем, що мають загальне цільове призначення. Зв'язок між програмами і (або) підсистемами встановлюється розробником, користувачем або іншими фахівцями

**8. Типи програмних систем**

На сьогоднішній день можна сказати, що більш-менш точно склалися такі групи програмного забезпечення:

* операційні системи та оболонки;
* системи програмування (транслятори, бібліотеки підпрограм, відлагоджувачі тощо);
* інструментальні системи;
* інтегровані пакети програм;
* динамічні електронні таблиці;
* системи машинної графіки;
* [системи управління базами даних (СУБД)](http://lib.mdpu.org.ua/e-book/vstup/glos.htm#%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0%20%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%96%D0%BD%D0%BD%D1%8F%20%D0%B1%D0%B0%D0%B7%D0%BE%D1%8E%20%D0%B4%D0%B0%D0%BD%D0%B8%D1%85);
* прикладне програмне забезпечення.

     Зрозуміло, цю класифікацію не можна вважати вичерпною, але вона більш-менш наочно відображає напрями удосконалення та розвитку програмного забезпечення.

**9. Вкажіть користувачів продуктів інженерії програмного забезпечення. Наведіть приклади продуктів інженерії програмного забезпечення -робочих продуктів.**

Програмні продукти можуть створюватися як:

- індивідуальна розробка під замовлення;

- розробка для масового поширення серед користувачів.

**10. Специфікування вимог, поняття, витоки**

– **процеси** зорієнтовано на формулювання та точний опис (специфікування) вимог, яким має відповідати програмне забезпечення з точки зору замовника;

– **продукти** – специфікації вимог;

– **ресурси** – мови специфікування вимог, діаграмери, інженери зі специфікуванням вимог, комунікатори для зв’язку із замовником.

**11. Проектування, поняття, витоки, результат**

– **процеси** зорієнтовано на створення архітектури та детального проекту програмного забезпечення згідно зі специфікаціями вимог;

– **продукти** – архітектурний та детальний проекти програми;

– **ресурси** – системи автоматизованого проектування, документатори, архітектори, системні програмісти.

**12. Конструювання, поняття, витоки, результат**

– **процеси** зорієнтовано на кодування програмного забезпечення згідно з детальним проектом і тестування її з метою виявлення та усунення наявних помилок;

– **продукти** – програмне забезпечення, що відповідає вимогам проекту та тести для її тестування;

– **ресурси** – засоби програмування та тестування, програмісти, тестери.

**13. Супроводження, поняття, витоки, результат**

– **процеси** – коригувальне, адаптувальне, удосконалювальне та відновлювальне супроводження. *Коригувальне супроводження* – це зміна програмного забезпечення з метою виправлення помилок, яких припустилися на попередніх фазах життєвого циклу. *Адаптувальне супроводження* – це зміна програмного забезпечення з метою пристосування (адаптації) його до змінених вимог замовника. *Удосконалювальне супроводження* – це зміна програмного забезпечення з метою поліпшення його характеристик (метод обчислень, ефективність функціонування, інтерфейс програми). *Відновлювальне супроводження* – це зміна програмного забезпечення з метою відновлення його працездатності або здобуття інформації про його будову та функціонування;

– **продукти** – супроводжуване програмне забезпечення;

– **ресурси** – вимірювачі, реструктуризатори, абстрактори, екстрактори, засоби програмування, програмісти, інженери із супроводження.

**14. Доменний аналіз, поняття**

– **процеси** зорієнтовано на аналіз доменної інформації (мається на увазі домен розробки програмного забезпечення) із метою виявлення архітектур, конструкцій, методів для використання їх у розроблювальному програмному забезпеченні;

– **продукти** – архітектури, компоненти, методи;

– **ресурси** – інструменти доменного аналізу, доменні експерти, інженери.

**15. Утилізація, поняття, витоки**

– **процеси** – це утилізація, що передбачає відновлення, переробку, повторне використання та знищення успадкованого програмного забезпечення. *Відновлення* – це подовження життєвого циклу програмного забезпечення завдяки відновленню її працездатності. *Переробка* – це „глибока” зміна програмного забезпечення (архітектури, функцій) або його міграція в іншу операційну систему чи мову програмування. *Повторне використання* – це створення і застосування на основі успадкованого програмного забезпечення таких компонентів, які можна використати в розробці нового програмного забезпечення. *Знищення*– це знищення невикористаного (неутилізованого) програмного забезпечення;

– **продукти** – відновлене або перероблене програмне забезпечення, повторно використовувані компоненти;

– **ресурси** – екстрактори, абстрактори, гранулятори, програмісти, експерти