**Контрольні запитання**

1. **Гіпервізори** — це програми, які дозволяють створювати та керувати віртуальними машинами (VM). Вони поділяються на два основних типи: гіпервізори типу 1 (bare-metal) і гіпервізори типу 2 (hosted). Ось їхні основні відмінності та сфера застосування:

### **Гіпервізор типу 1 (bare-metal)**

#### **Сфера застосування:**

* Використовується в корпоративних середовищах, великих дата-центрах та хмарних платформах.
* Ідеально підходить для роботи з великою кількістю віртуальних машин, де потрібна висока продуктивність і стабільність.

### **Гіпервізор типу 2 (hosted)**

#### **Сфера застосування:**

* Застосовується переважно на персональних комп'ютерах, для тестування та розробки програмного забезпечення, або для запуску кількох операційних систем на одному пристрої.
* Підходить для локальних робочих станцій та малих середовищ, де продуктивність не є критичним фактором.

### **Основні відмінності:**

1. **Рівень роботи**: Гіпервізор типу 1 працює напряму з апаратним забезпеченням, а типу 2 — через хост-операційну систему.
2. **Продуктивність**: Гіпервізор типу 1 забезпечує кращу продуктивність через відсутність проміжної хост-операційної системи.
3. **Сфера застосування**: Тип 1 використовується в серверних середовищах, а тип 2 — на персональних комп'ютерах для менш критичних завдань.

Обидва типи мають свої переваги в залежності від потреб користувача та ресурсів, які є в наявності.

**2. GNU General Public License (GNU GPL)** — це ліцензія на вільне програмне забезпечення, розроблена Річардом Столлманом у рамках проекту GNU. Основна мета цієї ліцензії полягає в захисті свободи користувачів програмного забезпечення, а також у гарантії, що програмне забезпечення залишається вільним і доступним для всіх.

**3. Програмне забезпечення з відкритим кодом** (Open Source Software, OSS) — це тип програмного забезпечення, вихідний код якого є відкритим і доступним для вивчення, зміни та розповсюдження. **Відкритий код дозволяє користувачам та розробникам вільно використовувати, адаптувати та вдосконалювати програму, що робить її більш гнучкою та сприяє розвитку спільноти.**

**4.** **Дистрибутив**  — це готовий набір програмного забезпечення, зазвичай операційної системи, який включає ядро, системні програми, утиліти та інші компоненти, необхідні для встановлення та використання операційної системи. Найчастіше цей термін використовується стосовно дистрибутивів Linux, хоча його можна застосовувати до будь-яких програмних продуктів.

**5. Задачі системного адміністрування, які можна реалізувати на базі ОС Linux?**

**1. Управління користувачами та групами**

### **2. Моніторинг та управління системними ресурсами**

### **3. Управління файлами та системами зберігання**

### **4. Мережеве адміністрування**

### **5. Управління серверами та службами**

### **6. Автоматизація завдань**

### **7. Безпека та управління доступом**

### **8. Віртуалізація та контейнеризація**

### **9. Журналювання та аудит**

**6. ОС Android та Linux пов'язані між собою тим , що** Android є операційною системою, побудованою на базі ядра Linux, але вона розроблена для мобільних пристроїв і має свої особливості. Хоча обидві системи базуються на тому ж самому ядрі, Android має інше користувацьке середовище, програмні інтерфейси та модель безпеки, яка оптимізована для смартфонів і планшетів.

Таким чином, Linux забезпечує основу для роботи Android, але сам Android є спеціалізованою операційною системою з власними компонентами, специфічними для мобільних пристроїв.

**7. Основні можливості Embedded Linux:**

1. Гнучкість і налаштованість
2. Відкритий вихідний код
3. Підтримка широкого спектру апаратних платформ
4. Система реального часу (Real-Time)
5. Безпека та мережеві можливості
6. Можливість інтеграції різноманітного ПЗ

### Сфери використання Embedded Linux:

1. Інтернет речей (IoT)
2. Автомобільна промисловість
3. Промислові системи управління:
4. Медичні пристрої:
5. Смарт-пристрої:
6. Маршрутизатори та комунікаційні пристрої:
7. Робототехніка:

Embedded Linux — це надійна і гнучка платформа, яка знаходить застосування у різних галузях: від споживчих електронних пристроїв до промислових і медичних систем. Його відкритий код, гнучкість у налаштуванні та можливість підтримки реального часу роблять його ідеальним рішенням для вбудованих систем.

**8. Як змінити тип завантаження Linux:**

#### **1. Для систем на основі systemd:**

Більшість сучасних дистрибутивів Linux використовують систему ініціалізації systemd, тому процес зміни типу завантаження відрізняється від старого підходу з використанням runlevels.

##### Перехід у текстовий режим (CLI):

Для завантаження системи в текстовому режимі (3 рівень запуску) можна скористатися командою:  
bash  
Копіювати код  
sudo systemctl set-default multi-user.target

* Це команда налаштовує multi-user.target, який є аналогом рівня 3 (CLI), за замовчуванням при завантаженні.

##### Перехід у графічний режим (GUI):

Для повернення до графічного режиму (5 рівень запуску):  
bash  
Копіювати код  
sudo systemctl set-default graphical.target

* Ця команда встановлює graphical.target, який відповідає за завантаження в графічному режимі.

##### Тимчасова зміна режиму:

Якщо потрібно тимчасово переключитися в текстовий режим без зміни налаштувань за замовчуванням:  
bash  
Копіювати код  
sudo systemctl isolate multi-user.target

* Це переключить систему в текстовий режим, але після перезавантаження графічний режим повернеться.

Для тимчасового переходу до графічного режиму:  
bash  
Копіювати код  
sudo systemctl isolate graphical.target

#### 2. Для систем зі старими init (SysVinit або Upstart):

У старіших версіях Linux, які використовують SysVinit, замість systemd використовувалися runlevels (рівні запуску).

##### Перехід у текстовий режим (CLI):

Щоб змінити рівень запуску на текстовий режим (3 рівень):  
bash  
Копіювати код  
sudo init 3

##### Перехід у графічний режим (GUI):

Для перемикання на графічний режим (5 рівень):  
bash  
Копіювати код  
sudo init 5

Щоб зробити ці зміни постійними, необхідно редагувати конфігураційні файли, наприклад, /etc/inittab (для старих систем) або налаштування в systemd для сучасних дистрибутивів.

### Відмінності між CLI та GUI:

1. Command Line Interface (CLI) — текстовий інтерфейс:
   * Інтерфейс: Користувач взаємодіє із системою через текстові команди, які вводяться з клавіатури в командний рядок.
   * Переваги:
     + Менше споживання ресурсів системи (пам'ять, процесор).
     + Більш точний і гнучкий контроль над системою.
     + Підходить для адміністраторів і досвідчених користувачів, особливо для управління віддаленими серверами.
   * Недоліки:
     + Вища крива навчання для новачків.
     + Відсутність візуальних елементів і графічних програм, що може ускладнити виконання деяких завдань.
2. Graphical User Interface (GUI) — графічний інтерфейс:
   * Інтерфейс: Користувач взаємодіє із системою через вікна, іконки, меню та інші графічні елементи, використовуючи мишу і клавіатуру.
   * Переваги:
     + Простота використання, особливо для нових користувачів.
     + Підтримка широкого спектру додатків із графічними інтерфейсами.
     + Інтуїтивний спосіб взаємодії із системою, особливо для тих, хто не знайомий з командним рядком.
   * Недоліки:
     + Більше споживання ресурсів (більше пам'яті та процесорного часу).
     + Менш точний контроль над системою в порівнянні з CLI.

### Висновок:

* CLI підходить для адміністрування, автоматизації та виконання складних команд, особливо на серверах і системах з обмеженими ресурсами.
* GUI ідеальний для користувачів, яким потрібен зручний і візуально орієнтований спосіб взаємодії із системою, особливо на настільних комп'ютерах.

### 