# Змiст

[HW4](#HW4)

[HW3](#HW3)

[HW5](#_HW5)

|  |  |
| --- | --- |
| HW 4  1. Створіть користувача з іменем "bob". 2. Додайте створеного користувача до групи sudo (щоб він міг виконувати команди як адміністратор). 3. Створіть сценарій у каталозі /home/bob/, який під час виконання змінить ім'я хоста для "ubuntu22". Атрибути виконання сценарію повинні бути встановлені виключно для користувача "bob". 4. Запустіть сценарій. Перезавантажте систему. Увійти в систему як "bob" користувача. 5. Встановіть "nginx". Перевірте, чи працює nginx, а також використовуйте netstat, щоб побачити, які порти є ВІДЧИНЕНО.   Надішліть результат своєї роботи у свій репозиторій GitHub, а потім додайте посилання на папка з виконаним завданням у відповіді на це завдання. | **рішення для пунктів 1 2 3**  sudo useradd bob  sudo usermod -aG sudo bob  cd ~./bob  vi .bashrc  #creat promt for bob  export PS1=’\u@ubuntu22 \w\$ ‘  wq  sudo source ~/.bashrc    **рішення для пунктів 5**  sudo apt install nginx  sudo systemctl status nginx    sudo apt install net-tools -y  sudo netstat -tunlp | grep nginx |
| HW3 # 1. Explain the stages involved in booting the Linux system.  2. How to view system logs?  3. -rw------- : Describe this permission. How to add an executable flag to the file?  4. What is the difference between apt and dpkg? | **1. Explain the stages involved in booting the Linux system.**   1. **BIOS (Basic Input/Output System)**    * BIOS - це програма, вбудована в материнську плату, яка визначає всі фізичні пристрої, тестує систему і допомагає запустити завантажувач операційної системи. 2. **Master Boot Record (MBR)**    * MBR знаходиться на початку жорсткого диска і містить інформацію про завантажувальні сектори диска та про завантажувальну програму для операційної системи, таку як GRUB або LILO. 3. **GRUB**    * GRUB (GRand Unified Bootloader) завантажує ядро операційної системи (Kernel), визначаючи параметри ядра. 4. **Kernel (Ядро)**    * Ядро керує апаратним забезпеченням, формує кореневий файловий розділ і запускає перший процес - init. 5. **Init**    * Init (який зараз частіше називають systemd) запускає процеси і сервіси для роботи системи. 6. **Run Levels (Рівні виконання)**    * Вони ініціюють системні виклики (sys calls), які взаємодіють з ядром.  Рівні виконання:  * + **Рівень виконання 0**: Вимкнення системи (shutdown).   + **Рівень виконання 1**: Однокористувацький режим (single user interface).   + **Рівень виконання 2**: Багатокористувацький режим без підтримки NFS (Network File System) - підтримка мережевої файлової системи.   + **Рівень виконання 3**: Багатокористувацький режим з підтримкою NFS (Network File System) - підтримка мережевої файлової системи.   + **Рівень виконання 4**: Графічний інтерфейс (GUI).   + **Рівень виконання 5**: Перезавантаження системи (restart).   **How to view system logs?** Jourrnalctl (system log journal for binar files) ; dmesg (view kernel messages) cat /var/log/syslog (view syslog files) ; grep (keyword) /var/log/syslog; tail -f /var/log/syslog (log in real time) **-rw------- : Describe this permission. How to add an executable flag to the file?** Це файл. Можна записувати і читати тільки для власника, група та інші користувачі не можуть читати, записувати та запускати файл; команда chmod +x file змінює дозволи на -rwx--x--x.  **4. What is the difference between apt and dpkg?**  dpkg встановлює пакети без вирішення залежностей пакетів. apt працює на базі архітектури dpkg та встановлює пакети з урахуванням залежностей. |
| HW5 | **Exercise 1: Hello World**  Write a Bash script that simply echoes "Hello, World!" when executed.    #!/bin/bash  #Say hello to world  echo "Hello World"  **Exercise 2 User Input**  Create a script that asks the user for their name and then greets them using that name.    #!/bin/bash  #Create a script that asks the user for their name and then greets them using that name.  #Запитуємо у користовуча його імя  echo "Say us what is your name is: "  read name  #Виводимо імя на екран  echo "Вітаємо $name"  **Exercise 3: Conditional Statements**  Write a script that checks if a file exists in the current directory. If it does, print a message saying it exists; otherwise, print a message saying it doesn't exist.    #!/bin/bash  #Enter a file name  echo "Write a filename: "  read filename  #Check file in the directory  if [ -e "$filename" ]; then  echo "$filename exists"  else  echo "There is no the $filename file"  fi  **Exercise 4: Looping**  Create a script that uses a loop to print numbers from 1 to 10.    #!/bin/bash  for ((i=1; i<=10; i++))  do  echo $i  done  #!/bin/bash  number=1  while [ $number -le 10 ]  do  echo $number  number=$((number +1))  done  **Exercise 5: File Operations**  Write a script that copies a file from one location to another. Both localtions should be passed as arguments    #!/bin/bash  SOURCE=$1  DESTINATION=$2  cp "$SOURCE" "$DESTINATION"  if [ $? -eq 0 ]; then  echo "Success"  fi  **Exercise 6: String Manipulation**  Build a script that takes a user's input as a sentence and then reverses the sentence word by word (e.g., "Hello World" becomes "World Hello").    #!/bin/bash  #Input sentence  echo "Type sentence: "  read sentence  reverse\_sentence=$(echo "$sentence" | tr ' ' '\n' | tac | xargs )  echo "That is your reverse sentence: $reverse\_sentence"  **Exercise 7: Command Line Arguments**  Develop a script that accepts a filename as a command line argument and prints the number of lines in that file.    #!/bin/bash  #We check does the file exist.  filename=$1  if [ ! -f "$filename" ]; then  echo "File did not find"  exit 1  fi  #We load file into the while cycle.  line\_count=0  while IFS= read -r line  do  line\_count=$(($line\_count +1))  done < $filename  echo "The file $filename has $line\_count lines."  **Exercise 8: Arrays**  Write a script that uses an array to store a list of fruits. Loop through the array and print each fruit on a separate line.    #!/bin/bash  fruits=("яблуко" "бананан" "orange" "груша")  for fruit in ${fruits[@]}  do  echo "$fruit"  done  **Exercise 9: Error Handling**  Develop a script that attempts to read a file and handles errors gracefully. If the file exists, it should print its contents; if not, it should display an error message.    #!/bin/bash  #Check does the file exist.  filename=$1  if [ ! -f $filename ]; then  echo "Error: file $filename doesn't exist"  exit 1  else  echo "Contents of the file $filename: "  cat $filename  fi  **### Systemd service**  Write script which watching directory "~/watch". If it sees that there appeared a new file, it prints files content and rename it to \*.back    #!/bin/bash  # Создание директории для мониторинга  WATCH\_DIR=~/Watch  mkdir -p "$WATCH\_DIR"  # Start of files creation monitoring  inotifywait -m -e create "$WATCH\_DIR" | while read path action file; do  # Fix inotifywait bag. Cut extra information from var “file"  filename=$(echo "$file" | awk '{print $NF}')  # Ignor temporary files  if [[ "$filename" == .\* ]] || [[ "$filename" == \*.swp ]] || [[ "$filename" == \*.swx ]]; then  continue  fi  # Display the file its content and rename  file\_path="$WATCH\_DIR/$filename"  echo "1) New file: $filename created in $WATCH\_DIR"  echo "2) Here is the content of the file: $file\_path"  cat "$file\_path"  mv "$file\_path" "$file\_path.back"  echo "3) File: $file\_path was renamed to $file\_path.back"  done  **Write SystemD service for this script and make it running**    [Unit]  Description=Watch Directory Service  After=network.target  [Service]  ExecStart=/home/vagrant/dan\_it\_homeworks/danit\_scripts/monitoring\_newfile\_creation.sh  WorkingDirectory=/var/lib/watch\_service  StandardOutput=file:/var/lib/watch\_service/watch\_directory.log  StandardError=journal  Restart=always  User=vagrant  Group=vagrant  [Install]  WantedBy=multi-user.target |