Міністерство освіти і науки України
ЧЕРНІГІВСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ

ОПЕРАЦІЙНІ СИСТЕМИ

Методичні вказівки
до виконання розрахунково-графічної роботи
для студентів за напрямом підготовки (спеціальністю)
6.170103 «Управління інформаційною безпекою»,
125 «Кібербезпека»
денної форми навчання

Обговорено і рекомендовано
на засіданні кафедри
кібербезпеки та математичного моделювання
Протокол № 8
від 19 лютого 2019 р.

Чернігів ЧНТУ 2019

Операційні системи. Методичні вказівки до виконання розрахунково-графічної роботи для студентів за напрямом підготовки (спеціальністю) 6.170103 «Управління інформаційною безпекою», 125 «Кібербезпека» денної форми навчання / Укл.: Петренко Т.А – Чернігів: ЧНТУ, 2019. – 26 с.

Укладач: Петренко Тарас Анатолійович, старший викладач кафедри кібербезпеки та математичного моделювання

Відповідальний за випуск: Ткач Юлія Миколаївна, д.п.н., доц., завідувач кафедри кібербезпеки та математичного моделювання

Рецензент: Ткач Юлія Миколаївна, д.п.н., доц., завідувач кафедри кібербезпеки та математичного моделювання

ЗМІСТ

ПЕРЕДМОВА

Метою викладання навчальної дисципліни "Операційні системи" є засвоєння необхідних знань з опанування теоретичних основ побудови, принципів проектування, конфігурування й застосування різних сучасних операційних систем, які забезпечують організацію обчислювальних процесів у корпоративних інформаційних системах економічного, управлінського, виробничого, наукового й іншого призначення, а також надання практичних навичок щодо автоматизації повсякденних завдань адміністрування та захисту інформації.

Предметом вивчення навчальної дисципліни є теоретичні концепції та методології, принципи функціонування, вибору і практичної реалізації складових операційної системи. Основи побудови операційних систем, новітні тенденції розвитку комп'ютерної техніки, задачі підтримки персонального комп'ютера в працездатному стані, налагоджування роботи його програмного забезпечення та конфігурації, своєчасний апгрейд та патч операційних систем.

Об'єктом вивчення дисципліни є типові механізми, технології, протоколи взаємодії різноманітних компонент операційної системи, як на рівні ядра, так і на рівні користувача. Необхідним елементом успішного засвоєння навчального матеріалу дисципліни є самостійна робота студентів з науково-технічною літературою та з сучасними технічними засобами комп'ютеризованої обробки інформації. Саме тому в програмі навчальної дисципліни "Операційні системи" передбачено виконання студентами розрахунково-графічної роботи.

Основними завданнями вивчення дисципліни "Операційні системи" є:

– засвоєння студентами необхідних знань з опанування теоретичних основ побудови, конфігурування й застосування різних сучасних операційних систем, які забезпечують організацію обчислювальних процесів у корпоративних інформаційних системах економічного, управлінського, виробничого, наукового й іншого призначення;

– опанування студентами теорії, методик та отримання досвіду з

проектування та програмування операційних систем;

– надання практичних навичок щодо автоматизації повсякденних завдань адміністрування;

– формування в студентів бази знань, умінь і навичок, необхідних для кваліфікованого та ефективного використання сучасних операційних систем в управлінні інформаційною безпекою конкретного підприємства.

Запропоновані завдання для розрахунково-графічної роботи студентів включають методичні вказівки, завдання для виконання та критерії оцінювання. За допомогою розрахунково-графічної роботи та запропонованих завдань досягається більш глибоке опанування теорії, що здійснюється за допомогою розвитку логічного мислення через вирішення задач та дає змогу студентам осмислити нові для них поняття. Завдання для розрахунку скомпоновані відповідно до розділів робочої програми «Операційні системи», II семестр навчання, що полегшує і робить більш зручною організацію навчального процесу і викладачам, і студентам.

Завдання для розрахунково-графічної роботи студентів можуть використовуватися як для аудиторної, так і домашньої роботи. Вони спрямовані на розвиток у студентів організаційних та аналітичних здібностей, а також уміння користуватися теоретичними посиланнями у вирішенні практичних ситуацій та вміння користуватися статистикою і спеціальною літературою. Завдання для розрахунково-графічної роботи студентів можуть значною мірою полегшити вивчення дисципліни студентами очної форми навчання.

Під час виконання розрахунково-графічної роботи студенти повинні ознайомитися та вивчити лекційний матеріал, запропонований викладачем. Основою для вивчення є літературні джерела, наведені в даній методичній розробці. За наявності незрозумілих питань студентам рекомендується звернутись за консультаціями до викладача з метою отримання всіх необхідних пояснень щодо організації розрахунково-графічної роботи, виконання завдань з програмування та пошуку додаткових літературних джерел. Викладачем

надаються додаткові роз’яснення та індивідуальні консультації для підвищення компетентності студентів та розширення спектру їх знань з даної дисципліни.

СТРУКТУРА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Назви змістових модулів і тем	денна форма					
	усього	у тому числі				
		л	п	лаб	інд	с.р.
Змістовий модуль №1. Основні положення ОС. Основи системного програмування.						
Тема 1. Вступ до ОС	6	2	-	-	-	4
Тема 2. Взаємодія з апаратною частиною	14	2	-	4	-	8
Тема 3. Бінарний інтерфейс.	16	-	-	6	-	10
Разом за змістовим модулем 1	**36**	**4**	**-**	**10**	**-**	**22**
Змістовий модуль №2. Управління в ОС. Файлова система.						
Тема 4. Управління пам’яттю	10	2	-	-	-	8
Тема 5. Виконувані файли	18	2	-	2	-	14
Тема 6. Управління процесами	8	2	-	-	-	6
Тема 7. Синхронізація	12	-	-	2	-	10
Тема 8. Файлова система	16	2	-	2	-	12
Разом за змістовим модулем 2	**64**	**8**	**-**	**6**	**-**	**50**
Змістовий модуль №3. Взаємодія з мережею. Безпека в ОС. Особливості ОС						
Тема 9. Взаємодія з мережею	14	2	-	4	-	8
Тема 10. Безпека.	8	2	-	-	-	6
Тема 11. Unix	16	-	-	4	-	12
Тема 12. Windows	12	-	-	-	-	12
Разом за змістовим модулем 3	**50**	**4**	**-**	**8**	**-**	**38**
Всього	**150**	**16**	**-**	**24**	**-**	**110**

КРИТЕРІЇ ОЦІНЮВАННЯ РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ

Розрахунково-графічна робота виконуються за окремим графіком. Студент самостійно готується до такого заняття за індивідуальним завданням. Обсяг розрахунково-графічної роботи визначається навчальним планом з дисципліни.

З даного курсу розрахунково-графічна робота проводиться у формі виконання індивідуальних завдань з розробки shell-скрипта в операційній системі Linux.

Шкала оцінювання знань студентів при виконанні розрахунково-графічної роботи

Рівень виконання розрахункової роботи	**Кількість балів**	
- shell-скрипт реалізований правильно і повністю виконує поставлені задачі, містить пояснення до всіх елементів - здійснено посилання на нормативну базу та технічну документацію - показано вміння самостійно формулювати висновки за результатами проведеного дослідження - присутній творчий підхід та використано новітні інформаційні технології	9…	10
- завдання виконані повністю, але при розв’язуванні допущені незначні помилки; - не аргументовано викладено матеріал; - у висновках містяться помилки та недоречності	6…	8
- завдання виконане не у повному обсязі та допущено значні помилки - не сформульовані висновки за результатами розрахунків	3…	5
- завдання виконано частково і неякісно - записаний тільки результат	0…	2

У зв’язку з тим що, розрахунково-графічна робота містить завдання що передбачає ґрунтовні знання з програмування та операційних систем, вона може бути виконана після вивчення всіх тем курсу, оцінюється вона після закінчення третього модуля і оцінка за виконання розрахунково-графічної роботи, додається до підсумкової модульної оцінки, переведеної за шкалою ECTS.

ВИМОГИ ДО ОФОРМЛЕННЯ РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ

Робота оформляється на листах А4 з однієї сторони, поля: з лівого боку – 20 мм, з правого боку – 10 мм, зверху – 20 мм, знизу – 20 мм. Завдання повинно бути виконано акуратно, розбірливим почерком (або надруковане), з детальними поясненнями та всіма проміжними результатами. В кінці програмного завдання пишеться висновок.

Вимоги до комп’ютерного набору розрахункової роботи:

– текстовий редактор – WORD;

– гарнітура шрифту – Times New Roman;

– кегль шрифту (розмір) – 14;

– міжрядковий інтервал – полуторний;

– абзац – 1,25 см;

– розташування тексту роботи – вирівнювання по ширині;

– міжрядковий інтервал між заголовком (назвою розділу чи підрозділу) і текстом повинна дорівнювати 1 інтервалу.

Приклад оформлення титульної сторінки розрахунково-графічної роботи наведено у Додатку А.

Повністю оформлена і виконана розрахункова робота подається на кафедру в термін, що визначений у плані-графіку виконання розрахункової роботи для перевірки її викладачем. Якщо робота виконана не вчасно без поважних причин, то студенту ставиться 0 балів («незадовільно») і він повинен виконати додатково один з варіантів, який вкаже викладач. Розрахункова

робота оцінюється після особистої співбесіди з викладачем. В разі зауважень з боку викладача, робота повинна бути доопрацьована в зазначений термін і подана на перевірку. До підсумкового контролю допускаються лише студенти, що вчасно здали і захистили свою роботу.

Варіант розрахунково-графічної роботи видається студенту викладачем (згідно порядкового номеру в списку академічної групи або в інший спосіб).

ЗАВДАННННЯ ДО РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ

Необхідно написати shell-скрипт, який опрацює текстовий файл log.txt у форматі Acess log веб-сервера Apache і виведе в консоль інформацію відповідно до варіантів завдання**.** Цей скрипт повинен використовувати стандартні інструменти shell (які включають такі утиліти, як: cut, grep, sort, awk, date та ін.), але не використовувати інші мови програмування, такі як C, Perl, Python та ін.

Файл складається з записів, кожен з яких займає один рядок.

Приклад запису: *host-24-225-218-245.patmedia.net - [01/0ct/2006: 6:33:45 - 0700] "GET / example / example.atom HTTP/1.1" 304 - "-" "NetNewsWire / 2.0b37 (Mac OS X; Lite; http://ranchero.com/netnewswire/) "*

Формат запису: Хост клієнта - [Штамп часу з часовою зоною] Рядок HTTP- запиту (тип, URL, версія) Код HTTP-відповіді Кількість переданих байт або '-', якщо відповідь не має тіла Рядок реферера ('-' означає прямий запит без реферера) Назва клієнта (браузер)

Розшифровка: **01/0ct/2006: 6:33:45 -0700** з хоста **host-24-225-218-245.patmedia.net** по протоколу **HTTP/1.1** був здійснений запит типу **GET** для отримання ресурсу що перебуває по посиланню **/example/example.atom.** Код відповіді на запит від сервера: **304**. Така відповідь не передбачає наявності тіла відповіді (кількість переданих байт - **0**). Запит виконувався безпосередньо, а не по посиланню з іншого сайту (поле реферер - порожнє). Клієнт використовував для звернення програму **NetNewsWire/2.0b37**, ОС клієнта: **Mac OS X**

Варіант №1.

1. Топ-10 реферерів по загальній кількості завантажених байтів зі запитів з них.

2. Кількість завантажених байтів для кожного з них як число.

3. Кількість завантажених байтів для кожного з них як процент від загальної кількості байт, завантажених за зверненнями з цих реферерів.

(Під Топ-10 розуміється до 10 штук реферерів відсортованих в порядку зменшення, починаючи з найбільшого значення).

Приклад результату:

1. http://www.example.org/example/When/200x/2006/09/25/ - 3100 - 74%
2. http://www.example.org/example/ - 1000 - 24%
3. http://www.example.org/example/genx/docs/Guide.html - 91 - 2%

Варіант №2

1. Топ-10 URL, які кикликали помилки клієнта (код відповіді починається з 4).

2. Кількість помилок для кожного з них як число.

3. Кількість помилок для кожної з них як процент від загальної кількості помилок для цих URL.

(Під Топ-10 розуміється до 10 штук URL в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. http://www.example.org/example/ - 50 - 50%
2. http://www.example.org/example/genx/docs/Guide.html - 30 - 30%
3. http://www.example.org/example/When/200x/2006/09/25/ - 20 - 20%

Варіант №3

1. Топ-10 реферерів для найпопулярнішої URL.

2. Кількість запитів від кожного з них як число.

3. Кількість запитів від кожного з них як процент від загально кількості запитів від цих реферерів до найпопулярнішої URL.

(Під Топ-10 розуміється до 10 штук URL в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. http://www.example.org/example/ - 50 - 50%

2. http://www.example.org/example/When/200x/2006/09/25/ - 30 - 30%

3. http://www.example.org/example/genx/docs/Guide.html - 20 - 20%

Варіант №4

1. Топ-10 дат по кількості запитів.

2. Кількість запитів в кожну з них як число.

3. Кількість запитів в кожну з них як процент від загальної кількості запитів за всі ці дати.

(Під Топ-10 розуміється до 10 штук дат в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. 2006-10-18 - 1200 - 36%

2. 2006-10-01 - 1130 - 34%

3. 2006-10-02 - 1000 - 30%

Варіант №5

Топ-10 дат по кількості унікальних хостів.

Кількість запитів в кожну з них як число.

Кількість запитів в кожну з них як процент від загальної кількості запитів з цих хостів за всі ці дати.

(Під Топ-10 розуміється до 10 штук дат в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. 2016-10-18 - 1200 - 36%

2. 2016-10-01 - 1130 - 34%

3. 2016-10-02 - 1000 - 30%

Варіант №6

1. Топ-10 дат по загальній кількості скачаних байт.

2. Кількість скачаних байт в кожну з них як число.

3. Кількість скачаних байт в кожну з них як процент від загальної кількості байт скачаних за всі ці дати.

(Під Топ-10 розуміється до 10 штук дат в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. 2016-10-18 - 1200 - 36%

2. 2016-10-01 - 1130 - 34%

3. 2016-10-02 - 1000 - 30%

Варіант №7

1. Топ-10 дат по кількості помилок клієнта (код відповіді починається з 4).

2. Кількість помилок для кожної з них як число.

3. Кількість помилок в кожну з них як процент від загальної кількості помилок за всі ці дати.

(Під Топ-10 розуміється до 10 штук дат в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. 2016-10-18 - 1200 - 36%

2. 2016-10-01 - 1130 - 34%

3. 2016-10-02 - 1000 - 30%

Варіант №8

1. Топ-10 URL по кількості запитів за дату 28 жовтня 2016р.

2. Кількість запитів до кожної з них як число.

3. Кількість запитів до кожної з них як процент від загальної кількості запитів до цих URL.

(Під Топ-10 розуміється до 10 штук URL в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. /example.com - 311 - 94%

2. /img/image.png - 10 - 3%

3. /robots.txt - 9 - 3%

Варіант №9

1. Топ-10 URL по загальній кількості скачаних байт.

2. Кількість скачаних байт з кожного з них як число.

3. Кількість скачаних бійт з кожного з них як процент від загальної кількості байт, скачаних з цих URL.

(Під Топ-10 розуміється до 10 штук URL в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. /example.com - 3100 - 74%

2. /img/image.png - 1000 - 24%

3. /robots.txt - 91 - 2%

Варіант №10

1. Топ-10 URL, які кикликали помилки клієнта (код відповіді починається з 4).

2. Кількість помилок для кожного з них як число.

3. Кількість помилок для кожної з них як процент від загальної кількості помилок для цих URL.

(Під Топ-10 розуміється до 10 штук URL в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. /example.com - 50 - 50%

2. /img/image.png - 30 - 30%

3. /robots.txt - 20 - 20%

Варіант №11

1. Топ-10 URL в яких був реферер.

2. Кількість реферерів для кождної з них як число.

3. Кількість реферерів для кожної з них як процент від загальної кількості реферерів для цих URL.

(Під Топ-10 розуміється до 10 штук URL в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. /example.com - 50 - 50%

2. /img/image.png - 30 - 30%

3. /robots.txt - 20 - 20%

Варіант №12

1. Топ-10 URL по кількості унікальних хостів за дату 28 жовтня 2016р.

2. Кількість запитів до кожного з них як число.

3. Кількість запитів до кожного з них як процент від загальної кількості запитів з цих хостів до цих URL.

(Під Топ-10 розуміється до 10 штук URL в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. /example.com - 311 - 94%

2. /img/image.png - 10 - 3%

3. /robots.txt - 9 - 3%

Варіант №13

1. Топ-10 хостів по кількості запитів за дату 1 жовтня 2006р.

2. Кількість запитів з кожного з них як число

3. Кількість звернень від кожного з них як процент від загальної кількості звернень з цих хостів.

(Під Топ-10 розуміється до 10 штук хостів в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. lj612152.inktomisearch.com

2. ac914c7e.ipt.aol.com - 3 -

3. fj301012.inktomisearch.com

Варіант №14

1. Топ-10 хостів по загальній кількості скачаних байт кожним з них.

2. Кількість скачаних байт кожним з них як число

3. Кількість скачаних байт кожним з них як процент від загальної кількості скачаних байт цими хостами.

(Під Топ-10 розуміється до 10 штук хостів в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. lj612152.inktomisearch.com – 1000 – 59%

2. ac914c7e.ipt.aol.com – 500 – 29%

3. fj301012.inktomisearch.com – 200 – 12%

Варіант №15

1. Топ-10 хостів які були перенаправлені сервером (код відповіді починається з 3).

2. Кількість перенаправленнь для кожного з них як число.

3. Кількість перенаправленнь для кожного з них як процент від загальної кількості пренаправленнь для цих хостів.

(Під Топ-10 розуміється до 10 штук хостів в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. lj612152.inktomisearch.com - 10 - 59%

2. ac914c7e.ipt.aol.com - 5 - 29%

3. fj301012.inktomisearch.com - 2 - 12%

Варіант №16

1. Топ-10 хостів,які прийшли через реферера.

2. Кількість рефererів для кожного з них як число.

3. Кількість рефererів для кожного з них як процент від загальної кількості рефererів для цих хостів.

(Під Топ-10 розуміється до 10 штук хостів в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. lj612152.inktomisearch.com - 10 - 59%

2. ac914c7e.ipt.aol.com - 5 - 29%

3. fj301012.inktomisearch.com - 2 - 12%

Варіант №17

1. Топ-10 програм (не розрізняти по версіям) по кількості запитів з дату 10 жовтня 2016р.

2. Кількість запитів від кожної з них як число.

3. Кількість запитів від кожної з них як процент від загальної кількості звернень від цих програм.

(Під Топ-10 розуміється до 10 штук програм в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. NetNewsWire - 5 - 42%

2. Feedfetcher-Google - 3 - 25%

3. Mozilla - 2 - 16%

4. BitTorrent - 2 - 16%

Варіант №18

1. Топ-10 програм (не розрізняти по версіям) по кількості скачаних байт кожною з них.

2. Кількість скачаних байт від кожної з них як число.

3. Кількість скачаних байт від кожної з них як процент від загальної кількості скачаних байт цими програмами.

(Під Топ-10 розуміється до 10 штук програм в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. NetNewsWire - 1000 - 59%

2. BitTorrent - 500 - 29%

3. Mozilla - 200 - 12%

Варіант №19

1. Топ-10 програм (не розрізняти по версіям) які були перенаправлені сервером (код відповіді починається з 3).

2. Кількість перенаправлень кожної з них як число.

3. Кількість пренаправлень кожної з них як процент від загальної кількості перенаправлень цих програм.

(Під Топ-10 розуміється до 10 штук програм в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. NetNewsWire - 10 - 59%

2. Mozilla - 5 - 29%

3. BitTorrent - 2 - 12%

Варіант №20

1. Топ-10 програм (не розрізняти по версіям) які прийшли через реферера.

2. Кількість рефererів для кожної з них як число.

3. Кількість рефererів для кожної з них як процент від загальної кількості рефererів для цих програм.

(Під Топ-10 розуміється до 10 штук програм в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. Mozilla - 10 - 59%

2. NetNewsWire - 5 - 29%

3. BitTorrent - 2 - 12%

Варіант №21

1. Топ-10 програм (розрізняти по версіям) по кількості запитів з дату 11 жовтня 2016р.

2. Кількість запитів від кожної з них як число.

3. Кількість запитів від кожної з них як процент від загальної кількості звернень від цих програм.

(Під Топ-10 розуміється до 10 штук програм в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. Feedfetcher-Google - 3 - 25%

2. Mozilla/5.0 - 2 - 16%

3. BitTorrent/4.0.0 - 2 - 16%

Варіант №22

1. Топ-10 програм (розрізняти по версіям) по кількості скачаних байт кожною з них.

2. Кількість скачаних байт від кожної з них як число.

3. Кількість скачаних байт від кожної з них як процент від загальної кількості скачаних байт цими програмами.

(Під Топ-10 розуміється до 10 штук програм в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. NetNewsWire/2.0.1 - 1000 - 59%

2. Feedfetcher-Google - 500 - 29%

3. Mozilla/5.0 - 200 - 12%

Варіант №23

1. Топ-10 програм (розрізняти по версіям) які викликали помилку клієнта (код відповіді починається з 4).

2. Кількість помилок для кожної з них як число.

3. Кількість помилок для кожної з них як процент від загальної кількості помилок цих програм.

(Під Топ-10 розуміється до 10 штук програм в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. NetNewsWire/2.0.1 - 10 - 59%

2. Mozilla/5.0 - 5 - 29%

3. BitTorrent/4.0.0 - 2 - 12%

Варіант №24

1. Топ-10 програм (розрізняти по версіям) які прийшли через реферера.

2. Кількість реферерів для кожної з них як число.

3. Кількість реферерів для кожної з них як процент від загальної кількості реферерів для цих програм.

(Під Топ-10 розуміється до 10 штук програм в порядку зменшення, починаючи з найбільшого значення)

Приклад результату:

1. Feedfetcher-Google - 10 - 59%

2. NetNewsWire/2.0.1 - 5 - 29%

3. BitTorrent/4.0.0 - 2 - 12%

Теоретичні відомості

Shell-скрипт - це програма, написана для інтерпретації командною оболонкою ОС (shell). Ця програма існує в текстовому вигляді і не потребує

окремого етапу компіляції перед виконанням. За угодою, першим рядком скрипта є вказівка конкретного інтерпретатора, який повинен виконувати його. Взагалі кажучи, в Unix-ОС скрипти не обов'язково повинні виконуватися саме командною оболонкою, а можуть бути написані на будь-якій мові, яка підтримує інтерпретацію (наприклад, Perl або Python).

Shell-скрипт Hello World:

```
#!/bin/sh
echo "Hello World"
```

Якщо цей текст зберегти в файл hello.sh в поточній директорії, то виконати його можна двома способами:

$ sh hello, sh - в цьому випадку ми запускаємо команду sh (власне, shell) і передаємо їй як аргумент ім'я файлу скрипта

$ chmod +x hello, sh; ./hello, sh - в цьому випадку виконуються 2 команди: спочатку файлу скрипта дається право на виконання, потім запускається сам файл і командна оболонка, в якій виконується ця команда, аналізує початок файлу Якщо це скомпільована програма, то вона містить у перших байтах тзв. магічний номер (magic number), який ідентифікує формат виконуваного файлу в який вона скомпілірованна - в цьому випадку shell передає управління завантажувачу програми, який підтримує відповідний формат Якщо це скрипт і він містить тзв. shebang-рядок (рядок, що починається з символів #!), то весь його вміст передається програмі, шлях до якої зазначений в цьому рядку (в даному випадку: /bin/sh). Інакше програма вважається скриптом самої командної оболонки і виконується нею самою.

Примітка 1: $ - запрошення консолі ОС. Такий запис означає, що команду потрібно виконати в консолі (без самого знака $).

Примітка 2: програмна оболонка по-замовчуванню в середовищі Linux - bash (Bourne Again Shell). Уданому прикладі в якості інтерпретатора зазначений sh (Bourne Shell), який є більш простим варіантом командної оболонки, попередником bash. У випадках, коли в скрипті не використовуються специфічні розширення bash, правилом хорошого тону є

вказувати в якості інтерпретатора саме sh (з міркувань більшої переносимості коду: не на всіх системах може бути встановлений bash).

Shell-скрипти для Boune Shell і її варіантів можуть використовувати ті ж самі команди, які можна вводити і з консолі операційної системи. Команда man - дозволяє отримати довідку по будь-якій команді. $ man sh дозволить вам вивчити синтаксис самого Shell. Важливими операторами Shell є перенаправлення виводу (>) і введення (<), а також конвеєр pipe (|), який дозволяє перенаправляти вивід однієї програми на введення іншої.

На рівні з вбудованими командами Shell може бути запущена будь-яка програма, якщо вона знайдена в пошуковому шляху PATH, або ж до неї зазначений повний шлях. PATH - це одна з змінних оточення Shell, яка доступна всім процесам. Отримати її значення можна так: $PATH, а встановити (в sh / bash): export PATH=…. Дізнатися значення всіх змінних оточення можна командою env. Серед змінних оточення є кілька спеціальних змінних, які встановлюються індивідуально для кожного запущеного процесу Наприклад, це змінна $*, в якій містяться всі аргументи, з якими запущена дана програма, чи $1, $2, …, які містять перший, другий і тд. аргумент

PATH не включає поточну директорію, тому запуск програми з поточної директорії, як правило, виконується за допомогою синтаксису . / (Тобто $. /program). Точка в Shell - це псевдонім для поточної директорії. Іншими псевдонімами є .. - директорія на рівень вище, і ~ - домашня директорія. Команди і програми можуть приймати рядкові аргументи, які кожна з них може інтерпретувати по-своєму Як правило, ці аргументи бувають 3-х типів:

Просто значення (числа, рядки), наприклад в $ echo "Hello World" "Hello World" - це просто рядок.

Шляхи, наприклад в $ cat hello.txt hello.txt - це шлях до файлу у поточній директорії. Повний шлях міг би виглядати так: /home/user/hello.txt

Аргументи-ключі: починаються з - або --, наприклад в $ wc -1 file.txt команда wc рахує кількість тільки рядків (ключ -1) у файлі file.txt. Ключ - - help дозволяє отримати коротку довідку по переважній більшості команд.

ДОДАТОК А

Титульна сторінка розрахунково-графічної роботи

Міністерство освіти і науки України
Чернігівський національний технологічний університет
Кафедра кібербезпеки та математичного моделювання

Розрахунково-графічна робота

з дисципліни „Операційні системи”

варіант ____

виконав(ла)
студент(ка) групи _____
_____
(прізвище, ім’я, по-батькові)

перевірив
_____
(прізвище, ім’я, по-батькові)
оцінка _____ балів
Підпис викладача _____

Чернігів 201_

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

Базова

1. Stallings, William (2008). Computer Organization & Architecture. New Delhi: Prentice-Hall of India Private Limited. p. 267. ISBN 978-81-203-2962-1.
2. Stallings (2005). Operating Systems, Internals and Design Principles. Pearson: Prentice Hall. p. 6.
3. Jump up Dhotre, I.A. (2009). Operating Systems. Technical Publications. p. 1.
4. Бовет Д. Ядро Linux / Бовет Д., Чезати М. ; пер. с англ. – СПб. : БХВ-Петербург, 2007. – 1104 с.
5. Третяк В. Ф. Основи операційних систем : навч. посібн. / В. Ф. Тре-тяк, Д. Ю.Голубничий, С. В. Кавун. – Х. : Вид. ХНЕУ, 2005. – 228 с.
6. Таненбаум Э. Современные операционные системы / Таненбаум Э. – СПб.: Питер, 2010. – 1120 с.
7. Шеховцов В. А. Операційні системи / В. А. Шеховцов. – К. : Видавнича група BHV, 2005. – 576 с.

Допоміжна

1. Бэкон Дж. Операционные системы / Бэкон Дж., Харрис Т. – К. : Издат. группа BHV ; СПб. : Питер, 2004. – 800 с.
2. Олифер В. Г. Сетевые операционные системы / В. Г. Олифер, Н. А. Олифер. – СПб. : Питер, 2002. – 544 с.
3. Столингс В. Операционные системы / Столингс В. – М. : Вильямс, 2002. – 848с.
4. Голубничий Д. Ю. Системне програмування і операційні системи : навч. посібн. Ч. 1 / Д. Ю. Голубничий, В. Ф. Третяк. – Х. : Вид. ХДЕУ, 2004. – 192 с.
5. Голубничий Д. Ю. Системне програмування та операційні системи : навч. посібн. Ч. 2 / Д. Ю. Голубничий, В. Ф. Третяк, С. В. Кавун. – Х. : Вид. ХНЕУ, 2005. – 264 с.

6. Сорокина С. И. Программирование драйверов и систем безопасности : учебн. пособ. / С. И. Сорокина, А. Ю. Тихонов, А. Ю. Щерба-ков. – СПб. : БХВ-Петербург, 2003. – 256 с.
7. Джонсон М. Разработка приложений в среде Linux / М. Джонсон, Э. Троян ; пер. с англ. – М. : ООО "И.Д. Вильямс", 2007. – 544 с.
8. Секунов Н. Ю. Программирование на C++ в Linux / Н. Ю. Секу-нов. – СПб.: БХВ-Петербург, 2004. – 368 с.

Інформаційні ресурси

1. Основи роботи з Linux

http://matt.might.net/articles/basic-unix/

http://www.funtoo.org/Linux_Fundamentals,_Part_l

2. Основи роботи з Bash

http://ods.com.ua/koi/unix/bash-conspect.html

http://www.tldp.org/LDP/abs/html/

http://www.davidpashley.com/articles/writing-robust-shell-scripts.html

http://mywiki.wooledge.org/BashFAQ

Bash by Example: http://matt.might.net/articles/bash-by-example/

3. Робота з текстовими даними

http://www.ibm.com/developerworks/aix/library/au-unixtext/index.html

http://radar.oreilly.com/2011/04/data-hand-tools.html

http://www.pement.org/awk/awklline.txt

http://sed.sourceforge.net/sedlline.txt

http://www.drbunsen.org/explorations-in-unix.html

4. Регулярні вирази

http://www.regular-expressions.info/ http://www.weitz.de/regex-coach/

5. Просунуті інструменти командного рядка

http://www.commandlinefu.com/commands/browse/sort-by-votes

http://matt.might.net/articles/bash-by-example/

http://kkovacs.eu/cool-but-obscure-unix-tools