УДК 378.018.43:004.9

Щербина Олександр Андрійович
кандидат технічних наук, доцент, доцент кафедри системного програмування і спеціалізованих комп'ютерних систем, Національний технічний університет України «Київський політехнічний інститут ім. Ігоря Сікорського», м. Київ, Україна
ORCID ID: 0000-0003-0224-1441
oscherbyna@i.ua

АВТОМАТИЗАЦІЯ СТВОРЕННЯ, НАПОВНЕННЯ І АДМІНІСТРУВАННЯ КАТЕГОРІЙ КУРСІВ САЙТУ MOODLE

Анотація. Ефективність і зручність використання сайту Moodle багато в чому визначається його структурою категорій курсів. Є два основні підходи до вибору структури категорій курсів на сайті університету. Перший їх структурує з точки зору викладачів: у категоріях курсів факультету розміщуються категорії курсів кафедр. Другий – з точки зору студентів: у категорії курсів факультетів розміщуються категорії курсів освітніх програм. У роботі розглядається структура, яка найкраще поєднує ці два підходи. У ній як базовий використовується перший підхід, а списки курсів за освітніми програмами формуються у формі гіперпосилань, що реалізуються за допомогою елементів курсу *Субкурс*, розміщених у спеціально створених для них службових курсах. Це дає змогу імпортувати в журнал оцінок службового курсу результатів навчання з усіх дисциплін певної освітньої програми і року навчання та розрахувати на їх основі поточний рейтинг кожного студента, який система Moodle оновлює щохвилини, а в зворотному напрямку експортувати списки студентів. Тоді зарахування студента у службовий курс чи відрахування з нього викликає такі ж дії в усіх курсах цієї освітньої програми. Така структура сайту має істотні переваги, однак її створення вимагає великого обсягу робіт адміністратора, отже, потребує автоматизації. Для цього в роботі використовується утиліта Moosh, призначена для адміністрування сайту Moodle через командний рядок, що значно пришвидшує процес адміністрування при порівнянні з виконанням тих самих дій через власний інтерфейс Moodle. Ще більшого пришвидшення можна досягати, якщо запускати на виконання в Moosh файли, що містять багато таких команд. Але основним змістом цієї роботи є автоматизація створення самих файлів з командами. Це більшою мірою пришвидшує процес адміністрування й підвищує його якість завдяки зменшенню кількості помилок, які мають місце під час ручного набору команд чи адміністрування сайту власними засобами Moodle. У роботі описано процес створення файлів команд, які забезпечують виконання в автоматичному режимі всіх робіт, пов'язаних зі створенням зазначеної вище структури сайту: створення категорій курсів, створення службових курсів та їх наповнення, а також подальше адміністрування сайту, пов'язане із зарахуванням студентів на курси нового навчального року, їх відрахування з минулорічних курсів тощо. Також розглядаються нові плагіни, які підтримують функціонування такої структури сайту.

Ключові слова: Moodle; автоматизація адміністрування; Moosh.

1. ВСТУП

Постановка проблеми. Як відомо, основним трендом розвитку сучасної освіти є її цифровізація. Вона зокрема передбачає не подальше нагромадження різнорідних і не пов'язаних між собою засобів і технологій, а їх гармонійну інтеграцію у спільну систему управління навчанням (Learning Management System – LMS). Безумовним лідером серед таких систем як за функціональними можливостями, так і за поширеністю є Moodle [1], [2]. Вона завжди розвивалася і продовжує розвиватися швидше за своїх конкурентів і (принаймні у сфері вищої освіти), починаючи з 2018 р., займає вже більше половини світового і 2/3 європейського ринку LMS [3], якщо тільки

доречно вживати слово «ринок» стосовно безкоштовного програмного забезпечення, яким є Moodle.

Система Moodle є складним і водночас дуже гнучким інструментом. Крім базового набору засобів, наявних у ядрі Moodle, для неї розроблено понад дві тисячі додаткових плагінів [4], що доповнюють і розширюють її функції, які в багатьох випадках можуть бути реалізовані різними способами. Тоді виникає проблема вибору найкращого з них [5]. Нерідко адміністрування сайту Moodle вимагає великих обсягів робіт, і тому потребує автоматизації. Однак способи і засоби автоматизації таких робіт у зарубіжній науковій літературі досі розглядалися дуже мало [6] – [9], а у вітчизняній не розглядалися зовсім.

Як відомо, базовою структурною одиницею системи Moodle є *курс.* У ньому зазвичай розміщують матеріали для вивчення певної дисципліни. На кожний курс зараховується викладач (чи декілька викладачів) і студенти. Курси на сайті Moodle розміщуються в *категоріях курсів,* як файли в папках. Тому постає проблема вибору такої структури категорій курсів сайту, яка забезпечує його найбільшу ефективність.

Існує два підходи до побудови структури категорій курсів сайту університету. Перший структурує категорії з точки зору викладача. Тут у категоріях курсів факультетів розміщуються підкатегорії курсів кафедр, де зберігаються курси, що викладаються цією кафедрою. Другий підхід структурує категорії з позицій студента. У категоріях курсів факультетів розміщуються підкатегорії спеціальностей чи освітніх програм, у яких містяться курси, що вивчаються за цією освітньою програмою.

Ці два, на перший погляд, рівноцінні підходи на практиці виявляються зовсім не рівноцінними. Великим недоліком другого підходу є дублювання курсів, що викладаються більше ніж в одній освітній програмі. Особливо це стосується дисциплін, що викладаються масово: математика, філософія, іноземна мова тощо. Вони мають бути представлені окремим курсом у категорії курсів кожної освітньої програми, у якій ця дисципліна вивчається. А кількість освітніх програм, за якими навчаються студенти, наприклад, у Київському національному університеті будівництва і архітектури, перевищує сотню. Отже, доводиться створювати, а потім регулярно оновлювати десятки однакових чи майже однакових копій курсів філософії чи математики в категорії курсів кожної з таких освітніх програм.

Звісно, викладачу краще працювати якщо не з єдиною, то принаймні з кількома версіями (якщо ці версії дійсно кардинально відрізняються) курсу, що зберігається в категорії курсів його кафедри. Некардинальні відмінності в програмах викладання дисциплін для різних освітніх програм краще реалізувати в межах одного спільного курсу за допомогою засобів обмеження доступу (*Access restrictions*), наприклад, від належності студентів до певної групи. Якщо додати в ці групи також і викладачів, встановити в курсі режим *Окремі групи* і зняти дозвіл *moodle/site:accessallgroups* у налаштуваннях ролі викладача, то кожний викладач бачитиме в цьому спільному курсі тільки своїх студентів, а вони бачитимуть лише ті елементи курсу, які призначені для їх групи.

На кафедрах фізики, хімії та інших, де викладаються схожі одна на одну за змістом дисципліни (бо власне за цією ознакою кафедри і формуються), викладачі мають змогу спільно створювати і спільно використовувати тестові питання, якщо зберігати їх не в кожному курсі, а на рівні категорії курсів кафедри. Оскільки в категорії курсів освітньої програми схожих дисциплін значно менше, ніж у категорії курсів кафедри, то і можливостей спільного створення і використання тестових питань теж менше.

Від вибору структури категорій курсів залежить і організація надання прав доступу. Наприклад, менеджер категорії курсів факультету надає своєму декану доступ

до всіх курсів цієї категорії і призначає кафедральних менеджерів, кожен з яких надає своєму завідувачу доступ до всіх курсів підкатегорії курсів кафедри, а кожному викладачу – доступ до тих курсів, які він викладає. Але чи зручно менеджеру категорії курсів освітньої програми призначати на ці курси викладачів різних кафедр та їхніх завідувачів (не одного, а багатьох) і надавати завідувачу доступ лише до тих курсів, які ведуть викладачі його кафедри? Очевидно, що ні. Викладачі більше спілкуються в межах кафедри, їх педагогічне навантаження розподіляється через кафедри, а не освітні програми. Тому ці та інші питання адміністрування сайту теж краще вирішувати на рівні кафедр, ніж на рівні освітніх програм.

Саме тому курси на сайті Moodle краще розміщувати в категоріях курсів *Факультет – Кафедра,* ніж *Факультет – Освітня програма*. Звісно, потреба мати на сайті списки курсів, згруповані за освітніми програмами, теж є. Але реалізувати їх краще не як список курсів, розташованих в одній спільній категорії, а як список гіперпосилань на ці курси.

Аналіз останніх досліджень і публікацій. Як показано нами в роботі [10], роль таких гіперпосилань можуть виконувати елементи курсу *Субкурс*, які імпортують в журнал оцінок курсу, де розташований субкурс, оцінку результату того курсу, на який цей субкурс посилається (рис. 1). Завдяки цьому можна зібрати в журналі оцінок певного курсу, який ми називаємо *службовим*, результуючі оцінки з усіх дисциплін, які вивчають студенти певної освітньої програми і року навчання, та розрахувати на їх основі поточний рейтинг кожного студента, який система Moodle оновлюватиме щохвилини. Це дає змогу реалізувати функції електронного деканату і приблизно на порядок зменшити обсяг роботи адміністратора, пов’язаний з переведенням студентів на наступний рік навчання [10]. Для цього у курсах навчальних дисциплін можна використати спосіб зарахування, який називається *Мета-курс,* і зв’язати його з тим самим службовим курсом, у який імпортуються оцінки. Тоді списки студентів у курсах навчальних дисциплін будуть синхронізовані зі списком студентів у службовому курсі так, що при зарахуванні студента в службовий курс або його відрахуванні з нього те саме відбуватиметься в усіх пов’язаних з ним курсах навчальних дисциплін. Отже, для переведення студентів на новий навчальний рік достатньо оновити контингент лише в службових курсах.

Рис. 1. Взаємодія службових курсів з курсами навчальних дисциплін

У такий спосіб у пропонованій структурі сайту, крім курсів навчальних дисциплін, створюються також службові курси. У кожний службовий курс записують студентів однієї освітньої програми і одного року навчання. Це може бути одна або декілька паралельних академічних груп. Службові курси не містять жодних навчальних матеріалів. Їх роль полягає в тому, щоб експортувати в курси навчальних дисциплін списки студентів, а в зворотному напрямку імпортувати та відобразити в спільному журналі оцінки результатів навчання студентів з усіх дисциплін, які вони зараз вивчають, та розрахувати на їх основі поточний рейтинг кожного студента. Також службовий курс можна використовувати як центр спілкування записаних у нього студентів. Тут для них розміщуються форуми, оголошення деканату, проводяться різні опитування та інші види роботи зі студентами, які не належать до конкретних навчальних дисциплін.

Отже, описана вище структура сайту має значні переваги над сайтами, які містять лише курси навчальних дисциплін. Однак спробуємо оцінити обсяг робіт для її створення і подальшого супроводу.

Якщо в університеті є 100 освітніх програм, то для кожної з них треба створити свою категорію курсів, у якій розмістити службові курси в кількості, що дорівнює числу років навчання: 4 для бакалаврів, 2 для магістрів, 3 для докторів філософії. Сумарно понад 300 службових курсів. Якщо впродовж навчального року за кожною освітньою програмою вивчається в середньому десяток дисциплін, то сумарна кількість субкурсів, які треба розмістити в службових курсах, перевищує 3000. Якщо створювати кожний із них «вручну», то треба буде ввести його назву, що співпадає з назвою дисципліни, створити посилання на її курс та задати деякі інші параметри. А ще кожний курс навчальної дисципліни треба зв’язати мета-курсами зі службовими курсами освітніх програм, у яких вона вивчається. Кількість мета-курсів така ж, як і субкурсів, – понад 3000.

Крім робіт зі створення і початкового наповнення службових курсів та їх категорій, щороку треба виконувати значні обсяги робіт по їх супроводу: записувати на курси першокурсників, видаляти випускників, а решту переводити на наступний рік навчання, тобто відраховувати з курсів минулого року і зараховувати на курси поточного.

Як бачимо, створення і підтримка описаної вище структури сайту потребує значного обсягу робіт. Тому їх виконання традиційними засобами адміністратору не завжди під силу. Звісно, всю цю роботу він може розподілити між багатьма виконавцями: факультетськими і кафедральними менеджерами сайту та всіма викладачами. Але тоді доведеться навчити кожного виконувати свою частку цієї роботи й контролювати правильність і вчасність її виконання. Це може призвести не до зменшення, а до збільшення обсягу робіт при однозначно гіршому результаті, навіть якщо йдеться про затрати часу самого адміністратора, не кажучи вже про сумарні затрати часу всіх учасників. Практика показує, що кращий результат дає централізоване виконання подібних робіт одним кваліфікованим виконавцем, який має змогу застосувати досконаліші засоби і способи роботи.

До таких засобів належить Moosh [11] – програма для адміністрування сайту Moodle через командний рядок. Як видно з публікацій [6] – [9], більшість користувачів саме так нею і користуються. Виконання команд адміністрування через командний рядок дійсно пришвидшує процес, але всього лише замінює один спосіб ручного виконання команд іншим, хоч і продуктивнішим, але теж ручним. Принаймні в зазначених публікаціях не згадується про використання можливості запускати на виконання файли, у яких записано багато таких команд. А це не тільки пришвидшує процес, а й переводить його на принципово вищий рівень – адміністрування системи

Moodle за допомогою програмування. Справді, записаний у такому файлі набір команд цілком може вважатися програмою, бо він може містити не тільки команди Moosh та команди операційної системи, а й цикли та інші елементи програмування, доступні в консолі Unix.

Однак процес написання програми, що містить сотні, а може й тисячі таких команд, теж займає багато часу, отже, теж потребує автоматизації.

Метою дослідження є розробка способів подальшого пришвидшення і покращення якості адміністрування системи Moodle шляхом автоматизації створення командних файлів для для Moosh або файлів csv для Moodle на прикладі виконання робіт зі створення, початкового наповнення і подальшого адміністрування описаної вище структури сайту. Також метою цього дослідження є огляд і тестування нових плагінів, які забезпечують функціонування сайту зазначеної структури.

2. РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

Програмні засоби. У якості основного інструменту для автоматизації виконання описаних вище робіт будемо використовувати створену Томашем Мурасом (Tomasz Muras) утиліту Moosh (розшифровується MOOdle SHell), яка дає змогу виконувати через командний рядок більшість команд адміністрування сайту Moodle. Детальні відомості про цю утиліту, яка поширюється за відкритою ліцензією GNU GPL, як її встановити і використовувати, набір її команд тощо доступні на ресурсі [11].

Для роботи з Moosh потрібен SSH-доступ до сайту Moodle, який можна реалізувати за допомогою ще однієї безкоштовної програми – клієнта віддаленого доступу PuTTY. Саме через командний рядок PuTTY спочатку інсталюємо саму Moosh, а потім, зайшовши в каталог файлів сайту, вводимо її команди.

Наприклад, щоб створити підкатегорії курсів з назвами «Бакалавр», «Магістр», «Доктор філософії» і «Кафедри» в категорії курсів факультету, що має ідентифікаційний номер 13, можна скористатися командою:

```
moosh -n category-create -p 13 -v 1 Бакалавр Магістр "Доктор філософії" Кафедри
```

У ній параметр *-v 1* вказує на те, що створені категорії будуть видимими (а не прихованими), а назву «*Доктор філософії*» взято в лапки, бо це назва однієї категорії, а не двох: «*Доктор*» і «*філософії*».

Тут і далі параметр *-n* інформує систему про те, що адміністратор, який зазвичай запускає Moosh від імені користувача *root*, розуміє наслідки можливих конфліктів прав доступу до файлів, які при виконанні деяких команд можуть створюватися в каталозі *moodledata* від імені користувача *root*, бо потім Moodle, що працює від імені користувача *www-data,* не зможе їх видалити. Щоб унеможливити такі конфлікти, розробник радить запускати Moosh від імені користувача `www-data: sudo -u www-data moosh` ... [11].

Як і в інших системах, що працюють з командним рядком, клавішами переміщення курсора вгору і вниз можна прокручувати на консолі команди, що виконувалися раніше, вносити в них виправлення і натисканням клавіші *Enter* повторно виконувати їх. Наприклад, виправивши в зазначеній вище команді 13 на інше число і натиснувши *Enter,* можна створити такі самі категорії курсів для іншого факультету. У результаті перелік категорій курсів на титульній сторінці сайту набуде вигляду, показаного на рис. 2.

Уже на цьому простому прикладі можна бачити велику економію часу при порівнянні зі створенням усіх цих категорій курсів через власний інтерфейс сайту

Moodle. Як зазначалося вище, ще більшого пришвидшення можна досягти, запускаючи на виконання файли, що містять багато таких команд. Але в цьому дослідженні ставиться ще вища мета – автоматизувати сам процес створення таких файлів. Крім подальшого пришвидшення робіт з адміністрування сайту, це покращить також якість, бо зменшить кількість помилок, які зазвичай виникають при ручному введенні команд або традиційному адмініструванні та навіть при традиційному програмуванні, тобто написанні тексту програми вручну.

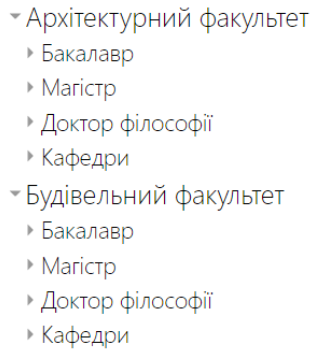


Рис. 2. Фрагмент структури категорій курсів на титульній сторінці сайту

Ще одним програмним засобом, який буде потрібний для цих робіт, є плагін *Configurable Reports* [4]. Він відкриває доступ до всіх таблиць бази даних Moodle і дає змогу виводити їх дані у вигляді звітів. Щоб ним скористатися, треба знати мову SQL-запитів до баз даних[1], а також те, як саме пов'язані між собою таблиці в базі даних Moodle[2]. Ще треба вміти переглядати і аналізувати структуру і вміст цих таблиць за допомогою таких програм, як-от: phpMyAdmin або Adminer чи плагіну Moodle Adminer [4].

Наприклад, про те, що ідентифікаційний номер (ID) згаданої вище категорії курсів архітектурного факультету дорівнює 13, можна дізнатися, увійшовши в цю категорію і побачивши в адресному рядку браузера число наприкінці її URL: *https://org2.knuba.edu.ua/course/index.php?categoryid=13*. Але, щоб не відкривати в такий спосіб кожну категорію, можна вивести ID і назви всіх категорій курсів сайту у вигляді звіту, задавши такий запит до бази даних:

```
SELECT cc.id, cc.name FROM prefix_course_categories cc
```

Тут *prefix_course_categories* – ім’я таблиці, що містить дані про категорії курсів сайту, яку для зручності тут позначено скороченим іменем (аліасом) *cc*, а *cc.id, cc.name* – імена стовпців цієї таблиці, що містять ID і назви категорій курсів. Якщо до списку полів додати ще й *cc.parent*, то для кожної категорії курсів знатимемо також ID її батьківської категорії.

При виконанні робіт будемо також використовувати Microsoft Word і Excel або LibreOffice для *злиття документів* і (у разі потреби) – для програмування на VBA або LibreOffice Basic.

---

[1] Див., наприклад: Г.С. Погромська. Побудова запитів на мові SQL: Навчальний посібник. Миколаїв. МНУ ім. В.О. Сухомлиського, 2014. [Електронний ресурс]. Режим доступу: https://dspace.mdu.edu.ua/jspui/bitstream/123456789/64/1/Погромська_Побудова%20запитів%20на%20мові%20SQL.Pdf

[2] Див., наприклад: Database schema introduction. [Електронний ресурс]. Режим доступу: https://docs.moodle.org/dev/Database_schema_introduction

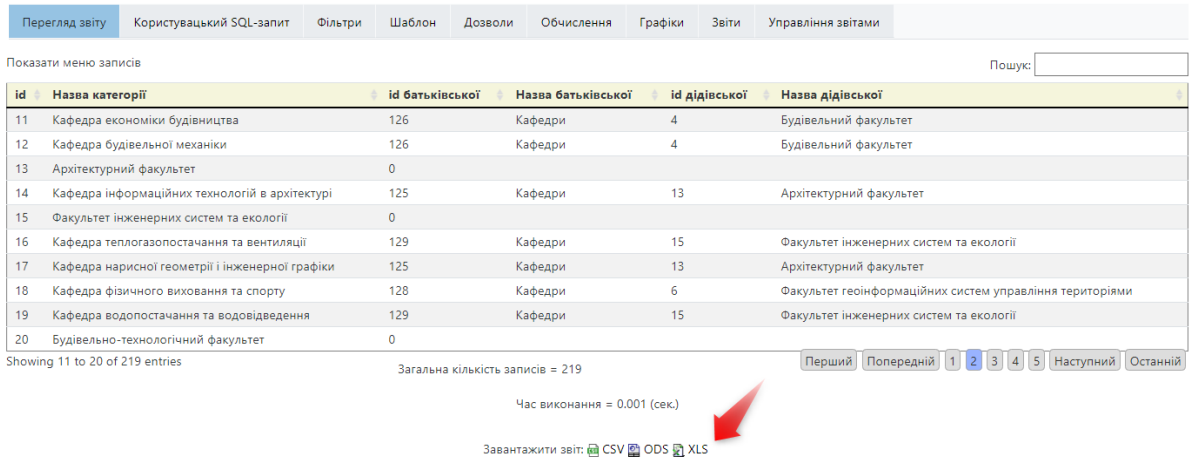
Вважаємо (оскільки саме така ситуація найчастіше зустрічається на практиці), що сайт Moodle встановлено на сервері з Unix-подібною операційною системою, а роботи з його адміністрування здійснюється на комп'ютері з операційною системою Windows.

Створення категорій курсів. Щоб створити в категоріях курсів *Бакалавр, Магістр* і *Доктор філософії* підкатегорії курсів відповідних освітніх програм (а за потреби також і підкатегорії курсів кафедр в категоріях *Кафедри*), треба знати ID цих категорій курсів. Оскільки назви категорій *Бакалавр, Магістр* і т. д. повторюються на різних факультетах, виведемо звіт, у якому крім ID і назви кожної категорій курсів відобразимо також ID і назву її батьківської і дідівської категорій. Для цього використаємо запит:

```
SELECT cc.id, cc.name AS "Назва категорії",
cc.parent AS "ID батьківської", pcc.name AS "Назва батьківської",
pcc.parent AS "ID дідівської", ppcc.name AS "Назва дідівської"
FROM prefix_course_categories cc
LEFT JOIN prefix_course_categories pcc ON cc.parent = pcc.id
LEFT JOIN prefix_course_categories ppcc ON pcc.parent = ppcc.id
ORDER BY cc.id
```

У результаті одержимо звіт (рис. 3), що відображає три рівні категорій курсів сайту.

Щоб створити для кожної освітньої програми її категорію курсів, сформуємо таблицю Excel (рис. 4), у яку запишемо код освітньої програми, її назву, а також ID батьківської категорії курсів, у якій ця категорія створюється.



id	Назва категорії	id батьківської	Назва батьківської	id дідівської	Назва дідівської
11	Кафедра економіки будівництва	126	Кафедри	4	Будівельний факультет
12	Кафедра будівельної механіки	126	Кафедри	4	Будівельний факультет
13	Архітектурний факультет	0			
14	Кафедра інформаційних технологій в архітектурі	125	Кафедри	13	Архітектурний факультет
15	Факультет інженерних систем та екології	0			
16	Кафедра теплогазопостачання та вентиляції	129	Кафедри	15	Факультет інженерних систем та екології
17	Кафедра нарисної геометрії і інженерної графіки	125	Кафедри	13	Архітектурний факультет
18	Кафедра фізичного виховання та спорту	128	Кафедри	6	Факультет геоінформаційних систем управління територіями
19	Кафедра водопостачання та водовідведення	129	Кафедри	15	Факультет інженерних систем та екології
20	Будівельно-технологічний факультет	0			

Рис. 3. Фрагмент звіту про структуру категорій курсів та їх ідентифікаційні номери

	A	B	C
1	**Код**	**Назва ОП**	**pID**
8	Б131.1	Інженерія логістичних систем	223
9	Б131.2	Інженерна механіка	223
10	Б133	Галузеве машинобудування	223
11	М122	Управління проєктами	104
12	М123	Комп'ютерна інжерерія	104
13	М125	Кібербезпека	104
14	М126.1	Інформаційні системи та технології	104
15	М126.2	Штучний інтелект. Когнітивні технології	104
16	М131	Інженерія логістичних систем	104

Рис. 4. Фрагмент таблиці для формування категорій курсів освітніх програм

Код освітньої програми може містити літеру: *Б* – бакалавр, *М* – магістр, *Р* (від PhD) або *Ф* – доктор філософії, (але не *Д*, бо тоді при сортуванні за алфавітом він опиниться між бакалавром і магістром). Після літери доцільно вказати число – код спеціальності або спеціалізації. Якщо для неї є кілька освітніх програм, то в їхній код треба додати *.1, .2* або *а, б* тощо, щоб код кожної освітньої програми був унікальним, оскільки він використовуватиметься для формування інших назв, таких як коротка назва курсу, яка на сайті Moodle теж повинна бути унікальною. Це важлива вимога, бо її невиконання призведе до спроби створити курс з уже існуючою короткою назвою, що спричинить помилку, дуже небажану під час виконання команд в автоматичному режимі.

Для заповнення стовпця *pID,* можна скористатися або самим показаним на рис. 3 звітом, зокрема його полем *Пошук*, або файлом Excel, у який цей звіт експортовано. Там теж можна здійснювати пошук, а ще виконувати операції сортування і фільтрації.

Тепер для кожного рядка таблиці, показаної на Рис. 4, потрібно сформувати команду

```
moosh -n category-create -p pID -v 1 "Код Назва ОП"
```

Для цього можна не вдаватися до програмування, а скористатися функцією *Злиття документів* у Word. Щоб виконати злиття, треба до документу Word, який називають *основним документом*, підключити документ Word, Excel тощо, що містить таблицю, яку називають *джерелом даних*. Тоді з'явиться можливість вставляти в основний документ *поля злиття*, імена яких співпадають з назвами стовпців у джерелі даних. Поля злиття (щоб відрізняти їх від звичайного тексту) тут і далі виділено сірим фоном.

У результаті злиття основного документа, що містить наведений вище шаблон команди, і джерела даних, показаного на рис. 4, створюється вихідний документ, у який вміст основного документу копіюється стільки разів, скільки рядків є в джерелі даних. І щоразу на місце полів злиття підставлятимуться дані з відповідного рядка. У такий спосіб у вихідному документі буде сформовано такий набір команд:

```
moosh -n category-create -p 223 -v 1 "Б131.1 Інженерія логістичних систем"
moosh -n category-create -p 223 -v 1 "Б131.2 Інженерна механіка"
moosh -n category-create -p 223 -v 1 "Б133 Галузеве машинобудування"
moosh -n category-create -p 104 -v 1 "М122 Управління проєктами"
moosh -n category-create -p 104 -v 1 "М123 Комп'ютерна інженерія" і т. д.
```

Є кілька типів вихідних документів. Для цього випадку краще вибрати документ типу *Каталог,* у якому не створюються непотрібні тут розриви сторінок.

Далі треба:

- скопіювати сформовані описаним вище способом команди з документу Word у файл текстового документу, що має кодування UTF-8, назвавши цей файл, наприклад, *prog.sh*;
- додати в перший рядок цього файлу коментар *#!/bin/bash*, який інформує Unix-подібну операційну систему про те, що цей файл містить команди для виконання;
- скопіювати файл на сервер, наприклад, у каталог */tmp*;
- конвертувати цей файл у формат Unix-подібних операційних систем[3], виконавши *dos2unix /tmp/prog.sh*;

[3] Детальніше про це див.: Dos2Unix / Unix2Dos - Text file format converters. [Електронний ресурс].

– запустити цей файл на виконання командою *bash /tmp/prog.sh*.

Завершуючи етап створення категорій курсів, зазначимо, що зазвичай ми починаємо роботу не з порожнім сайтом, а з тим, який вже має якесь наповнення. Тому, можливо, не всі описані вище роботи доведеться виконувати саме так. Наприклад, якщо якісь категорії курсів на сайті вже є, то замість їх створення буде достатньо перемістити в них як окремі курси, так і цілі категорії курсів разом з підкатегоріями і курсами, які вони містять. Для цього в розпорядженні адміністратора сайту Moodle є зручні засоби керування курсами і категоріями.

Якщо освітніх програм в університеті не дуже багато, то, може, й не варто створювати окрему категорію курсів для кожної з них чи мати окремі категорії курсів для бакалаврів, магістрів і докторів філософії, чи навіть ділити освітні програми за факультетами, а розміщувати службові курси різних освітніх програм у спільних (чи навіть в одній) категоріях курсів. Оскільки в кожній категорії курси сортуються за алфавітом, то такий об'єднаний список службових курсів буде впорядковано насамперед за освітнім рівнем, в другій – за номером спеціальності й у третій – за роком навчання.

Створення службових курсів розпочнемо зі створення джерела даних. Для цього показаний на рис. 3 і оновлений після створення категорій курсів освітніх програм звіт експортуємо в таблицю Excel і додаємо в неї ще одну колонку під назвою *Код*, у якій за допомогою формули *=LEFT(B2;FIND(" ";B2)-1)* виведемо код освітньої програми (перше слово в стовпці *Назва категорії*). Використання цього коду в назвах службових курсів полегшить пошук і роботу з ними. Для подальшого використання збережемо цю таблицю у файлі *Звіти.xlsx* на робочому аркуші *Категорії*.

Основний документ для формування команд створення службових курсів освітніх програм бакалаврів 1–4 років навчання можна записати так:

```
moosh -n course-create --category ID --fullname "*Код 1-й рік навчання" "*Код 1-й курс"

moosh -n course-create --category ID --fullname "*Код 2-й рік навчання" "*Код 2-й курс"

moosh -n course-create --category ID --fullname "*Код 3-й рік навчання" "*Код 3-й курс"

moosh -n course-create --category ID --fullname "*Код 4-й рік навчання" "*Код 4-й курс"
```

або дещо компактніше, з використанням команди циклу:

```
for ((i = 1; i <=4; i++ )); do
      moosh -n course-create --category ID --fullname "*Код $i-й рік навчання"
      "*Код $i-й курс"
done
```

Щоправда ці команди створюють зовсім порожні курси, а нам може знадобитися, щоб вони вже мали якесь початкове наповнення: форуми для спілкування студентів, різні корисні ресурси і посилання тощо. Це початкове наповнення може зберігатися в якомусь шаблоні. Ним може служити певний курс чи файл резервної копії курсу.

На жаль, у параметрах зазначених вище команд немає змоги задати ім'я такого шаблону. Доведеться після кожної з них виконувати команду

```
moosh -n course-restore -e /tmp/shablon.mbz ID,
```

Режим доступу: https://dos2unix.sourceforge.io/

яка відновлює резервну копію курсу, що зберігається у файлі */tmp/shablon.mbz,* в існуючий курс із зазначеним у команді ID. Це саме той ID, який виводиться системою на консоль після успішного виконання команди створення курсу.

Якби ми вводили команди з клавіатури, то, побачивши цей ID після виконання першої команди, вписали б його у другу. Але наші команди не вводяться з клавіатури, а мають бути заздалегідь записаними у файл. Тому виникає питання, як ID, який буде виведено на консоль у результаті виконання однієї команди, використати в наступних?

Для цього існує такий механізм. Значення, що виводиться системою при виконанні першої команди, треба записати в якусь змінну:

```
<Змінна>=`<Команда1>`
```

а потім підставити значення цієї змінної в наступні команди:

```
<Команда2> … $<Змінна>…
<Команда3> … $<Змінна>…
```

Головне, щоб при виконанні першої команди система виводила на консоль лише потрібний нам ID і нічого зайвого (повідомлення про помилки, попередження тощо). Якщо, незважаючи на них, треба продовжити виконання команд, то доведеться у тексті цих повідомлень програмним чином віднайти ID і в наступні команди підставляти лише його.

У нашому випадку ми маємо справу з командою *course-create*. Помилка при її виконанні, найімовірніше, може статися через те, що в системі вже є курс з такою короткою назвою. Таку помилку ігнорувати не можна. Треба з'ясувати, чому вона виникла, виправити її й тільки після цього продовжити.

Отже, формування службових курсів за певними шаблоном можна реалізувати так:

```
for ((i = 1; i <=4; i++ )); do
      m=`moosh -n course-create --category ID --fullname "*Код $i-й рік навчання" 
      "*Код $i-й курс"`
      moosh -n course-restore -e --ignore-warnings /tmp/shablon.mbz $m
done
```

Тут застосовано додатковий параметр *--ignore-warnings,* завдяки якому можливі попередження про те, що резервну копію курсу-шаблону створено на вже застарілій версії Moodle, будуть проігноровані.

Створювати службові курси можна не тільки командами Moosh. Таку можливість надає і сам Moodle, якщо увійти в меню *Керування сайтом – Курси – Завантажити курси,* і там ввести файл у форматі csv, який у першому рядку містить заголовки стовпців таблиці:

```
fullname;shortname;category;templatecourse
```

а в решті рядків – її наповнення, що формується шляхом злиття з таким основним документом:

```
*Код 1-й рік навчання;*Код 1-й курс;ID;Шаблон СКБ
*Код 2-й рік навчання;*Код 2-й курс;ID;Шаблон СКБ
*Код 3-й рік навчання;*Код 3-й курс;ID;Шаблон СКБ
*Код 4-й рік навчання;*Код 4-й курс;ID;Шаблон СКБ
```

Тут (і взагалі у файлах csv) символ «;» – це роздільник, що відділяє один стовпець таблиці від іншого, а *Шаблон СКБ* – це коротка назва курсу, вміст якого копіюється в усі створювані в такий спосіб службові курси бакалаврів. При потребі можна

використати різні шаблони для різних років навчання і освітніх рівнів. Файли у форматі csv зручно переглядати і при потребі редагувати в програмі Excel або LibreOffice Calc.

Незалежно від того, який основний документ використовується і який файл формується (файл з командами для Moosh чи файл у форматі csv для Moodle), під час цього злиття у джерелі даних треба встановити фільтр, щоб у злитті брали участь тільки ті рядки, у яких батьківська категорія курсів має значення *Бакалавр* (рис. 5).

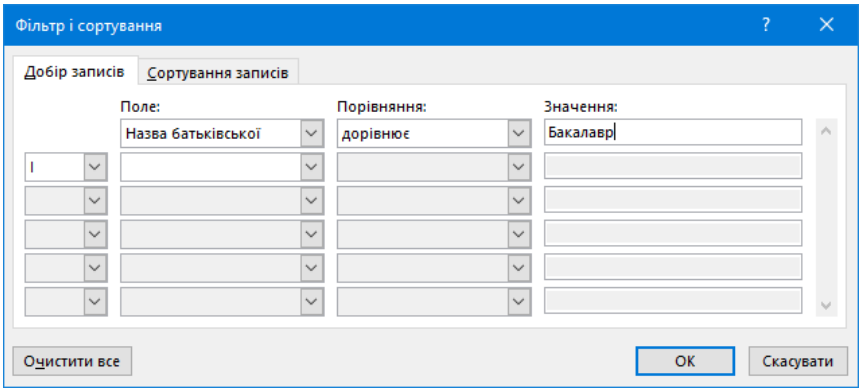


Рис. 5. Фільтрація записів у джерелі даних під час створення службових курсів

Наголошуємо, що йдеться про фільтрацію, яка здійснюється саме *під час* злиття документів у Word, а не в Excel до початку цього процесу.

Для створення службових курсів магістерських освітніх програм з наведених вище основних документів треба видалити рядки 3-го і 4-го років навчання (або встановити для параметра циклу верхню межу *i <=2*), а фільтрацію під час злиття виконувати за умовою *Назва батьківської = Магістр*. Аналогічно створюються і службові курси освітніх програм доктора філософії.

Щоб список курсів в її категорії сортувався за алфавітом автоматично, рекомендуємо встановити на сайті плагін Re-sort Courses [4], бо без нього щойно створені курси відображатимуться в черговості протилежній тій, у якій вони створювалися, тобто курс, створений останнім, знаходитиметься вгорі списку. Тоді їх доведеться сортувати вручну.

Відзначимо, що всі назви створених тут службових курсів починаються символом *. Це зроблено для того, щоб за цією ознакою можна було відрізнити службові курси від курсів навчальних дисциплін.

Для подальшої роботи нам знадобляться окремі списки обох типів курсів, із зазначенням їх назв, ID і назв всіх трьох рівнів категорій курсів, у яких вони знаходяться. Службові курси, відсортовані в алфавітному порядку за їх короткими назвами, можна вивести за допомогою такого запиту до плагіна конфігурованих звітів:

```
SELECT c.shortname AS "Коротка назва",
c.id AS "cID", c.fullname AS "Службовий курс",
cc.id AS "ccID", cc.name AS "Назва категорії",
cc.parent AS "ID батьківської", pcc.name AS "Назва батьківської",
pcc.parent AS "ID дідівської", ppcc.name AS "Назва дідівської"
FROM prefix_course c
LEFT JOIN prefix_course_categories cc ON c.category = cc.id
LEFT JOIN prefix_course_categories pcc ON cc.parent = pcc.id
LEFT JOIN prefix_course_categories ppcc ON pcc.parent = ppcc.id
```

```
WHERE (pcc.name = "Бакалавр" OR pcc.name = "Магістр" OR pcc.name = "Доктор
філософії") AND c.fullname LIKE '*%'
ORDER BY c.shortname
```

Запит для виводу курсів навчальних дисциплін має такий вигляд:

```
SELECT c.id AS "cID", c.fullname AS "Дисципліна", c.shortname AS "Коротка назва",
cc.id AS "ccID", cc.name AS "Назва категорії",
cc.parent AS "ID батьківської", pcc.name AS "Назва батьківської",
pcc.parent AS "ID дідівської", ppcc.name AS "Назва дідівської"
FROM prefix_course c
LEFT JOIN prefix_course_categories cc ON c.category = cc.id
LEFT JOIN prefix_course_categories pcc ON cc.parent = pcc.id
LEFT JOIN prefix_course_categories ppcc ON pcc.parent = ppcc.id
WHERE pcc.name = "Кафедри" AND c.fullname NOT LIKE '*%'
ORDER BY c.fullname
```

Зауважимо, що в цей звіт потрапляють лише курси навчальних дисциплін, розташованих безпосередньо в категорії курсів кафедри, а не в її підкатегоріях. Тому курси, які зараз не викладаються, але ще можуть бути корисними в майбутньому, треба перемістити саме в підкатегорії.

Обидва звіти експортуємо у Excel і зберігаємо у тому самому файлі *Звіти.xlsx* на аркушах *Службові* і *Дисципліни*.

У категорії курсів кожної кафедри доцільно мати ще один вид службового курсу. Він називається так само, як і кафедра, наприклад, **КАФЕДРА ФІЗИКИ.* Цю назву дійсно варто виділити великими літерами, щоб вона відрізнялася від назв розташованих нижче курсів навчальних дисциплін, бо при сортуванні за алфавітом цей курс завжди очолюватиме список завдяки символу *.

За аналогією зі службовими курсами для студентів, цей службовий курс має бути центром спілкування викладачів цієї кафедри, яким у ньому надається роль *Студент*. Тут розміщуються різні оголошення для викладачів, форуми, корисні посилання і файли тощо. Надання викладачами звітних та інших документів (індивідуальні плани, робочі програми дисциплін тощо) краще реалізувати в цьому курсі через діяльність типу *Завдання*. Це полегшить контроль за процесом збору таких документів і впорядкує їх подальше зберігання безпосередньо в діяльностях *Завдання*. Якщо йдеться про надання викладачами документів, доступ до яких повинні мати їх колеги, то можна використати діяльність типу *База даних* і т. д. За ці діяльності можна навіть виставляти оцінки, щоб з їх допомогою вести облік виконання викладачами відповідних робіт.

Створення і початкове наповнення цих службових курсів може здійснюватися аналогічно.

Створення субкурсів і мета-курсів, які пов’язують службові курси з курсами навчальних дисциплін (рис. 1), розпочнемо зі створення джерела даних. Для цього скопіюємо з освітньої програми таблицю, що містить назви її навчальних дисциплін, семестри, у яких вони викладаються, кількість годин або кредитів тощо. У неї додамо і заповнимо такі стовпці:

- ***sID*** – ID службового курсу того року навчання, у якому викладається ця дисципліна;
- ***semestr1*** – поле, у якому ставиться 1, якщо дисципліна викладається в першому (осінньому) семестрі, 0 – якщо ні;
- ***semestr2*** – те саме для другого (весняного) семестру;
- ***dID*** – ID курсу навчальної дисципліни.

Якщо дисципліна викладається впродовж кількох навчальних років, то в цій таблиці вона має бути представлена не одним, а кількома рядками з відповідними даними в кожному з них.

На сайті може бути чимало курсів з однаковими чи подібними назвами, і адміністратор навряд чи знає, який із них вивчається в кожній освітній програмі. Тому відповідальність за заповнення стовпця ***dID*** має покладатися на гарантів освітніх програм. Адміністратор може лише проконтролювати правильність його заповнення, порівнюючи назви дисциплін в освітній програмі з назвами дисциплін у звіті, які мають ID=***dID***. Для зручності перевірки ці назви можна відобразити в додатковому стовпці за допомогою формули *=VLOOKUP(G2;'[Звіти.xlsx]Дисципліни'!$A:$B;2;FALSE)*. Щоб ця формула працювала, значення в останньому лівому стовпці вказаного у ній звіту треба перевести в числовий формат і відсортувати звіт за їх зростанням.

Перш ніж перейти до формування основного документу для злиття, з'ясуємо, у якій послідовності доцільно розміщувати субкурси в службових курсах.

Дисципліни, які вивчаються студентами впродовж навчального року, розділимо на три групи: 1) ті, що вивчаються лише в першому семестрі; 2) ті, що вивчаються в обох семестрах; 3) ті, що вивчаються тільки в другому семестрі. Якщо впорядкувати субкурси за належністю до цих груп, то в кожному семестрі перелік дисциплін, які зараз вивчаються, утворюватиме суцільний масив 1) + 2) або 2) + 3). У межах кожної з груп назви дисциплін сортуватимемо за алфавітом (рис. 6).

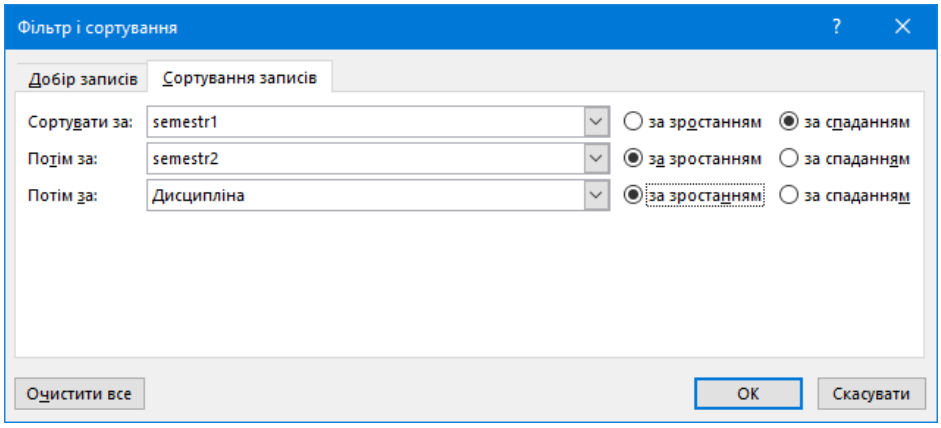


Рис. 6. Сортування записів у джерелі даних під час створення субкурсів

Основний документ, що використовється для цього злиття, повинен містити описані нижче команди.

Створення субкурсу:

```
m=`moosh -n activity-add --name "semestr1semestr2 Дисципліна" subcourse sID`
moosh -n activity-config-set activity $m subcourse refcourse dID
moosh -n activity-config-set activity $m subcourse intro dID
moosh -n activity-config-set activity $m subcourse instantredirect 1
moosh -n activity-config-set activity $m subcourse coursepageprintgrade 1
moosh -n activity-config-set activity $m subcourse completioncourse 1
```

Як бачимо, назва субкурсу складається з даних про семестри, у яких викладається ця дисципліна, і її назви, наприклад, *10 Опір матеріалів, 11 Фізика, 01 Правознавство.* ID курсу, на який посилається субкурс і з якого він імпортує результуючі оцінки, задається параметром *refcourse,* у який записується числове значення *dID.* Щоправда у

вікні редагування налаштувань субкурсу цей параметр відображається не числом, а назвою цього курсу. Тому для зручності число *dID* можна записати також у поле *intro.* Це поле опису, яке можна відображати під назвою кожного елементу курсу, але за замовчуванням воно не відображається. Наступні параметри забезпечують безпосередній перехід за посиланням на зазначений курс, відображення його результуючої оцінки не тільки в журналі, а й на сторінці службового курсу, а також те, що діяльність субкурс вважатиметься завершеною тоді, коли такий же статус матиме курс, на який цей субкурс посилається.

Створення мета-курсу за допомогою Moosh поки що неможливе, бо така команда в його наборі команд відсутня. Однак у Moosh передбачена можливість додавання користувачами власних команд. Користуючись наведеними в [11] рекомендаціями і взявши за зразок програмний код подібної команди, а саме *cohort-enrol,* можна написати і додати в Moosh код власної команди *meta-enrol,* з використанням якої створення в курсі ***dID*** мета-курсу, що імпортує списки студентів зі службового курсу ***sID***, матиме такий вигляд:

```
moosh -n meta-enrol -c dID sID
```

Альтернативою написанню коду власної команди може бути використання плагіну Upload enrolment methods [4], який дає змогу створювати мета-курси за допомогою файлу csv з таким заголовком

```
operation,method,shortname,metacohort,disabled,group,role
```

і наповненням, створеним за допомогою злиття з таким основним документом:

```
add,meta,dshortname,sshortname,0,,student
```

Тут *dshortname* і *sshortname* – короткі імена курсу навчальної дисципліни й службового курсу, які за аналогією можна додати в джерело даних із відповідних звітів за допомогою функції *VLOOKUP.*

Зауважимо, що цей плагін у якості роздільника стовпців у файлі csv використовує тільки кому, тоді як наша версія Excel використовує для цього крапку з комою, бо комою у нас прийнято відділяти цілу частина числа від дробової. Щоб не переналаштовувати Excel з нашого стандарту на англійський і навпаки, переглядати такі файли зручніше в програмі LibreOffice, де передбачено вибір роздільника при відкритті кожного файлу csv.

Заповнення в курсах полів semestr1 і semestr2 – це ще одна корисна робота, яку можна виконати з цим же джерелом даних. Справа в тому, що при здійсненні моніторингу активності студентів і викладачів на курсах треба враховувати, у яких семестрах викладається кожний курс. Для цього в курсах доцільно створити спеціальні поля на кшталт «прапорець»: *semestr1* і *semestr2,* захистивши їх від випадкової чи навмисної зміни викладачами (бо це може вивести їхні курси з-під такого моніторингу) і заповнивши їх однойменними даними з освітніх програм. А для цього треба в меню *Керування сайтом – Курси – Завантажити курси* в режимі *Тільки оновити існуючі курси, Оновити тільки даними з файлу* завантажити файл csv з таким заголовком

```
shortname;customfield_semestr1;customfield_semestr2
```

і наповненням, створеним за допомогою злиття з таким основним документом:

```
dshortname;semestr1; semestr2
```

На завершення відзначимо, що описані вище операції не обов’язково виконувати для кожної освітньої програми окремо. Після заповнення і перевірки таблиці джерела даних, можна об’єднати кілька таких таблиць в одну, і один раз виконати для неї злиття документів, завантаження результату на сервер, його конвертування і виконання. Те ж стосується створення і використання файлів csv.

Зарахування студентів у службові курси та відрахування з них – це робота, яка, на відміну від розглянутих вище, здійснюється не тільки під час створення сайту з показаною на рис. 1 структурою, а й під час його подальшої експлуатації, адже щороку треба зараховувати на курси першокурсників, видаляти випускників, а решту переводити на наступний рік навчання, тобто відраховувати з курсів минулого року й зараховувати на курси поточного. Як зазначалося вище, все це робиться через службові курси. Сформовані там списки студентів потім експортуються в усі курси навчальних дисциплін за допомогою мета-курсів.

Перш ніж розглянути це питання, зупинимось на способах кодування назв академічних груп. У наших університетах (на відміну від школи) немає єдиного способу їх кодування на кшталт: 1-й А, 7-й Б тощо. У кожному університеті є власна система кодування, причому нерідко назви груп, подібно до назв шкільних класів, змінюються з року в рік. Так, у Київському університеті будівництва і архітектури назви академічних груп містять скорочену назву спеціальності, курс (тобто рік навчання) і порядковий номер групи. Наприклад ПЦБ-12 – це друга за номером академічна група, що навчається на першому курсі за спеціальністю *Промислове і цивільне будівництво*. Через рік ця ж група називатиметься вже ПЦБ-22 і т. д.

Така система є не зручною для використання на сайті, зокрема і через необхідність щороку змінювати назви груп. Тому ми запропонували паралельно з нею запровадити іншу систему, де назва групи складається з коду освітньої програми і року випуску. Наприклад, Б123-24 – це група, яка навчається на бакалавраті за спеціальністю *123 – Комп'ютерна інженерія* і випускається з університету в 2023/24 начальному році. Якщо таких груп не одна, а декілька, то до позначення Б123-24 можуть додаватися ще й порядковий номер групи, дані про форму навчання, наприклад, «з» – заочна тощо. Однак на зарахування студентів на курси ці додаткові дані не впливають.

Зауважимо, що додавати в назву групи саме рік випуску, а не вступу зручніше, бо освітні програми різних освітніх рівнів мають різну тривалість. А так ми маємо змогу, наприклад, починаючи з осені 2022 року, видаляти з сайту всі групи, у назві яких присутні символи -22, незалежно від того, хто це: бакалаври, магістри чи доктори філософії.

Зарахування студентів і аспірантів у службові курси здійснюватимемо за допомогою контингентів, які в інших варіантах українського перекладу ще називають когортами (cohortes), глобальними групами або гуртами. Повний список назв створених на сайті контингентів можна вивести у звіт таким запитом до бази даних:

```
SELECT ct.name FROM prefix_course_cohort ct
ORDER BY ct.name
```

Цей звіт треба експортувати в Excel і там видалити з нього рядки, які не є контингентами академічних груп, тобто контингенти викладачів тощо. Оскільки список відсортовано за алфавітом, зробити це доволі легко.

На аркуш зі списком академічних груп (рис. 7) додамо поля, у яких зазначимо поточний навчальний рік і тип операції: зарахування (1) чи відрахування (0). В залежності від них, а також від освітнього рівня і року випуску академічної групи в стовпці *Рік* за формулою

```
=$A$1+$A$2-MID(A5;FIND("-";A5;1)+1;2)+3-IF(LEFT(A5;1)="М";2;0)-
IF(LEFT(A5;1)="Ф";1;0)
```

розрахуємо спочатку рік навчання, для якого здійснюється це зарахування чи відрахування. А потім використаємо значення року, щоб в стовпці *СК* за формулою *=CONCATENATE("*";LEFT(A5;FIND("-";A5;1)-1);" ";B5;"-й курс")* сформувати коротку назву службового курсу, з яким виконується ця операція, а в стовпці *cID* за формулою *=VLOOKUP(C5;[Звіти.xlsx]Службові!$A:$B;2;FALSE)* – ID цього курсу.

	A	B	C	D	E	F	G	H
1	22	навчальний рік						
2	1	зарахування (1) чи відрахування (0)						
3								
4	Група	Рік	СК	cID	Команда			
5	Б123-25-1	1	*Б123 1-й курс	1750	moosh -n cohort-enrol -c 1750 "Б123-25-1"			
6	Б123-25-2	1	*Б123 1-й курс	1750	moosh -n cohort-enrol -c 1750 "Б123-25-2"			
7	Б123-24	2	*Б123 2-й курс	1751	moosh -n cohort-enrol -c 1751 "Б123-24"			
8	Б123-23	3	*Б123 3-й курс	1752	moosh -n cohort-enrol -c 1752 "Б123-23"			
9	Б123-22	4	*Б123 4-й курс	1753	moosh -n cohort-enrol -c 1753 "Б123-22"			
10	М125-23	1	*М125 1-й курс	1843	moosh -n cohort-enrol -c 1843 "М125-23"			
11	М125-22	2	*М125 2-й курс	1844	moosh -n cohort-enrol -c 1844 "М125-22"			
12	Ф123-24	1	*Ф123 1-й курс	1929	moosh -n cohort-enrol -c 1929 "Ф123-24"			
13	Ф123-23	2	*Ф123 2-й курс	1930	moosh -n cohort-enrol -c 1930 "Ф123-23"			
14	Ф123-22	3	*Ф123 3-й курс	1931	moosh -n cohort-enrol -c 1931 "Ф123-22"			

Рис. 7. Формування команд зарахування контингентів безпосередньо в Excel

Тепер таблицю на рис. 7 (якби не три рядки над назвами стовпців) можна було б використати як джерело даних для злиття з основними документами, що містять таку команду для зарахування контингентів:

```
moosh -n cohort-enrol -c cID "Група"
```

і якусь таку

```
moosh -n cohort-unenrol -c cID "Група"
```

для їх відрахування. Хоча, на жаль, другої команди в Moosh наразі немає. Точніше сама команда *cohort-unenrol* є, але можливості задати в ній у якості параметрів ID курсу й назву контингенту, який треба з нього видалити, поки що немає. Щоб це виправити, код другої команди можна дописати самому, взявши за зразок першу.

Замість виконання процедури злиття документів у Word, як це завжди робилося досі, у цьому випадку команди Moosh можна сформувати прямо в Excel, у стовпці *Команда* за допомогою формули *=CONCATENATE("moosh -n cohort-";IF($A$2=0;"un";""); "enrol -c ";D5;" ";CHAR(34);A5;CHAR(34))*. Тоді буде достатньо лише скопіювати цей стовпець у текстовий файл, переслати на сервер, конвертувати і виконати.

Альтернативою написанню власного коду команди Moosh тут теж може бути використання вже згадуваного вище плагіна Upload enrolment methods [4]. Він дає змогу як зараховувати, так відраховувати студентів за допомогою файлу csv, прописавши *add* або *del* в полі *operation*. Робити це можна і з мета-курсами, і з континентами, прописавши *meta* або *cohort* в полі *method*. Єдине що в полі *metacohort* треба прописати не ім'я контингенту (*name*), як у команді Moosh, а його ідентифікаційний номер (*idnumber*). Плагін, який ми далі використовуватимемо для автоматичного запису студентів у контингенти, хоча і може створювати нові контингенти автоматично, однак за таких умов поле *idnumber* він нічим не заповнює. Тому, щоб скористатися для цього зазначеним плагіном, контингенти академічних груп (ще не заповнені студентами) доведеться створювати через меню *Керування сайтом* –

Користувачі – Облікові записи – Контингенти – Завантаження контингентів, вказавши в полі *idnumber* те саме значення, що й у полі *name.*

Отже, щоб створити файл csv для зарахування (або відрахування) контингентів студентів у службові курси, можна спочатку відкрити в LibreOffice файл, що містить тільки ці два рядки:

```
operation,method,shortname,metacohort,disabled,group,role
add (або del),meta,СК,Група,0,,student
```

Потім на місце виділених сірим фоном полів вставити посилання на клітинки у рядку 5 (першому рядку даних у таблиці на рис. 7) із зазначеними назвами стовпців. Решту рядків файлу csv створюємо з решти даних цієї таблиці за допомогою автозаповнення.

Файли з командами для Moosh чи csv файли для Moodle можна формувати також і за допомогою програм на VBA або LibreOffice Basic. Власне, щоб створити таку програму, навіть необов'язково вміти програмувати. Можна при першому виконанні описаних вище робіт просто увімкнути запис макроса. У результаті всі дії, які виконуватимуться в Excel або LibreOffice Calc під час цього запису, будуть збережені у вигляді програми на VBA чи LibreOffice Basic. Тоді наступного разу для виконання тих самих дій буде достатньо лише запустити макрос на виконання.

Можна частину програми створити за допомогою макроса, а іншу частину написати самому. Наприклад, зазначену вище операцію збереження вмісту стовпця *Команда* (уже в потрібному для виконання форматі, що не потребує конвертування за допомогою *dos2unix*) у текстовому файлі, ім'я якого щоразу обирає користувач, на VBA можна запрограмувати так:

```
Sub FileSave()
t = "#!/bin/bash" + Chr(10)
i = 5
Do While Cells(i, 5) <> ""
 t = t + Cells(i, 5) + Chr(10)
 i = i + 1
Loop
fileSaveName = Application.GetSaveAsFilename( _
 fileFilter:="Text Files (*.sh), *.sh")
With CreateObject("ADODB.Stream")
 .Type = 2: .Charset = "utf-8": .Open
 .WriteText t
 Set binaryStream = CreateObject("ADODB.Stream")
 binaryStream.Type = 1: binaryStream.Mode = 3
 binaryStream.Open:
 .Position = 3: .CopyTo binaryStream 'Skip BOM bytes
 .Flush: .Close
 binaryStream.SaveToFile fileSaveName, 2
 binaryStream.Close
End With
End Sub
```

На завершення відзначимо, що видалення з курсу минулорічних студентів здійснюється не одночасно з додаванням нових, а через певний час, який деканат дає їм на ліквідацію заборгованостей. Упродовж цього часу при виборі відповідної групи викладачам у їхніх курсах будуть доступні як нові, так і минулорічні студенти.

Плагіни для автоматичного створення і наповнення контингентів і груп

В обліковому записі кожного користувача повинні міститися дані про те, де саме він навчається чи працює. За назвою для цього найбільш доречне поле *Підрозділ (department)*, у якому для студентів можна вказати академічну групу, а для викладачів – кафедру.

Auto-cohort plugin [4] автоматично додаватиме користувачів у контингент, назва якого вказана в полі *department* їх облікового запису, якщо в налаштуваннях цього плагіна в параметрі *Main template* прописати *{{ department }}*. Додавання відбуватиметься при кожному оновленні облікового запису окремого користувача або для всіх користувачів – при виконанні команд синхронізації крон: *cli/sync_user.php* або *cli_sync_users.php*.

У такий же спосіб можна налаштувати плагін Auto Group [4]. Він автоматично записуватиме користувача у вказану в полі *department* групу, незалежно від того, яким способом цей користувач був зарахований у курс. Більш того, плагін створить в курсі нову групу, якщо в ньому з'явиться хоча б один представник цієї групи, і видалить групу з курсу, якщо в ньому більше не залишається жодного її представника. Все це стосується груп, автоматично створених самим плагіном, і не стосується інших груп, які викладач може створювати на свій розсуд.

Зауважимо, що раніше переписати студентів з одного курсу на інший разом з їхніми групами було доволі складною задачею. Наприклад, у роботі [10] для цього навіть доводилось вдаватися до зарахування студентів на курс двома способами одночасно. Зараз реалізацію основних потрібних викладачу функцій роботи з групами перебирає на себе цей плагін. Тому при зарахуванні на курс студентів за допомогою контингентів чи мета-курсів їх параметри щодо створення груп можна навіть не заповнювати.

При встановленні плагіна треба залишити значення за замовчуванням *department* у параметрі *Group by*. Також вказати, на які ролі поширюється робота плагіна.

Однак, незважаючи на налаштування за замовчуванням, перед використанням плагін треба активувати в кожному курсі, вибравши *Користувачі - Групи - Auto Groups*, та задати там такі параметри: *Create new group set = Profile field, Group by = department, Eligible Roles = Студент.* У результаті в меню *Учасники – Групи – Auto Group* буде створено приблизно такий рядок (рис. 8):

Рис. 8. Правило автоматичного створення груп на курсі

Отже, плагін дає змогу створювати не одне, а багато правил автоматичного створення і наповнення груп, причому ці налаштування робляться для кожного курсу окремо і зберігаються в таблицях, доданих цим плагіном у базу даних Moodle. Тому виконати це налаштування в автоматичному режимі для всіх курсів сайту поки що немає можливості.

Щоб не робити це «вручну», доведеться написати код ще однієї команди Moosh і виконати з нею аналогічну послідовність дій: створюємо в Configurable Reports і експортуємо в Excel звіт, що містить ID потрібних курсів, а потім формуємо там команди Moosh, які здійснять налаштування плагіна Auto Group у всіх цих курсах. Хоча скрізь, де це можливо і зручно, можна обійтися і без Excel, якщо сформувати команди Moosh у самому звіті, а потім експортувати їх безпосередньо в текстовий файл.

3. ВИСНОВКИ ТА ПЕРСПЕКТИВИ ПОДАЛЬШИХ ДОСЛІДЖЕНЬ

Отже, у статті запропоновано спосіб заміни традиційного ручного адміністрування сайту Moodle автоматизованим, що здійснюється не людиною-адміністратором, а програмою. Водночас сам текст програми теж формується автоматично, шляхом експорту даних сайту у вигляді звітів Configurable Reports та їх перетворення на командні файли Moosh або файли даних csv, за допомогою яких на сайт вносяться відповідні зміни: автоматично створюються і наповнюються курси і категорії курсів, створюються зв'язки між курсами у вигляді субкурсів та мета-курсів, зараховуються і відраховуються студенти тощо. Це не тільки дає велику економію часу, а й зменшує кількість, притаманних традиційному адмініструванню помилок, тобто покращує якість адміністрування сайту.

Порівнюючи розглянуті в статті способи формування командних файлів Moosh і даних файлів csv, можна зробити висновок, що найбільш універсальним і наочним серед них є злиття документів у Word. У його основному документі добре видно структуру даних файлу csv чи команд Moosh, туди легко вносити виправлення, а під час злиття є можливість здійснювати сортування і фільтрацію в джерелі даних.

У простіших випадках можна обійтись і без злиття документів, формуючи команди чи дані прямо в Excel. А у ще простіших можна обійтися і без Excel, формуючи їх безпосередньо у звіті Configurable Reports.

Якщо порівнювати використання команд Moosh і файлів csv, то можна зробити висновок, що кожний із цих засобів має свої переваги і недоліки. Будь-які команди Moosh вводяться через ту саму консоль, тоді як кожний тип файлів csv завантажується через свій власний інтерфейс Moodle, що не так зручно. На прикладах, розглянутих у цьому дослідженні, файл csv часто був не просто більш гнучким інструментом, а й більш функціональним. Однак, якщо порівняти загальну кількість функцій, які можна реалізувати кожним із них, то перевага буде таки за Moosh.

У статті були розглянуті всі етапи створення і наповнення структури курсів, показаної на рис. 1. Через великий обсяг робіт (необхідність створення тисяч субкурсів і мета-курсів) її реалізація традиційними засобами не завжди під силу адміністратору. Але з використанням запропонованих рішень він спроможний це зробити. Однак, звісно, область їх використання не обмежується лише розглянутою вище задачею. Вони можуть бути успішно використані для вирішення багатьох інших задач, які потребують великих обсягів робіт з адміністрування системи Moodle, і, на нашу думку, мають бути в арсеналі кожного адміністратора.

Перспективи подальших досліджень автор вбачає у вдосконаленні розглянутих тут програмних засобів, перш за все в написанні коду деяких поки що відсутніх команд для утиліти Moosh, а також у створенні засобів, які дозволяють формувати командні файли Moosh або файли csv для Moodle швидше і комфортніше. Це може бути реалізовано шляхом створення інтегрованого програмного середовища, яке міститиме застосовані вище, а можливо й інші програмні засоби.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] I. Subashkevych, V. Korniat V. Loboda, I. Sihetii, M. Opachko, N. Sirant. Using Moodle in an Information Educational Environment of HEIs under Distance Learning. *BRAIN. Broad Research in Artificial Intelligence and Neuroscience*. 2021, V. 12, Issue 4, pages: 346-357| doi: https://doi.org/10.18662/brain/12.4/254

[2] S. Gamage, J.R. Ayres, and M.B. Behrend, A systematic review on trends in using Moodle for teaching and learning. *IJ STEM* Ed 9, 9 (2022). https://doi.org/10.1186/s40594-021-00323-x

[3] LMS global stats – you're probably half right. [Електронний ресурс]. Режим доступу: https://www.elearningworld.org/lms-global-stats-youre-probably-half-right/

[4] Plugins. [Електронний ресурс]. Режим доступу: https://moodle.org/plugins/

[5] G. García-Murillo, P. Novoa-Hernández and R. S. Rodríguez, Technological Satisfaction About Moodle in Higher Education—A Meta-Analysis, *in IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 15, no. 4, pp. 281-290, Nov. 2020, doi: 10.1109/RITA.2020.3033201.

[6] Tomasz Muras. *Moodle 3.1 LTS modules development*. [Електронний ресурс]. Режим доступу: https://leanpub.com/moodle

[7] R. Beuran, D.Tang, Z. Tan, S. Hasegawa, Y. Tan, and Y. Shinoda. Supporting cybersecurity education and training via LMS integration: CyLMS. *Education and Information Technologies*, 2019, 24(6), 3619-3643.

[8] A. Büchner. *Moodle 3 administration*. Packt Publishing Ltd, 2016.

[9] Rüdian, Sylvio, and Niels Pinkwart. "Generating adaptive and personalized language learning online courses in Moodle with individual learning paths using templates." *2021 International Conference on Advanced Learning Technologies (ICALT)*. IEEE, 2021.

[10] О.А. Щербина. Реалізація функцій електронного деканату засобами платформи Moodle. *Інформаційні технології і засоби навчання*, 2015, Том 50, №6. С. 139-151.

[11] Moosh. [Електронний ресурс]. Режим доступу: https://moosh-online.com/

Матеріал надійшов до редакції 22.10.2022 р.

AUTOMATION OF THE CREATION, FILLING AND ADMINISTRATION OF COURSE CATEGORIES ON THE MOODLE WEBSITE

Oleksandr A. Shcherbyna
PhD of Technical Sciences, Associate Professor,
Associate Professor at the Department of System Programming and Specialized Computer Systems,
National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Kyiv, Ukraine
ORCID ID 0000-0003-0224-1441
oscherbyna@i.ua

Abstract. The choice of course categories structure of the Moodle site largely determines its effectiveness and ease of use. There are two main approaches to choosing the course category structure on the university website. The first one structures them from the point of view of teachers: the categories of courses of departments are placed in the categories of courses of the faculty. The second is from the point of view of students: the categories of courses of educational programs are placed in the category of courses of faculties. The paper considers a structure that best combines these two approaches. It uses the first approach as a basic one, and the lists of courses for educational programs are formed in the form of hyperlinks, which are implemented using the elements of the Subcourse course placed in service courses specially created for them. This makes it possible to import the resulting grades from all disciplines of a certain educational program and year of study into the gradebook of the service course and calculate the current rating of each student on their basis, which the Moodle system updates every minute, and to export lists of students in the opposite direction. Then enrolling or withdrawing a student in a service course causes the same actions in all courses of this educational program. Such site structure has significant advantages, but its creation requires a large amount of work by the administrator, so it needs automation. For this, the Moosh utility is used, designed for the administration of the Moodle site through the command line interface, which significantly speeds up the administration process compared to performing the same actions through the native Moodle interface. An even greater speedup can be achieved by running files containing many such commands for execution

in Moosh. But the main content of this work is the automation of the creation of the command files themselves. This further speeds up administration processes and improves their quality by reducing the number of errors that occur during manual typing of commands or site administration with Moodle's own tools. The work describes the process of creating command files that ensure the automatic execution of all work related to the creation of the above-mentioned site structure: creation of course categories, creation of service courses and their filling, as well as further site administration related to student enrollment for courses of the new academic year, their deduction from last year's courses, etc. New plugins that support the functioning of such a site structure are also being considered.

Keywords: Moodle; automation of administration; Moosh.

REFERENCES (TRANSLATED AND TRANSLITERATED)

[1] I. Subashkevych, V. Korniat V. Loboda, I. Sihetii, M. Opachko, N. Sirant. Using Moodle in an Information Educational Environment of HEIs under Distance Learning. BRAIN. *Broad Research in Artificial Intelligence and Neuroscience*. 2021, V. 12, Issue 4, pp. 346-357| https://doi.org/10.18662/brain/12.4/254 (in English)
[2] S. Gamage, J.R. Ayres, and M.B. Behrend, A systematic review on trends in using Moodle for teaching and learning. *IJ STEM Ed* 9, 9 (2022). https://doi.org/10.1186/s40594-021-00323-x (in English)
[3] LMS global stats – you're probably half right. [Online]. Access: https://www.elearningworld.org/lms-global-stats-youre-probably-half-right/ (in English)
[4] Plugins. [Online]. Access: https://moodle.org/plugins/ (in English)
[5] G. García-Murillo, P. Novoa-Hernández and R. S. Rodríguez, Technological Satisfaction About Moodle in Higher Education – A Meta-Analysis, in IEEE *Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 15, no. 4, pp. 281-290, Nov. 2020, doi: 10.1109/RITA.2020.3033201. (in English)
[6] Tomasz Muras. Moodle 3.1 LTS modules development. [Online]. Access: https://leanpub.com/moodle (in English)
[7] R. Beuran, D.Tang, Z. Tan, S. Hasegawa, Y. Tan, and Y. Shinoda. Supporting cybersecurity education and training via LMS integration: CyLMS. *Education and Information Technologies*, 2019, 24(6), 3619-3643. (in English)
[8] A. Büchner. *Moodle 3 administration*. Packt Publishing Ltd, 2016. (in English)
[9] Rüdian, Sylvio, and Niels Pinkwart. "Generating adaptive and personalized language learning online courses in Moodle with individual learning paths using templates." *2021 International Conference on Advanced Learning Technologies* (ICALT). IEEE, 2021. (in English)
[10] O.A. Shcherbyna. Implementation of functions of the electronic dean's office using the Moodle platform. *Information Technologies and Learning tools*, 2015, Volume 50, no. 6. pp. 139-151. (in Ukrainian)
[11] Moosh. [Online]. Access: https://moosh-online.com/ (in English)