МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО

МЕТОДИЧНІ ВКАЗІВКИ
ЩОДО ВИКОНАННЯ САМОСТІЙНОЇ РОБОТИ
З НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
«ІНЖЕНЕРІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»
ДЛЯ СТУДЕНТІВ ДЕННОЇ ФОРМИ НАВЧАННЯ
ЗІ СПЕЦІАЛЬНОСТІ 123 – «КОМП'ЮТЕРНА ІНЖЕНЕРІЯ»
(ЧАСТИНА ІІ)»

КРЕМЕНЧУК 2020

Методичні вказівки щодо виконання самостійної роботи з навчальної дисципліни «Інженерія програмного забезпечення» (частина ІІ) для студентів денної форми навчання зі спеціальності 123 – «Комп'ютерна інженерія»

Укладач к. т. н., доц. О. Г. Славко

Рецензент к. т. н., доц. В. М. Сидоренко

Кафедра комп'ютерних та інформаційних систем

Затверджено методичною радою КрНУ імені Михайла Остроградського
Протокол № ___від _____ 2020 р.

Голова методичної ради _____ проф. В. В. Костін

ЗМІСТ

ВСТУП

Навчальна дисципліна «Інженерія програмного забезпечення» є логічним продовженням курсу «Комп'ютерні системи», «Програмування», «Комп'ютерні мережі», «Мережні інформаційні технології», «Організація баз даних». Предметом вивчення навчальної дисципліни є теорія і практика розробки прикладних мережних додатків на базі сучасних технологій розробки програмного забезпечення в комп'ютерних мережах.

У межах навчальної дисципліни розглядаються питання прикладного програмування в комп'ютерних мережах, зокрема web-додатків. Студенти набувають практичних навичок і здатності застосовувати мережні технології, методи проектування та інструменти для розроблення мережних програмних продуктів на сучасних мережних платформах, а саме: теоретичні та практичні аспекти технології розробки сучасних веб-додатків; основні типи веб-додатків, методології їх проектування та розробки; інформаційні системи з використанням веб-додатків, інструментальні засоби, що підтримують розробку програмного забезпечення професійно орієнтованих веб-додатків, технічні засоби інформаційних систем на основі веб-додатків у предметній галузі.

Мета та завдання навчальної дисципліни: набуття студентами загальних теоретичних і практичних знань у галузі створення прикладних мережних додатків, які базуються на сучасних мережних технологіях і платформах, а також організація функціонування та супровід мережних додатків, побудованих за клієнт-серверною технологією.

Місце навчальної дисципліни у навчальному процесі: курс «Інженерія програмного забезпечення» тісно пов'язаний з такими дисциплінами, як «Комп'ютерні системи», «Програмування», «Комп'ютерні мережі», «Мережні інформаційні технології», «Організація баз даних» та з іншими спеціальними дисциплінами навчального плану.

У результаті вивчення навчальної дисципліни студент повинен:

знати:

– основні сучасні технології та методологію розробки мережних додатків;

– методи організації процесу розробки web-додатків;

– стандарти, що лежать в основі web-технологій;

– підходи до побудови мережних додатків з урахуванням сучасних вимог інформаційного простору;

– принципи взаємодії web-додатка з іншими компонентами інформаційної системи;

– інструментальні засоби створення серверної та клієнтської частин мережних додатків;

уміти:

– організовувати процес проектування та розробки web-додатка;

– розробляти серверну частину мережних додатків для різних мережних платформ, використовуючи різні технології;

– розробляти клієнтську частину мережних додатків для різних мережних платформ, використовуючи різні технології;

– обґрунтовано вибирати програмну платформу й інструментарій для розробки web-додатків;

– застосовувати стандарти web-технологій;

– здійснювати супровід мережних додатків.

1 ТЕМИ ТА ПОГОДИННИЙ РОЗКЛАД ЛЕКЦІЙ І САМОСТІЙНОЇ РОБОТИ

Назви змістових модулів і тем	Кількість годин				
	денна форма				
	усього	у тому числі			
		л	п	лаб	с.р.
1	2	3	4	5	6
Модуль 2					
Змістовий модуль 3 Архітектурне проектування програмного забезпечення					
Тема 1 Архітектурне проектування	10	2		2	6
Тема 2 Архітектурні шаблони	12	4		2	6
Тема 3 Архітектура розподілених систем	10	2		2	6
Тема 4 Проектування інтерфейсу користувача	10	2		2	6
Усього за змістовим модулем 3	42	10		8	24
Змістовий модуль 4 Верифікація програм. Методи перевірки та тестування програм і систем					
Тема 5 Тестування програм та систем	14	2		4	8
Тема 6 Функціональне тестування програмного забезпечення	12	2		4	6
Тема 7 Структурне тестування програмного забезпечення	14	4		4	6
Тема 8 Метрики об’єктно орієнтованих програмних систем	14	4		4	6
Усього за змістовим модулем 4	54	12		16	26
ІНДЗ (КР, РГ, к/р)	30	-	-	-	30
Семестровий контроль (іспит)	4	-	-	-	4
Усього за модулем 2	130	22		24	84

2 ЗАВДАННЯ ДЛЯ САМОСТІЙНОЇ РОБОТИ

Тема. Динамічний HTML. Об’єкти JavaScript

Мета: набуття практичних навичок з програмування мовою javascript, отримання уявлення про практичноме використанні об’єктної моделі веб-документа (DOM) і використання веб-форм.

Короткі теоретичні відомості

CAPTCHA (від англ. «Completely Automated Public Turing test to tell Computers and Humans Apart») – це повністю автоматизований публічний тест Тьюринга для розрізнення комп’ютерів і людей) комп’ютерний тест, який використовують для того, щоб визначити, ким є користувач системи: людиною або комп’ютером. Основна ідея тесту – запропонувати користувачу таке завдання, яке може розв’язати людина, але яке незрівнянно складно надати для розв’язання комп’ютером. Переважно це завдання на розпізнавання символів. CAPTCHA найчастіше використовується за необхідності запобігти використанню інтернет-сервісів ботами, зокрема для запобігання автоматичного відправлення повідомлень, реєстрації, скачування файлів, масових розсилань тощо.

Елементи керування – це інтерактивні об’єкти, що дозволяють отримати дані від користувача. Їх призначення і зовнішній вигляд ідентичні елементам користувальницького інтерфейсу сучасних операційних систем з графічним інтерфейсом (кнопки, поля введення, чекбокси та ін.).

Елемент input

Тег <input> представляє різні елементи залежно від значення атрибута type.

Атрибути елемента input

type визначає тип поля введення. За замовчуванням дорівнює text.

name – ім’я поля введення. Використовується як ідентифікатор змінної для передавання даних на сервер і для програмного звернення до елемента з

скрипта javascript.

id – ідентифікатор елемента. Повинен бути унікальним у межах веб-документа.

checked означає, що checkbox або radio буде вибраний.

maxlength визначає кількість символів, яку користувачі можуть увести в поле введення. У разі перевищення кількості допустимих символів броузер реагує на спробу введення нового символу звуковим сигналом і перешкоджає його введенню.

size визначає візуальний розмір поля введення на екрані в символах.

src – URL, що вказує на картинку (використовується спільно зі значенням type = "image").

value – значення за замовчуванням або встановлене значення.

Елемент textarea

Тег <textarea> використовується для того, щоб дозволити користувачеві вводити більше, ніж один рядок інформації (багаторядковий текст). Під час передачі значення з textarea зберігаються всі символи форматування (табуляція, переклад рядка, повернення каретки).

Атрибути, що використовуються з тегом <textarea>, задають його розміри (у символах і рядках):

rows – висота поля введення в символах;

cols – ширина поля введення в рядках.

Приклад використання тега <textarea>:

<Textarea rows = 10 cols = 50> Київ, Бориспільське шосе, буд. 9Б, офіс 448 </ textarea>

Елемент select

Елемент select відображає на сторінці список вибору, який може бути поданий такими способами:

select – список, що випадає;

select single – розгорнутий список;

select multiple – список з множинним вибором.

Приклади опису елемента select:

<Select name = "group">

<Option> понеділок, середа, п’ятниця </ option>

<Option> вівторок, четвер, субота </ option>

<Option> Субота </ option>

</ Select>

<Select single name = "group" size = "3">

<Option> зима </ option>

<Option> весна </ option>

<Option> літо </ option>

<Option> осінь </ option>

</ Select>

Об’єкт document

Об’єкт document – це абстрактна структура даних, що є повним описом веб-сторінки. Набір властивостей і методів цього об’єкта дозволяє керувати як поведінкою веб-сторінки цілком, так і окремих її об’єктів (елементів керування, посилань, текстових блоків, зображень і т. д.). Доступ до властивостей і методів реалізований за допомогою стандартних програмних інтерфейсів.

Властивості об’єкта Document

Почнемо з властивостей, загальних для всіх браузерів. Більшість їх доступні як для читання, так і для зміни. Усі значення властивостей – строкові.

title – текст заголовка документа (вміст елемента title);

fgColor і bgColor – колір тексту і колір фону документа;

linkColor, vLinkColor, aLinkColor – кольори невідвіданих, відвіданих і активних гіперпосилань;

lastModified (тільки для читання) – дата зміни документа;

referrer (тільки для читання) – адреса джерела переходу;

URL, location – власна адреса документа.

Більш цікаві та корисні для розробника властивості-масиви об’єкта Document. Усі вони, природно, мають властивість length (кількість елементів у масиві). Більшість властивостей, специфічних для об’єктів, що зберігаються в цих масивах, асоціюються з атрибутами відповідних елементів HTML (список неповний):

об’єкт Anchor (якір) має єдину властивість name;

об’єкт Link (посилання) має властивості href, target;

об’єкт Image (зображення) має властивості src, width, height.

До об’єктів документа, що зберігається в масивах images, controls і інших, а також до елементів форм можна звертатися по імені (властивість name) чи кодом (властивість id). Нехай, наприклад, у документі є опис <img src = "images / cat.jpg" id = "cat_id" name = "cat_name"> і він є *n*-м зображенням, яке є в документі. До цього елемента img можна звернутися такими способами:

1) як до елемента масиву images за індексом (індексація починається з 0): window.document.images [*n* – 1].

2) як до елемента хеш-массива images по ключу (значення name як ключ масиву): window.document.images ["cat_name"].

3) використовуючи значення атрибута name як властивість об’єкта: window.document.cat_name;

4) використовуючи значення атрибуту id і властивість getElementById: window.document.getElementById ("cat_id").

Методи об’єкта Document

open() відкриває новий документ, при цьому всі його елементи вилучаються.

close() закриває раніше відкритий документ.

write() записує в документ заданий як аргумент рядок.

writeln() аналогічний попередньому, але виведений в документ рядок закінчується символом переведення рядка.

Методи write() і writeln() дуже корисні та часто використовуються для динамічного формування вмісту документа. Ось так, наприклад, можна внести у документ дату його останньої зміни:

<Script> document.write (document.lastModified); </ script>

Події

Для всіх елементів документа є можливість відстежувати різні події (завантаження, переміщення миші, натискання миші та ін.) і викликати функції обробки таких подій.

Порядок виконання самостійної роботи

1. Ознайомитися з теоретичними відомостями.

2. Написати скрипт, який перевіряє код захисту від автоматичного постингу і вирізає посилання з форми введення коментаря (на сторінці відгуків і коментарів)

Пояснення. Під час розв'язання завдання потрібно написати клієнтську програму на javascript, яка генерує арифметичний приклад, відповідь на який повинен дати користувач. Інший варіант – генерація довільного рядка, який повинен відтворити користувач. Після того, як користувач увів відповідь, програма має перевірити його правильність.

У лістингах 1–6 наведені приклади простих скриптів, що ілюструють базові можливості javascript під час роботи з об'єктами веб-документа. Для розв'язання завдань використовуйте пропоновані приклади як зразки. Ураховуйте, що різні браузери можуть по-різному виконувати код javascript (або навіть не виконувати його зовсім).

Лістинг 1. Обмеження кількості символів

```
<html>
<head>
<title>Ограничение количества вводимых символов</title>
<script type="text/javascript">
var maxLen = 25;
```

```
function checkMaxinput(form) {
        if (form.message.value.length > maxLen)
                form.message.value = form.message.value.substring(0, maxLen);
        else
                form.remLen.value = maxLen –
                form.message.value.length;
                }
</script>
</head>
<body>
<form name=myform action="somehandler.cgi">
<h1>Ограничение количества вводимых символов<h1>
<textarea name=message cols=28 rows=4 onKeyDown="checkMaxinput(this.form)"
        onKeyUp="checkMaxinput(this.form)"></textarea>
<p>Осталось <input readonly type=text name=remLen size=3 value="25">
символов</p>
</form>
</body>
</html>
```

Лістинг 2. Перевірка вводу

```
<html>
<head>
<title>Проверка ввода
</title>
<SCRIPT type="text/javascript">
function checkIt(){
        var t0=document.getElementById('first').value;
        var t1=document.getElementById('second').value;
        if (t0 == "" || t0 == "Имя") {
                alert("Вы не указали свое имя!"); return false;
                }
        if (t1 == "") {
                alert("Вы не ввели необходимую информацию!");
                return false;
                }
return true;
</SCRIPT>
</head>
<body>
<form method='get' action='somescript.php'>
<input id="first" type="text" size=60px value='Имя'>
<br>
<textarea id="second" rows=4 cols=60></textarea>
<br>
```

```
<input type='submit' onClick="if (!checkIt()){return false;}" value="OK">
</form>
</body>
</html>
```

Лістинг 3. Керування окнами

```
<html>
<head>
<title>Открытие/закрытие нового окна</title>
</head>
<body>
<p><a name="demoOpen" onClick="mywindow =
window.open('window.htm','mywin','height=120, width=300, left=100,
top=30');">Открыть</a>
<a name="demoClose" onClick="mywindow.window.close();">Закрыть</a>
</body>
</html>
```

Лістинг 4. Зміна оформлення

```
<html>
<HEAD>
<TITLE>Изменение цвета объекта по щелчку мыши</TITLE>
</head>
<BODY>
<p onClick="fgColor='#3CB094';bgColor='#FFFF00';">CLICK 4 REDRAW</p>
</BODY>
</HTML>
```

Лістинг 5. Поточний час

```
<html>
<HEAD>
<TITLE>Часы, отображающие текущее время</TITLE>
<script type="text/javascript">
function fulltime() {
        var time=new Date();
        document.clock.full.value=time.toLocaleString(); // 1-ый вариант
        document.getElementById("jsclock").innerHTML=time.toLocaleString(); //
2-ой вариант
        setTimeout('fulltime()',500) }
</script>
</head>
<body>
<form name=clock>
<input type=text size=20 name=full><!-- 1-ый вариант -->
<span id="jsclock"></span><!-- 2-ой вариант -->
```

```
</form>
<script type="text/javascript"> fulltime(); </script>
</BODY> </HTML>
```

Листинг 6. Визначення браузера (використаний об’єкт navigator)

```
<HTML>
<HEAD>
<TITLE>Сведения о браузере</TITLE>
</HEAD>
<BODY>
<h1>Для навигации в Web вы используете:</h1>
<ul>
<SCRIPT type="text/javascript">
        document.write("<li>Имя программы:<b>"+navigator.appName+"</b>");
        document.write("<li>Версия:<b>"+navigator.appVersion+"</b>");
        document.write("<li>Пользовательский
агент:<b>"+navigator.userAgent+"</b>");
        document.write("<li>Платформа: <b>"+navigator.platform+"</b>");
</SCRIPT>
</ul> </BODY> </HTML>
```

Зміст звіту

1. Титульна сторінка.
2. Опис виконання роботи відповідно до пп. 1–2.

Питання для самоперевірки

1. Які існують сучасні методології розробки ПЗ?
2. Що передбачає управління ризиками в інжиніринговій компанії?
3. Які етапи має розробка програми управління ризиками в ІТ?
4. Перелічити методи тестування під час набору персоналу.
5. Які існують графічні середовища зображення проектної діяльності?
6. Що передбачає управління вартістю проекту?
7. Коротко описати основи розробки бізнес-планів.

Література: [1, с. 54–75; 2, с. 112–125; 5, с. 1–34].

3 КРИТЕРІЇ ОЦІНЮВАННЯ ЯКОСТІ ВИКОНАННЯ САМОСТІЙНОЇ РОБОТИ

У 8-му семестрі студенти виконують самостійну роботу, що складається з 1 завдання. Загальна кількість балів, яку отримують студенти за виконання самостійної роботи, становить максимально 5 балів.

Шкала оцінювання: національна та ECTS

Сума балів за всі види навчальної діяльності	Оцінка ECTS	Оцінка за національною шкалою	
		Для іспиту, курсового проекту (роботи), практики	Для заліку
90–100	A	Відмінно	Зараховано
82–89	B	Добре	
74–81	C		
64–73	D	Задовільно	
60–63	E		
35–59	FX	Незадовільно з можливістю повторного складання	Не зараховано з можливістю повторного складання
0–34	F	Незадовільно з обов’язковим повторним вивченням навчальної дисципліни	Не зараховано з обов’язковим повторним вивченням навчальної дисципліни

СПИСОК ЛІТЕРАТУРИ

1. Браун Д. М. Разработка веб-сайта. Взаимодействие с заказчиком, дизайнером и программистом. СПб.: Питер, 2009. 336 с.
2. Веллинг Л., Томсон Л. Разработка Web-приложений с помощью PHP и MySQL: пособие. М.: Питер, 2008. 800 с.
3. Форристал Д. Защита от хакеров Web-приложений: учеб. пособие. М.: ДМК Пресс, 2008. 496 с.
4. Зандстра М. PHP: объекты, шаблоны и методики программирования. 3-е изд. М.: Вильямс, 2010. 560 с.
5. Ленгсторф Д. PHP и jQuery для профессионалов. М.: Вильямс, 2010. 352 с.
6. Дари К., Баланеску Э. PHP и MySQL: создание интернет-магазина. М.: Вильямс, 2010. 640 с.
7. Хокинс С. Администрирование Web-сервера Apache и руководство по электронной коммерции. М.: Вильямс, 2001. 336 с.
8. Боуэн Р. Ридруэйо Д. Л., Лиска А. Apache. Настольная книга администратора: пер. с англ. К.: ДиаСофт, 2002. 384 с.
9. Бакор А. Apache Tomcat для профессионалов. М.: Кудиц-Образ, 2005. 544 с.
10. Гонсалвес Э. Изучаем Java EE 7. СПб.: Питер, 2014. 640 с.
11. Монахов В. В. Язык программирования Java и среда NetBeans. 3-е изд. СПб.: БХВ-Петербург, 2011. 704 с.
12. Кузнецов М. В., Симдянов И. В. PHP. Практика создания Web-сайтов. 2-е изд., перераб. и доп. СПб.: БХВ-Петербург, 2009. 1264 с.

Методичні вказівки щодо виконання самостійної роботи з навчальної дисципліни «Інженерія програмного забезпечення» (частина ІІ) для студентів денної форми навчання зі спеціальності 123 – «Комп'ютерна інженерія»

Укладач к. т. н., доц. О. Г. Славко

Відповідальний за випуск зав. кафедри КІС М. І. Гученко

Підп. до др. _____. Формат 60×84 1/16. Папір тип. Друк ризографія.
Ум. друк. арк. ____. Наклад _____ прим. Зам. №_____. Безкоштовно.

Редакційно-видавничий відділ
Кременчуцького національного університету
імені Михайла Остроградського
вул. Першотравнева, 20, м. Кременчук, 39600