

**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА**

ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ

Кафедра теорії та технології програмування

«ЗАТВЕРДЖУЮ»

Заступник декана
з навчальної роботи

_____ Кашпур О.Ф.

«_____» _____ 2018 року

**РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
"ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ "**

для студентів

галузь знань **12 «Інформаційні технології»**
(шифр і назва)

спеціальність **122 «Комп'ютерні науки»**
(шифр і назва спеціальності)

освітній рівень **бакалавр**
(молодший бакалавр, бакалавр, магістр)

освітня програма **«Інформатика»**
(назва освітньої програми)

вид дисципліни **обов'язкова**

Форма навчання	денна
Навчальний рік	2018/2019
Семестр	7
Кількість кредитів ECTS	4
Мова викладання, навчання та оцінювання	українська
Форма заключного контролю	екзамен

Викладачі: **к.ф.-м.н., доц. Кузенко В.Ф.**

Пролонговано: на 20__/20__ н.р. _____ (_____) «__» 20__ р.
(підпис, ПІБ, дата)

на 20__/20__ н.р. _____ (_____) «__» 20__ р.
(підпис, ПІБ, дата)

КИЇВ – 2018

Розробник: Кузенко Володимир Федорович, к.ф.-м.н., доцент кафедри «Теорії та технології програмування»

ЗАТВЕРДЖЕНО

В. о. зав. кафедри «Теорії та технології програмування»

_____ (Панченко Т.В.)
(підпис) (прізвище та ініціали)

Протокол № ____ від «____» _____ 20__ р.

Схвалено науково-методичною комісією факультету комп'ютерних наук та кібернетики

Протокол від «____» _____ 20__ року № ____

Голова науково-методичної комісії _____ (Хусаїнов Д.Я.)
(підпис) (прізвище та ініціали)

«____» _____ 20__ року

Затверджено вченою радою факультету комп'ютерних наук та кібернетики

Протокол від «____» _____ 20__ року № ____

Голова Вченої ради _____ Анісімов А.В.
(підпис) (прізвище та ініціали)

ВСТУП

1. Мета дисципліни – засвоєння базових знань та оволодіння навичками щодо використання інформаційних технологій при розробці програмних систем (ПС), зокрема, оволодіння навичками моделювання і проектування розподілених програмних систем, використання сервісно-орієнтованої архітектури, *REST*-архітектури, оволодіння окремими навичками *Web*-програмування та програмування під мобільні платформи, використання хмарних обчислень.

2. Попередні вимоги до опанування або вибору навчальної дисципліни (за наявності):

1. Знати:

- основні поняття та шаблони об'єктно-орієнтованого проектування програмного забезпечення;
- основні етапи життєвого циклу ПС;

2. Вміти:

застосовувати на практиці інструментальні засоби проектування та розробки програмного забезпечення.

3. Володіти навичками:

програмування із використанням об'єктно-орієнтованих мов.

3. Анотація навчальної дисципліни (до 700 символів):

Навчальна дисципліна “Інформаційні технології” є складовою програми підготовки фахівців за першим (бакалаврським) рівнем вищої освіти *галузі знань* 12 „Інформаційні технології” зі *спеціальності* 122 „Комп'ютерні науки”, *освітньо-професійної програми* – „Інформатика”.

Дана дисципліна є обов'язковою навчальною дисципліною за *програмою “Інформатика”*.

Викладається у 7 семестрі 4 курсу в **обсязі – 120 год.**

(**4 кредити ECTS**) зокрема: *лекції – 30 год., лабораторні – 24 год., консультації – 2 год., самостійна робота – 64 год.* У курсі передбачено **2 частини** та **2 контрольні роботи**. Завершується дисципліна – **екзаменом у 7 семестрі**.

В результаті вивчення навчальної дисципліни студент повинен:

знати основні поняття стосовно моделювання, проектування та розробки програмних систем із використанням віддаленої взаємодії на основі клієнт-серверної архітектури, зокрема, поняття сервісно-орієнтованої архітектури, *REST*-архітектури, *Web*-програмування, хмарних обчислень;

вміти застосовувати на практиці програмні засоби, фреймворки та інструментальні середовища при розробці програмних систем, орієнтованих на віддалену взаємодію, зокрема, на використання технологій, спряжених із *Web*-, мобільними платформами, хмарними обчисленнями.

Для допуску до дисципліни „Інформаційні технології” освітньо-професійної програми «Інформатика» студент повинен опанувати компетентності та результати навчання, які надають дисципліни “Об'єктно-орієнтоване програмування”, “Інструментальні середовища та технології програмування”, “Системне програмування” програми «Інформатика». Дисципліна „Інформаційні технології” є базовою для засвоєння дисциплін спеціалізації, дисциплін

вільного вибору студента програмістського спрямування програми «Інформатика» та виробничої практики "Інформаційні системи та технології".

4. Завдання (навчальні цілі):

набуття знань, умінь та навичок (компетентностей) на рівні новітніх досягнень у програмуванні, відповідно до кваліфікації фахівець з інформаційних технологій. Зокрема, розвивати:

- здатність використовувати інтелектуальні інформаційні технології;
- здатність використовувати інструментальні засоби розробки клієнт-серверних застосувань;
- здатність забезпечити виконання розподілених обчислень, використання мов програмування при розробці та експлуатації розподіленого програмного забезпечення.
- здатність оцінювати та забезпечувати якість виконуваних робіт;

5. Результати навчання за дисципліною:

Результат навчання (РН) (1. знати; 2. вміти; 3. комунікація*; 4. автономність та відповідальність*)		Форми (та/або методи і технології) викладання і навчання	Методи оцінювання та пороговий критерій оцінювання (за необхідності)	Відсоток у підсумковій оцінці з дисципліни
Ко д	Результат навчання			
РН 1.1	Знати основні поняття, патерни, технології розробки програмних систем із використанням віддаленої взаємодії на основі клієнт-серверної архітектури	Лекції, лабораторні роботи	Тест(3%), екзамен(10%)	13%
РН 1.2	Знати суть і підгрунтя сервісно-орієнтованої архітектури, технології <i>Web</i> -сервісів, використання <i>REST</i> -обмежень у розподілених програмних системах.	Лекції, лабораторні роботи	Тест(3%), екзамен(15%)	18%
РН 1.3	Знати суть і підгрунтя мовних засобів, фреймворків та інструментальних середовищ розробки розподілених програмних систем, орієнтованих на віддалену взаємодію із використанням <i>Web</i> -технологій, мобільних платформ, хмарних обчислень.	Лекції, лабораторні роботи	Тест(4%), екзамен(15%)	19%
РН 2.1	Вміти застосовувати на практиці технології розробки програмних систем, що орієнтовані на клієнт-серверну взаємодію	Лабораторні роботи, самостійна робота	Захист лабораторних робіт	10%
РН 2.2	Вміти застосовувати технологію <i>Web</i> -сервісів і, зокрема, <i>REST Web</i> -сервісів у програмних системах.	Лабораторні роботи, самостійна робота	Захист лабораторних робіт	20%

* заповнюється за необхідністю, наприклад для практик, лабораторних курсів тощо.

РН 2.3	Отримання навичок використання мовних засобів, фреймворків та інструментальних середовищ підтримки віддаленої взаємодії із залученням <i>Web</i> -технологій, мобільних платформ, хмарних обчислень.	Лабораторні роботи, самостійна робота	Захист лабораторних робіт	20%
---------------	--	---------------------------------------	---------------------------	-----

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання

Програмні результати навчання	1.1	1.2	1.3	2.1	2.2	2.3
<i>(з опису освітньої програми)</i>						
ПР9. Використовувати інструментальні засоби розробки клієнт-серверних застосувань, проектувати концептуальні, логічні та фізичні моделі баз даних, розробляти та оптимізувати запити до них.	+	+	+	+	+	+
ПР14. Застосовувати знання методології та CASE-засобів проектування складних систем, методів структурного аналізу систем, об'єктно-орієнтованої методології проектування в процесі побудови і практичного застосування функціональних моделей організаційно-економічних і виробничо-технічних систем.	+	+	+	+	+	+

7. Схема формування оцінки.

7.1 Форми оцінювання студентів:

- семестрове оцінювання:

1. Тести: РН 1.1., РН 1.2, РН 1.3 — $10(3+3+4)/6$ балів.
2. Лабораторні роботи : РН 2.1 — $10(5+5)/6$ балів.
2. Лабораторні роботи : РН 2.2 — $20(5+5+5+5)/12$ балів.
4. Лабораторні роботи : РН 2.3 — $20(5+5+5+5)/12$ балів.

- підсумкове оцінювання (у формі екзамену):

- максимальна кількість балів які можуть бути отримані студентом: 40 балів;
- результати навчання які будуть оцінюватись: РН1.1, РН1.2, РН1.3;
- форма проведення і види завдань: письмова

Структура екзаменаційної роботи та критерії оцінювання:

2 теоретичних питання (по 12 балів), 1 задача (16 балів)

Запитання для підготовки до екзамену

1. Поняття програмної інженерії. Моделювання у програмній інженерії.
2. Життєвий цикл програмних систем (ПС). Моделі життєвого циклу ПС. Ітеративно-інкрементні моделі життєвого циклу. Керування ризиками.
3. Візуальне моделювання. Моделювання та CASE-технології. Уніфікована мова моделювання UML. Призначення UML у розрізі проектування ПС. Види діаграм UML. Спрощена стратегія використання UML-діаграм при моделюванні ПС.
4. Засоби розширення UML: стереотипи (stereotype), помічені значення (tagged value), обмеження (constraint). Профілі предметних областей.

5. Діаграми прецедентів. Моделювання контексту та вимог до ПС. Прецеденти. Специфікація прецедентів у Rational Rose. Потіки подій та сценарії.
6. Актори, основні актори. Відношення між акторами та прецедентами. Відношення узагальнення для прецедентів та акторів.
7. Організація прецедентів. Відношення залежності між прецедентами. Відношення включення (include) та розширення (extend). Варіанти діаграм прецедентів.
8. Реалізація прецедентів. Використання діаграм послідовностей. Анатомія діаграм послідовності. Двохетапне розроблення діаграм послідовностей. Узгодженість (цілісність) моделей. Діаграми класів-учасників VOPC (View of Participating Classes) прецедентів.
9. Використання класів при проектуванні ПС. Класи етапу аналізу: прикордонні (boundary) або інтерфейсні класи, класи-сутності (entity), управляючі (control) класи (класи-менеджери). Класи етапу проектування. Діаграми співробітництва (collaboration) та їх використання.
10. Відношення між класами та їх виявлення (узагальнення, залежність, асоціація, агрегація, композиція). Проектування класів, відношень між класами. Проектування атрибутів та операцій. Пакетування класів.
11. Діаграми класів та патерни проектування. Структура патернів. Класифікація патернів: породжуючі, структурні, поведінкові.
12. Приклади патернів: “Singleton”, “Adapter”, “Proxy”, “Decorator”, “Composite”, “Bridge”, “Observer”.
13. Патерн IOC&DI. Ілюстративний приклад.
14. Патерн IOC&DI. Spring: IoC + декларативний стиль.
15. Використання діаграм класів для кодогенерації. Кодогенерація та реінженіринг.
16. Платформа .NET. Основні складові частини CLI. Середовище виконання .NET. Загальна система типів CTS.
17. Поняття керованого коду. .NET-компіляція. Міжмовна інтеграція у .NET.
18. Метадані. Самоопис керованого коду .NET. Механізм рефлексії.
19. Мова CIL. Just In Time (JIT) компіляція.
20. Платформа .NET. Збірки .NET.
21. Платформа .NET. Управління пам'яттю. Збирання сміття.
22. Типи об'єктів для віддаленої взаємодії. Домени. Типи marshal-by-value. Сериалізація. Типи marshal-by-reference.
23. Серверна активізація. Режими Singleton та SingleCall. Клієнтська активізація.
24. Канали. Стандартні типи каналів: TcpChannel та HttpChannel. Конфігурування інфраструктури .NET Remoting: програмне; з використанням конфігураційних файлів.
25. Основи управління часом життя об'єктів .NET.
26. Веб-служби (Web Services) та сервісно-орієнтована архітектура (COA). Стандарти веб-служб.
27. Документування веб-служб: генерація документації для сприйняття людиною (з використанням веб-браузерів), генерація документації, орієнтованої на використання програмами – wsdl-файли.
28. Розробка веб-служб на платформі .NET. Директива @ WebService. Тест-форми веб-служб. Утиліта .NET WebService Studio як універсальний клієнт.
29. Розробка веб-служб на платформі Java.
30. Розробка клієнтських програм для веб-служб на платформі .NET. Утиліта Wsd.exe.
31. Розробка клієнтських програм для веб-служб на платформі Java.
32. Протокол SOAP. Конверт, заголовок SOAP-повідомлення.
33. Стандарти XML, XML-Schema. Простори імен, монікери XML.
34. Структура wsdl-файлів. Оркестровка Web-сервісів. BPEL (BPEL4WS – Business Process Execution Language for Web Services). Візуальне проектування BPEL-програм.
35. Основи архітектури WCF. Кінцеві точки.
36. Прив'язки WCF. Стандартні прив'язки.

- 37.Метадані WCF-служб.
- 38.WCF. Behavior. Режими інстанціації (instance).
- 39.WCF. Підтримка асинхронних викликів. Використання транзакцій.
- 40.WCF. DataContract.
- 41.Сумісність служб ASMX та WCF.
- 42.REST веб-служби. Ідеологія відповідного архітектурного стилю. JAX-RS (Java API for RESTful Web Services).
- 43.REST веб-служби. Приклад проекту із використанням Http Get та Http POSTзапитів.
- 44.REST веб-служби. Приклади різних репрезентацій для Http Get запитів. Анотація @Produces.
- 45.REST веб-служби. Класи ресурсів та їх використання. Анотації @ApplicationPath, @Path.
- 46.REST веб-служби. Форматування ресурсних даних. Анотації @XmlRootElement, @XmlType, @XmlElement.
- 47.Спрощена архітектура RMI. Особливості програмування RMI/JRMP-проектів.
- 48.“Віддалені” інтерфейси та класи реалізації “віддалених” інтерфейсів.
Порівняння RMI/JRMP та RMI/IIOP. “Експортування” об'єктів. Статичний метод ExportObject.
- 49.RMI/IIOP-проекти. Використання gmic та orbd.
- 50.JNDI та конкретні служби іменування. Особливості налаштування JNDI.
- 51.(JRMP – IIOP) портабельність Java RMI-проектів.
- 52.Сумісність RMI/IIOP- та CORBA-проектів.
- 53.Поняття web-проектів. Сервлети. Життєвий цикл сервлетів. GenericServlet. HttpServlet.
- 54.Контейнерне управління сервлетами (web-сервери; типи HttpServletRequest, HttpServletResponse; використання декларативних засобів у web-проектах; дескрипторний файл web.xml, анотація @WebServlet.
- 55.Сервлетні об'єкти їх особливості та використання (отримання HTTP-сторінок; використання HTTP-get та HTTP-post команд у web-проектах; singleton-властивість сервлетних об'єктів; особливості підтримки клієнтських сесій; класи HttpSession, Cookie; JSESSIONID-cookie).
- 56.Web-проект для спрощеної моделі електронного магазину.
- 57.Java Server Pages (JSP), їх особливості та використання (трансляція у сервлети; виклики JSP із браузера та сервлетів, "перенаправлення" запитів JSP-сторінці; використання MVC-архітектури при web-проектуванні та роль JSP у ній).
- 58.Основи Action-oriented Frameworks: диспетчерський сервлет, mapping, action-класи.
- 59.Використання технології action-класів у web-проектуванні. Приклад.
- 60.Struts, WebWork, Spring як приклади Action-орієнтованих фреймворків.
- 61.ASP.NET-проекти. Життєвий цикл сторінок.
- 62.Структура ASP.NET-проектів та їх розробка. Використання технології відокремлення коду. Засоби візуального проектування, управляючі елементи (контроли). Валідація даних.
- 63.Проблеми postback-запитів та можливості збереження стану сторінок. Сховані поля ASP.NET-проектів. Використання ViewState та ControlState. Можливості використання Cookie, Session-та Application-контейнерів.
- 64.Знайомство з AJAX на модельному проекті електронного магазину.
- 65.Засоби ASP.NET AJAX Extensions.
- 66.Приклад Java-проекту з AJAX на основі JSP та сервлетів.
- 67.Фреймворк Google Web Toolkit (GWT).
- 68.AJAX-проекти із використанням GWT.
- 69.Розробка Android-програм. Android Developer Tools, Android SDK Manager, Virtual Device Manager. Архітектура Android OS. Розробка клієнтів веб-служб на платформі Android.
- 70.Можливості візуального проектування програм під Android OS. Активності (activities) та розмітки (layouts). Віджети. Програмне оперування віджетами. Приклад.

71. Android OS. Управління активностями. Передача даних в іншу активність. Динамічне наповнення активності віджетами. Приклад.
72. Хмарні обчислення. Класифікація хмарних сервісів (Platform as a Service, Infrastructure as a Service, Software as a Service). Платформа Microsoft Azure.
73. Розгортання веб-проектів у хмарі (на прикладі).

Для отримання загальної позитивної оцінки з дисципліни оцінка за екзамен не може бути меншою 24 балів

Студент не допускається до екзамену, якщо під час семестру набрав менше ніж 20 балів.

7.2 Організація оцінювання:

Терміни проведення форм оцінювання:

1. Тест РН 1.1. : до 5 тижня семестру.
2. Тест РН 1.2. : до 11 тижня семестру.
3. Тест РН 1.3. : до 15 тижня семестру.

Лабораторні роботи — щотижня, починаючи з другого навчального тижня

У випадку відсутності студента з поважних причин відпрацювання та перездачі МКР здійснюються у відповідності до „Положення про порядок оцінювання знань студентів при кредитно-модульній системі організації навчального процесу” від 1 жовтня 2010 року.

У разі неякісного виконання лабораторної роботи, викладач має право не зарахувати лабораторну роботу, або знизити за неї бали.

Студент має право здавати лабораторні роботи після закінчення визначеного для них терміну, але з втратою одного балу за кожен тиждень, який пройшов з моменту закінчення терміну її здачі.

7.3 Шкала відповідності оцінок

Відмінно / Excellent	90-100
Добре / Good	75-89
Задовільно / Satisfactory	60-74
Незадовільно / Fail	0-59
Зараховано / Passed	60-100
Не зараховано / Fail	0-59

8. Структура навчальної дисципліни. Тематичний план лекцій і лабораторних занять

№ лекції	Назва лекції	Кількість годин		
		Лекції	Лаб. занят.	Самост. робота
	Змістовий модуль 1. Об'єктна парадигма та технології віддаленої взаємодії			
1	Тема 1. Уніфікована мова моделювання <i>UML</i> . Спрощена стратегія використання <i>UML</i> -діаграм при моделюванні програмних систем (ПС).	2	1	6
2	Тема 2. Огляд технологій <i>J2EE</i> . Технологія <i>Java RMI</i> . Версії <i>JRMP</i> та <i>IIOP</i> .	2	1	6
3	Тема 3. (<i>JRMP</i> – <i>IIOP</i>) <i>портабельність Java RMI-проектів</i> . <i>Сумісність RMI/IIOP- та CORBA-проектів</i> . Технологія <i>Java IDL</i> .	2	1	6
4	Тема 4. <i>Міжмовна інтеграція у .NET</i> . <i>Віддалена взаємодія об'єктів .NET (.NET-Remoting)</i>	2	2	6
5	Тема 5. Сервісно-орієнтована архітектура. Веб-служби на платформі <i>.NET</i> .	2	2	8
6	Тема 6. Веб-служби на платформі. <i>Java</i>	2	2	6
7	Тема 7. Технології <i>Windows Communication Foundation</i>	2	2	6
8	Тема 8. <i>REST веб-служби</i> .	2	2	6
	Змістовий модуль 2. Розробка програмних систем із використанням віддаленої взаємодії			
9	Тема 9. <i>Web-технології Java Servlet та JSP</i>	2	1	6
10	Тема 10. <i>Web-розробка із використанням ASP Net</i>	2	1	6
11	Тема 11. <i>Web MVC- проектування</i> . Технологія <i>action servlet</i> . Поняття фреймворку. <i>Action-oriented Frameworks</i> . Огляд <i>Web MVC</i> фреймворків.	2	2	6
12	Тема 12. Проблема валідації даних у <i>Web</i> -проектах. Підтримка <i>AJAX</i> .	2	1	6
13	Тема 13. Фреймворк <i>Spring</i> , патерн (принцип) <i>IOC&DI</i>	2	2	8
14	Тема 14. Програмування під мобільні платформи. Приклад розробки під <i>OS Android</i>	2	2	6
15	Тема 15. Хмарні обчислення та технології. Платформа <i>MS Azure</i> .	2	2	8
	ВСЬОГО	30	24	64
	Консультація		2	
	Екзамен			

Загальний обсяг 120 год., в тому числі:

Лекцій – **30 год.**

Лабораторні заняття - **24 год.**

Консультації – **2 год.**

Самостійна робота - **64 год.**

9. Рекомендовані джерела:

Основні:

1. Богте У., Богте М., Дранишников И., UML и Rational Rose, «Лори», М., 2008, 600с.
2. Соммервилл И., Инженерия программного обеспечения. М, «Вильямс», 2002.- 624 с.
3. Лавріщева К. Програмна інженерія, К., 2008, 319 с.
4. Хохгуртль Б., C# и Java: межплатформенные Web-сервисы, М., Кудиц-образ, 2004, 410с.
5. Машнин Т.С. Web-сервисы Java, БХВ-Петербург, 2012, 560 с.
6. Erik Wilde, Cesare Pautasso. REST: From Research to Practice, 2011, 528 p.
7. Джон Фландерс. Введение в службы RESTful с использованием WCF. MSDN, 2009. (msdn.microsoft.com/ru-ru/magazine/dd315413.aspx)
8. Перри Б., Java сервлеты и JSP: сборник рецептов, М., КУДИЦ-Пресс, 2009, 768с.
9. Хемрадхани А. Гибкая разработка приложений на Java с помощью Spring, Hibernate и Eclipse. — М.:Издательский дом "Вильямс", 2008, 352 с.

Додаткові:

1. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++. Второе издание М.: Бином, СПб.: Невский диалект, 2000.
2. Якобсон А., Буч Г., Рамбо Дж. Унифицированный процесс разработки программного обеспечения – СПб.: Питер, 2002.
3. Коберн А. Современные методы описания требований к системам.– М.: Лори, 2002.
4. Буч. Г. , Рамбо Дж. , Джекобсон А. Язык UML. Руководство пользователя –М.: ДМК, 2000.
5. Рамбо Дж., Якобсон А., Буч Г. UML: Специальный справочник – СПб.: Питер, 2002.
6. Фаулер М., Скотт К. UML в кратком изложении – М.: Мир, 1999.
7. Кватрани Т. Rational Rose 2000 и UML. Визуальное моделирование. М.: ДМК, 2001.
8. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирования. Паттерны проектирования – СПб.: Питер-ДМК, 2001.
9. Эммерих В. Конструирование распределенных объектов – М.: Мир, 2002.
10. Троелсен Э. C# и платформа .NET.— СПб.: Питер, 2004.—796 с.
11. Платт Д.С. Знакомство с Microsoft .NET— М.: Издательско-торговый дом «Русская Редакция », 2001.— 240с.
12. Просиз Дж. Программирование для Microsoft .NET — М.: Издательско-торговый дом "Русская Редакция", 2003.— 704 стр.
13. Рихтер Дж. Программирование на платформе Microsoft .NET Framework — М.: Издательско-торговый дом «Русская Редакция », 2003 — 512 стр.
14. Шилдт Г. , Холмс Дж. Искусство программирования на Java.
15. Хорстманн К., Корнелл Г. .Java 2. Библиотека профессионала. Том 2. Тонкости программирования. — М.:Издательский дом "Вильямс", 2002.— 1120 с.
16. Ахмед Х.З., Амриш К.Е. Разработка корпоративных Java-приложений с помощью J2EE и UML . К.: Вильямс, 2002.
17. <http://msdn.microsoft.com>
18. <https://developers.google.com/appengine/>
19. <http://www.hibernate.org/>
20. <http://www.springframework.org/>
21. <http://developer.android.com/index.html>