ЗАТВЕРДЖУЮ Начальник кафедри інформаційних технологій та телекомунікаційних систем, к.т.н., доцент Олександр ПРИДАТКО «___»____20___ р.

МЕТОДИЧНА РОЗРОБКА

для проведення практичного заняття з студентами (курсантами) 1-го курсу спеціальності 122 «Комп'ютерні науки» з дисципліни «Основи програмування»

Практичне заняття 5.4: «Перший за стосунок в IDE Eclipse/ IntelliJ IDEA»

Мета: здобути навики щодо створення нового Java-проекту та написання найпростішої програми із виводу текстової стрічки у консоль. Закріпити теоретичні знання щодо структури програмного коду на Java.

Час: 2 годин

Місце заняття: комп'ютерна лабораторія

Матеріальне забезпечення:

- ПК з відповідним ПЗ та доступом до мережі Internet;

- Методичні рекомендації щодо виконання практичної роботи.

Після практичного заняття студенти (курсанти) повинні:

вміти: реалізовувати найпростіші програми із виводу текстових стрічок у консоль середовища розробки Eclipse.

знати: порядок створення нового Java-проекту та його структури, а також структури програмного коду написаного мовою Java.

Література:

1. Віртуальне навчальне середовище ЛДУ БЖД «Віртуальний університет» [Електронний ресурс]. – Доступний з : http://virt.ldubgd.edu.ua/course/view.php?id=1528

2. Head First Java (изучаем Java): пер. с англ. / Kathy Sierra, Bert Bates. – Москва : «Эксмо», 2012. – 718 с.

План заняття:

1. Перший застосунок в середовищі розробки

- 2. Структура коду програми
- 3. Видача індивідуальних практичних завдань

2.1. Перший застосунок в середовищі розробки

В якості першого застосунку реалізуємо найпростішу програму із виводу на екран стрічки «Hello world!». Спершу реалізуємо застосунок без використання IDE (з метою повноти уяви про трудомісткість процесу програмування без IDE).

Для написання коду програми запускаємо звичайний Notepad (як згадувалось в лекції) та набираємо код програми (про логіку коду пізніше):

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello world");
    }
}
```

Далі необхідно зберегти цей файл з розширенням .java та назвою, яка відповідає назві класу – HelloWorld. Місце збереження файлу не має значення, проте з метою полегшення процесу доступу до файлу, рекомендовано обрати місце збереження вище до кореневого каталогу. В нашому прикладі створено папку «work» в кореневому каталозі С:\.

Наступним кроком у виконанні програми є запуск командного рядка. Для виконання цієї операції необхідно у вікні пошуку програмного меню «Пуск» ввести «cmd».



За замовчування командний рядок завантажується із доступом до папки користувача. З метою зміни адреси доступу (переходу до підкаталогу *вищого* рівня) необхідно виконати команду «cd..» потрібну кількість разів. Для переходу до підкаталогу *нижчого* рівня необхідно застосувати команду «cd» після якої вказувати ім'я наступного за ієрархією підкаталогу. В нашому прикладі потрібно здійснити перехід до підкаталогу «work» кореневого каталогу С:\, як це зображено на прикладі:



В папці «work» попередньо збережено файл з кодом програми HelloWorld.java. Переконатись в цьому можливо з допомогою командного рядка. З цією метою застосовують команду «dir»:

C:\work>dir

В результаті виконання команди буде виведено інформацію про відповідний підкаталог та розміщені у ньому файли:

📾 Администратор: C:\Windows\system32\cmd.exe	×
Microsoft Windows [Version 6.1.7601] Copyright (c) 2009 Microsoft Corporation. All rights reserved.	^
C:\Users\napt>cd	
C:\Users>cd	
C:\>cd work	
C:\work>dir Том в устройстве С не имеет метки. Серийный номер тома: 36F1-BACE Содержимое папки C:\work	
27.09.2017 18:15 <dir> 27.09.2017 18:15 <dir> 27.09.2017 12:48 120 HelloWorld.java 27.09.2017 13:44 0 Program 2 файлов 120 байт 2 папок 17 709 678 592 байт свободно</dir></dir>	
C:\work>	

Як видно з представленого прикладу в зазначеній папці міститься файл HelloWorld.java. Наступним кроком у виконанні програми є компіляція файлу з кодом програми.

Для того, щоб зв'язати зазначений файл з компілятором java, необхідно вказати шлях доступу до javac. Місце розміщення компілятора відповідає директорії встановлення JDK, в зазначеному прикладі це:

```
C:\work>c:\Java\jdk1.8.0_144\bin\javac
```

a6o: C:\work> "c:\Program Files\Java\jdk1.8.0_144\bin\javac"

Для виконання компіляції, після зазначення адреси розміщення javac, необхідно вказати ім'я файлу з розширенням за прикладом:

C:\work>c:\Java\jdk1.8.0_144\bin\javac HelloWorld.java aбo: C:\work> "c:\Program Files\Java\jdk1.8.0_144\bin\javac" HelloWorld.java (ОДНА команда!)

Після виконання компіляції, в командному рядку не відбудеться жодних змін, проте за адресою підкаталогу з вихідним кодом програми з'явиться ще один файл з ідентичною назвою та іншим розширенням (.class) – це файл з відкомпільованим байт-кодом програми. Переконатись в цьому можливо шляхом повторного застосування команди «dir»:

🖼 Администратор: C:\Windows\system32\cmd.exe	- 0	x
27.09.2017 18:15 <dir> 27.09.2017 18:15 <dir> 27.09.2017 12:48 120 HelloWorld.java 27.09.2017 12:48 0 Program 2 файлов 120 байт 2 папок 17 709 678 592 байт свободно</dir></dir>		^
C:\work>c:\Java\jdk1.8.0_144\bin\javac HelloWorld.java		
C:\work>dir Том в устройстве С не имеет метки. Серийный номер тома: 36F1-BACE		
Содержимое папки C:\work		
27.09.2017 18:29 <dir> . 27.09.2017 18:29 <dir> . 27.09.2017 18:29 <dir> . 27.09.2017 18:29 438 HelloWorld.class 27.09.2017 12:48 120 HelloWorld.java 27.09.2017 13:44 0 Program 3 файлов 558 байт 2 папок 17 826 455 552 байт свободно</dir></dir></dir>		
C:\work>		*

З метою виводу в командний рядок результатів компіляції необхідно виконати дії із зазначенням місця розташування JVM, яка виконуватиме відповідний байткод:

C:\work>c:\Java\jdk1.8.0_144\bin\java HelloWorld



Результатом роботи програми буде вивід стрічки без відтворення деяких символів (або взагалі набір незрозумілих символів). Це результат виконання невідповідної кодової сторінки, для заміни якої необхідно повернутись до каталогу користувача, в нашому випадку це C:\Users\парт та використати команду:

C:\Users\парт>**chcp 1251**

Або ж реалізувати виконання цієї команди перебуваючи безпосередньо в каталозі «work»:

C:\work>chcp 1251

Після чого повернутись повторно виконати програму:

C:\work>c:\Java\jdk1.8.0_144\bin\java HelloWorld

Результатом роботи програми буде вивід стрічки «Привіт світе!»

🛤 Администратор: C:\Windows\system32\cmd.exe	×
C:\>cd work	~
C:\work>c:\Java\jdk1.8.0_144\bin\java HelloWorld Прив?т св?те!	
C:\work>cd	
C:\>cd Users	
C:\Users>cd napт	
C:\Users\парт>chcp 1251 Текущая кодовая страница: 1251	
C:\Users\napt>cd	
C:\Users>cd	
C:\>cd work	
C:\work>c:\Java\jdk1.8.0_144\bin\java HelloWorld Привіт світе!	
C:\work>_	-

Якщо в результаті роботи програми все ж таки одержана стрічка з незрозумілих символів, потрібно переконатись у відповідності шрифтів виводу тексту в консоль. Для цього необхідно перейти в меню командного рядка та обрати категорію «властивості». За обраною категорією в закладці «шрифт» необхідно обрати шрифт Lucida Console aбo Consolas.

Cit.	Администратор: С:\W	/indows\system32\cmd.exe	- 0	X
đ	Восстановить Переместить Размер	.8.0 <u>1</u> 44\bin\java HelloWorld		
-	Свернуть Развернуть			
x	Закрыть Изменить Умолчания	251 ница: 1251		
c.\ C:\	Свойства usersacu >cd work			
С:\ При С:\	work>c:\Java\jdk1 віт світе! work>	.8.0 <u>1</u> 44\bin\java HelloWorld		Ŧ

🖬 Свойства: "C:\Windows\system32\cmd.exe"					
Общие Шрифт	Расположение	Цвета			
Вид окна		Размер			
	Ľ	12 14 16 18 20 24 28 36			
Шрифт 🔲 X Tr Consolas To Lucida Console Точечные шрифты	Кирный				
Выбранный шриф C:\WINDOWS> SYSTEM SYSTEM32 DEADME TYT	or:Lucida Console dir <dir> 1 <dir> 1 25025 1</dir></dir>	Размер знако Ширина (пик Высота (пико	ов: сели): 7 сели): 12		
		ОК	Отмена		

Слід зауважити, що в разі допущення в первинному коді програми будь-якої помилки, текстовий редактор цього не повідомить. Відстежити посилку можливо лише після компіляції програми з допомогою командного рядка. До прикладу, допустимо помилку в методі println :

```
public class HelloWorld {
    public static void main (String [] args){
        System.out.prinln("Привіт світе!");
    }
}
```

В результаті компіляції буде отримано:



Для виправлення помилки необхідно повернутись до текстового редактора, виправити помилку та повторно запустити програму на компіляцію з допомогою командного рядка.

Власне незручність при роботі з помилками та трудомісткість процесу компіляції є основними недоліками такого методу програмування. З метою полегшення цього процесу використовують інтегровані середовища розробки, які орієнтовані на автоматичну перевірку синтаксису коду під час його написання та швидку компіляцію програми із виводом результату в консоль.

З метою наочності реалізуємо програму виводу стрічки «Привіт світе!» з використанням IDE.

Під час запуску інтегрованого середовища розробки Eclipse необхідно зазначити каталог де зберігатимуться усі Java-проекти. Якщо середовище щойно встановлене та раніше не використовувалось, то таку директорію необхідно створити та вказати під час першого запуску середовища (створення каталогу доступне під час запуску IDE).



Після першого запуску Eclipse область Packege Explorer, звичайно, буде пустою. Для створення першого проекту необхідно перейти в меню File>New>Project. В вікні створення нового проекту необхідно вибрати Java>Java Project та пройти далі.

			🗢 Java - Java - Eclipse	particle 2.6 days Press, represented in programmer
Java - Java - Eclipse	arch Project	Run Window Help	File Edit Source Refactor Navigate Search Project Run Window Help	a (a v cà v
New A	lt+Shift+N >	🖄 Java Project	I Package Explorer ⊠ E S V - □	
Open File		Project		🗢 New Project 🖂 🖂 🛛
😋 Open Projects from File System		Package		Select a wizard
Close	Ctrl+W	Class		Create a Java project
Close All C	trl+Shift+W	🕜 Interface		Wizards: type filter text
I Save	Ctrl+S	🚱 Enum		> 😓 General
Save As		Annotation		4 😂 Java Project
🐻 Save All 🛛 🖸	Ctrl+Shift+S	Source Folder		Java Project from Existing Ant Buildfile Mayen Comparison
Revert		🍐 Java Working Set		V 🖉 Comples
Move		Folder		
🕑 Rename	F2	Untitled Text File		
🔊 Refresh	F5	JUnit Test Case		
Convert Line Delimiters To	+	💣 Task		
Print	Ctrl+P	📸 Example		
Switch Workspace	÷	Other Ctrl+N		
Restart				
import				
Export				Problems 52 @ Javadoc (B. Declaration
Properties	Alt+Enter			0 items Description Resource Path Location 7
Exit				resource rear LOCAUDIT I

Наступним кроком є присвоєння імені для проекту. В цьому ж вікні можна вибрати версію Java, залежно від особливостей процесу розробки. За замовчуванням буде запропоновано останню версію JDK, яка встановлена на комп'ютері.

New Java Project	
Create a Java Project Create a Java project in the workspace or in an external location.	r
Project name: Project1	
☑ Use default location	
Location: D:\prydatko\Придаткo\prydatko_logos\Java\Project1	Browse
JRE	
Use an execution environment JRE: JavaSE-1.8	-
Use a project specific JRE: jre1.8.0_144	
O Use default JRE (currently 'jre1.8.0_144')	Configure JREs
Project layout	
Use project folder as root for sources and class files	
Oreate separate folders for sources and class files	Configure default
Working sets	
Add project to working sets	New
Working sets:	▼ Select
? < Back Next > Finite	ish Cancel

Јаva-проекти створюють з метою розміщення класів, проте деякі проекти можуть мати велику кількість класів (особливості об'єктно-орієнтованого програмування), тому з метою їх логічної структуризації в Java-проект введено елемент пакету. Для створення пакету необхідно знову звернутись до меню File>New>Packege (за умови виділеної папки з проектом) або натиснути на директорію пакету правою клавішею маніпулятора (миші) та виконати New>Packege. Аналогічним чином в межах пакету створюється клас. Нагадуємо, що клас являється основним файлом, що містить первинний код програми.

	ava - Java - Eclipse				
File	Edit Source Refactor Navigate	Search Proje	ect Ru	ın Window H	ielp
	New	Alt+Shift+N	→ 🛃	Java Project	
	Open File			Project	
	Open Projects from File System		ŧ	Package	
	Close	Ctrl+W	C	Class	
	Close All	Ctrl+Shift+W	0	Interface	
	Save	Ctrl+S	G	Enum	
	Save As		0	Annotation	
B	Save All	Ctrl+Shift+S	s #	Source Folder	
	Revert		2	Java Working	Set
	Move			Folder	
-9	Rename	F2	2	File	ila
8	Refresh	F5	5	IUnit Test Cas	e
	Convert Line Delimiters To		> ~	Task	-
æ	Print	Ctrl+P		Evenuele	
_	Cutteb Wadana a			j Example	
	Switch workspace		7 🗖	Other	Ctrl+N
	NC3tar C				
è	Import				
È	Export				
	Properties	Alt+Enter	r		
	Exit				

Після вибору позиції «створити новий проект», середовище запропонує дати назву новостворюваному пакету.



Новостворений пакет буде автоматично розміщений в каталозі «src». Слід зауважити, що в каталозі «src» буде збережено усю структуру Java-проекту. Крім того, при створенні нового проекту автоматично створиться бібліотека JRE System Library, яка потрібна для запуску програми.

Дещо інший порядок створення класу. Виконавши аналогічні дії за умови виділеного новоствореного пакету (File>New>Class), середовище виведе вікно де необхідно зазначити назву класу. **Важливо!!!** Усі класи прийнято називати з великої літери. В прикладі це буде клас Main (головний). В зазначеному вікні користувачеві надається можливість встановити модифікатор доступу (про це пізніше), вибрати в якості місця збереження інший пакет, автоматично створити головний метод, генерувати конструктор або коментарі тощо. Для початку ми не будемо користуватись усіма представленими можливостями, окрім того, що встановимо позначку навпроти автоматичної генерації методу «public static void main (String [] args)» (щоб не набирати її потів власноруч).

lava Class		0
This package nan with a lowercase	e is discouraged. By convention, package names usu letter	ally start
Source folder:	Project1/src	Browse
Package:	Packege1	Browse
Enclosing type:		Browse
Name:	Main	
Modifiers:	public package private pro abstract final static	tected
Superclass:	java.lang.Object	Browse
Interfaces:		Add
		Remove
Which method stubs	would you like to create?	
	vublic static void main(String[] args)	
	Constructors from superclass Inherited abstract methods	
Do you want to add	comments? (Configure templates and default value <u>h</u>	ere)
	enerate comments	
0		

Переваги використанняи IDE для створення класів очевидні. В результаті автоматичної генерації головного методу користувачеві не потрібно писати код із назвою класу, оголошувати метод, слідкувати за синтаксисом, щоб не допустити помилки тощо. Усі ці операції Eclipse виконує автоматично.

У результаті проведених дій в Packege Explorer буде відображено новостворений метод, а в області написання коду – згенеровано головний метод:



В області написання коду Eclipse автоматично генерує стрічку:

// TODO Auto-generated method stub

Це коментар, який не відіграє в програмі жодного значення окрім інформування про певні особливості того чи іншого методу. В нашому випадку цю стрічку можна видалити.

З метою формування власного коментаря необхідно в любій довільній області виділити необхідний текст та натиснути комбінацію клавіш Ctrl+/ (за умови англійської розкладки клавіатури).

Якщо звернутись до первинного коду програми із виводу стрічки «Привіт світе!» (який було написано раніше), то можна відстежити, що у генерованому в Eclipse варіанті не вистачає лише (потрібно дописати):

System.out.println("Hello world");

```
Main.java ⊠
1 package Packege1;
2
3 public class Main {
4
5⊖ public static void main(String[] args) {
6 System.out.println("Привіт світе!");
7
8 }
9
10
11
```

Завершивши написання коду необхідно зберегти напрацювання (File>Save або Ctrl+S) та запустити компілятор. Запуск коду на компіляцією із подальшим виконанням програми та виведенням результату в консоль здійснюється з допомогою натискання кнопки Run, меню Run>run, або комбінацією клавіш Ctrl+F11.

Eclipse - Java - DevJava1/src/Package1/Main.java - Eclipse	and in the local lines.	Eclipse - Java - DevJava1/src/Package1/Main.java - Eclipse	of the local division in which the real of the local division in which the local division in the local divisio	-
File Edit Source Refactor Navigate Search Project Run	Window Help	File Edit Source Refactor Navigate Search Project	Run Window Help	
New Alt+Shift+N ► Open File	• : ∲ 💋 ⋧ 😼 🎚 🎹 : 灯 + 🖗 + 🗘 + ⇒	□ • □ □ • • • • • ● <td< th=""><th>Run 🗞 Debug</th><th>Ctrl+F11 F11</th></td<>	Run 🗞 Debug	Ctrl+F11 F11
Close Ctrl+W Close All Ctrl+Shift+W Save Ctrl+S	<pre>package Package1; public class Main { public static void main(String[] args) { System.out.println("Привіт csire!");</pre>	 	Run History Run As Run Configurations Debug History Debug As	+
Save All Ctrl+Shift+S Revert	}		Debug Configurations Toggle Breakpoint	Ctrl+Shift+B
Idove F2 Rename F2 Refresh F5 Convert Line Delimiters To →			Toggle Line Breakpoint Toggle Method Breakpoint Toggle Watchpoint Skip All Breakpoints	Ctrl+Alt+B
Print Ctrl+P Switch Workspace Restart			Remove All Breakpoints Jg Add Java Exception Breakpoint Head Class Load Breakpoint	
Import Export Properties Alt+Enter			All References All Instances Instance Count Watch	Ctrl+Shift+N
1 Main java [DevJava1/src/Package1] 2 PrintStream class [java.io.PrintStream] 3 Main1.java [DevJava1/src/Package1] Exit	e Declaration Declaration Console 23		Inspect Display Execute Force Return External Tools	Ctrl+Shift+I Ctrl+Shift+D Ctrl+U Alt+Shift+F

В результаті роботи програми в консоль буде виведено стрічку «Привіт світе!».



При допущенні помилки в процесі написання коду, компілятор автоматично повідомлятиме користувача про це:

```
🚺 *Main.java 🔀
  1 package Packege1;
  2
  3
     public class Main {
  4
  50
         public static void main(String[] args) {
1 6
             System.out.prinln("Привіт світе!");
  7
X
  8
         ł
  9
 10
 11
```

Існує два типи помилок, синтаксична та логічна. Синтаксичні помилки пов'язані недотриманням правил написання коду, наприклад відсутня дужка, крапка з комою, лапки тощо. Логічні помилки це помилки у написання логіки коду, наприклад застосування не того методу, невірна ініціалізація змінної, або ж помилка у написанні методів (як зроблено в прикладі з методом println). В обох варіантах компілятор надає допомогу щодо можливих варіантів виправлення помилок.

Для того щоб скористатись допомогою компілятора у виправлені помилки достатньо натиснути на піктограму, яка сигналізує про неї. В результаті чого середовище надає перелік можливих варіантів виправлення помилки у вигляді контекстного меню:



Приклад реалізації програми у інтегрованому середовищі IntelliJ IDEA

Для того, щоб створити проект у IntelliJ IDEA вибираємо *New Project -> Java -* > *Select SDK вказати jdk* (перший раз програма може попросити вказати прямий шлях до папки, в якій знаходиться jdk) –>*Next*.

	🚇 Welcome to IntelliJ II					
	Intellij IDEA 2021.3			ject Open	Get from V	
😰 New Project						
Java Maven Gradle Android IntelliJ Platform Plugin JavaFX Java Enterprise Spring Initializr Quarkus MicroProfile Ktor	Project SDK: 18 Additional Lit 18 Grood Kotlin 11 SOLS 16 Corrol	Oracle OpenJDK versii Oracle OpenJDK versii (2) Oracle OpenJDK versii (3) java version "1.8.0 Azul Zulu version 15.0. Amazon Corretto versi etto-11 Amazon Corr etto-11 Amazon Corr etto-11 (2) Amazon C njdk-17 Oracle OpenJ vnload JDK I JDK				
 Groovy Grails App Forge Kotlin Web Multi-module Project JavaScript Empty Project 			edj Previous Next			

Після чого відкриється нове вікно, де потрібно буде вибрати *Create project from template*, щоб автоматично згенерувався main-метод.



В новому вікні вказуємо назву проекту та місце, де він буде зберігатись та вибираємо *Finish*.

😫 New Project		
Project name:		
Project location:		
Base package:		
	Select project File Directory Project file will be stored inth is directory Project file will be stored into the stored into the stored into the stored into the stored above to equilibly locate it is the the stored into the stored above to equilibly locate it is the the stored into the stored above to equilibly locate it is the the stored into the stored above to equilibly locate it is the the stored into the stored above to equilibly locate it is the the stored into the stored above to equilibly locate it is the the stored above to equilibly locate it is the the stored above to equilibly locate it is the the stored above to equilibly locate it is the the stored above to equilibly locate it is the the stored above to equilibly locate it is the stored above to equilibly lo	
	Previous	

Все проект створений, залишилось реалізувати логіку. Вся логіка проекту реалізується у класах, які знаходяться у папці *scr*. При створенні проекту у вас мав би згенеруватись клас *Main.java* (у моєму випадку – Application.java) та каркас методу *main()*.



2.2. Структура коду програми

Для кращого розуміння ієрархії коду програми проведено його короткий аналіз. Для прикладу візьмемо вже добре відомий код програми із виводу стрічки «Привіт світе!»:

```
public class HelloWorld {
    public static void main (String []
    args){
        System.out.println("Привіт світе!");
    }
}

    Файл.java
    Клас
    Клас
    Метод
    Метод
    Логіка методу
    (вираз / код
    програми)
```

Для кращої уяви про структуру програмного коду поділимо його на три категорії: клас; метод; вираз. Усі з зазначених категорій виділені різними кольорами та відповідають наведеним прикладам.

Файл з первинним кодом (розширення java) містить в собі клас. Власне клас це і є програма, або її частина. Маленькі програми можуть складатись з одного класу. Усе, що віднесено до класу розміщується між фігурними дужками (так званий блок класу):

public class HelloWorld {

```
public static void main (String [] args){
System.out.println("Привіт світе!");
}
```

}

Клас може вміщувати один або декілька методів. Власне методи містять в собі усі інструкції щодо виконання програми (код програми). Методи обов'язково потрібно розміщувати в середині класів (між фігурними дужками класу).

```
public class HelloWorld {
    public static void main (String [] args){
        System.out.println("Привіт світе!");
    }
}
```

Логіка методу (інструкції щодо виконання програми) має розміщуватись в середині методу. Іншими словами, код методу – це набір його виразів (що та як він робить).

```
public class HelloWorld {
    public static void main (String [] args){
        <u>System.out.println("Привіт світе!");</u>
    }
}
```

В наведеному прикладі вираз методу виводить у консоль стрічку «Привіт світе!».

Отже, узагальнимо ієрархію коду: *клас — метод — вираз (логіка програми).*

Коли JVM (віртуальна машина Java) починає свою роботу, вона першою чергою шукає клас (власне чому клас в ієрархії коду займає перше місце). Після знаходження класу JVM шукає метод та заходить в нього. Далі JVM почерзі виконує все, що знаходиться між фігурними дужками головного методу. Будь-яка програма написана на Java має що найменше один клас та головний метод.

Тепер зоглянемо детальніше кожен елемент представленого коду:



Не потрібно намагатись відразу запам'ятати всю структуру коду та вивчити, що як називається. Основна мета – це *ознайомлення* з елементами коду. Далі в процесі вивчення основ програмування користувачі неодноразово звертатимуться до структури та ієрархії коду, та більш детально ознайомляться з ним.

2.3. Видача індивідуальних практичних завдань

Вкінці практичного заняття студенти отримують індивідуальне практичне завдання, результат виконанням якого необхідно завантажити в віртуальне навчальне середовище (категорія: контрольна частина).

Завдання полягає у написанні найпростішої програми із виводу інформації про свою родину в такому форматі:

```
Батько: Чумаченко Степан Дмитрович, 1965 року народження, підприємець;
……
Сестра: Чумаченко Надія Степанівна, 1998 року народження, студентка;
……
```

Первинний код програми зберігається у файлі з розширенням .java за адресою яку вказано під час запуску середовища розробки в каталозі «src»:

...\назва каталогу з Java-проектом\назва проекту\src\назва пакету\файл.java

Методичну розробку розробив: Заступник начальника кафедри УПІТтаТ підполковник служби цивільного захисту

Олександр Придатко

Методична розробка обговорена на засіданні кафедри УПІТтаТ Протокол № _____ від "___" _____ 20__ р.