OBJECT ORIENTED TECHNIQUES USING JAVA(ACSE0302)

Unit: 5

GUI Programming, Generics and Collections

Course Details
(B.Tech 3rd Sem /2nd Year)

Dr. Ratna Nitin Patil
Department Artificial Intelligence

Table of Contents

Conti….

Conti…

Brief Introduction

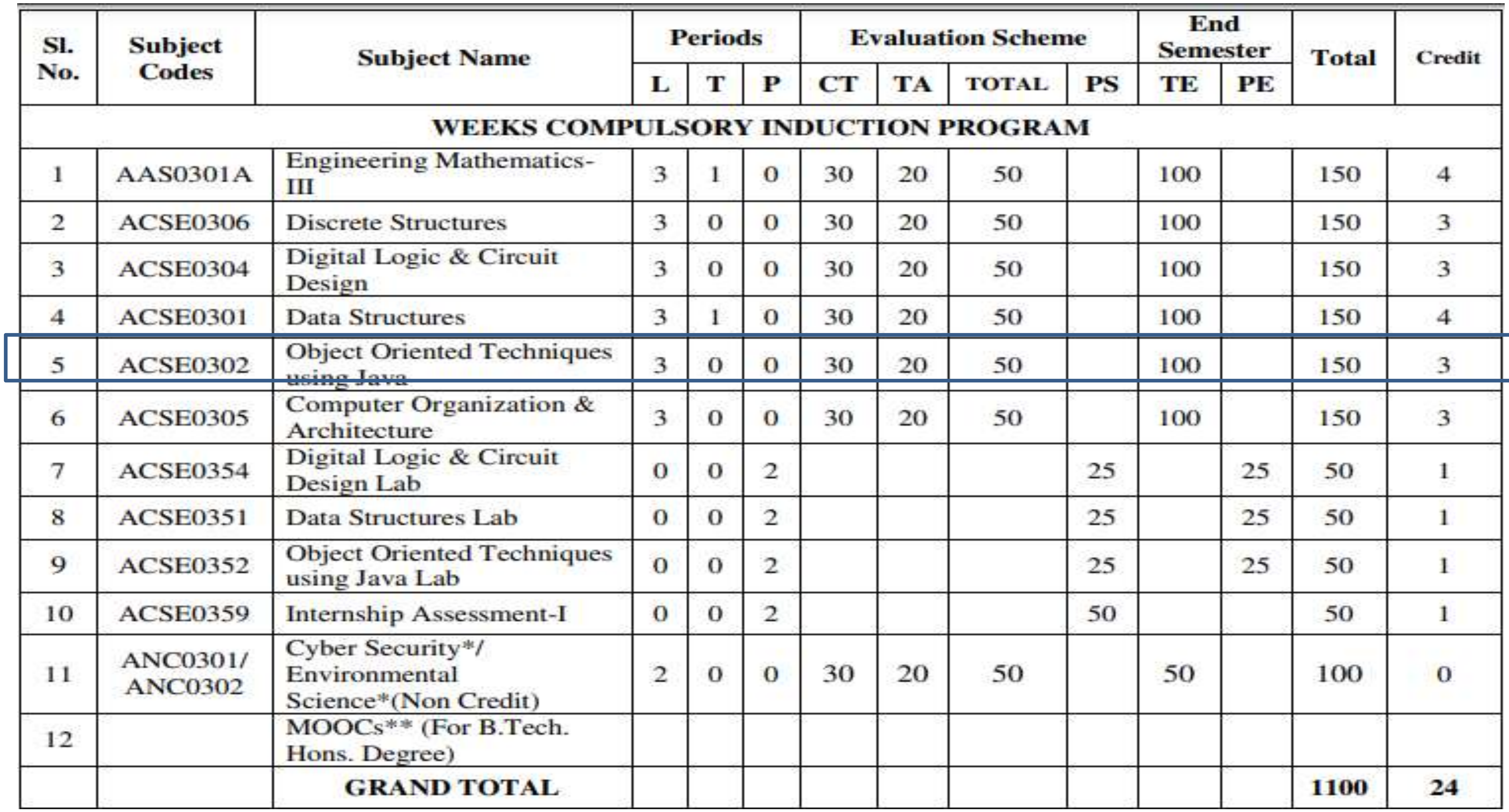
I hold Bachelor of Engineering degree from University of Pune, Maharashtra and completed Masters in Computer Engineering from Thapar University, Patiala, Punjab in 2001. I completed PhD in Computer Engineering from Dr. Babasaheb Ambedkar Marathwada University, Aurangabad in 2021.

I worked as an Associate Professor at Kanpur Institute of Technology, Kanpur, St Peters College of Engineering, Chennai, Sriram College of Engineering, Chennai, Mar Baselios College of Engineering, Trivandrum.

I also worked at VIT, Pune & Presidency University, Bangalore. I have approximately 25 years of teaching experience at various Institutes / Engineering Institutions in different parts of the country.

My research interest includes Machine Learning, Natural Language Processing and Analysis of Algorithms.

Evaluation Scheme



Sl. No.	Subject Codes	Subject Name	Periods			Evaluation Scheme				End Semester		Total	Credit
			L	T	P	CT	TA	TOTAL	PS	TE	PE		
WEEKS COMPULSORY INDUCTION PROGRAM													
1	AAS0301A	Engineering Mathematics-III	3	1	0	30	20	50		100		150	4
2	ACSE0306	Discrete Structures	3	0	0	30	20	50		100		150	3
3	ACSE0304	Digital Logic & Circuit Design	3	0	0	30	20	50		100		150	3
4	ACSE0301	Data Structures	3	1	0	30	20	50		100		150	4
5	ACSE0302	Object Oriented Techniques using Java	3	0	0	30	20	50		100		150	3
6	ACSE0305	Computer Organization & Architecture	3	0	0	30	20	50		100		150	3
7	ACSE0354	Digital Logic & Circuit Design Lab	0	0	2				25		25	50	1
8	ACSE0351	Data Structures Lab	0	0	2				25		25	50	1
9	ACSE0352	Object Oriented Techniques using Java Lab	0	0	2				25		25	50	1
10	ACSE0359	Internship Assessment-I	0	0	2				50			50	1
11	ANC0301/ ANC0302	Cyber Security*/ Environmental Science*(Non Credit)	2	0	0	30	20	50		50		100	0
12		MOOCs** (For B.Tech. Hons. Degree)											
		GRAND TOTAL										1100	24

Syllabus

UNIT-I	**Introduction**	8 Hours
Object Oriented Programming: Introduction and Features: Abstraction, Encapsulation, Polymorphism, and Inheritance. Modeling Concepts: Introduction, Class Diagram and Object Diagram. Control Statements: Decision Making, Looping and Branching, Argument Passing Mechanism: Command Line Argument		
UNIT-II	**Basics of Java Programming**	8 Hours
Class and Object: Object Reference, Constructor, Abstract Class, Interface and its uses, Defining Methods, Use of “this” and “super” keyword, Garbage Collection and finalize () Method. Inheritance: Introduction and Types of Inheritance in Java, Constructors in Inheritance. Polymorphism: Introduction and Types, Overloading and Overriding. Lambda expression: Introduction and Working with Lambda Variables. Arrays: Introduction and its Types.		

Syllabus

UNIT-III	Packages, Exception Handling and String Handling	8 hours
Packages: Introduction and Types, Access Protection in Packages, Import and Execution of Packages. Exception Handling, Assertions and Localizations: Introduction and Types, Exceptions vs. Errors, Handling of Exception. Finally, Throws and Throw keyword, Multiple Catch Block, Nested Try and Finally Block. Assertions and Localizations Concepts and its working. String Handling: Introduction and Types, Operations, Immutable String, Method of String class, String Buffer and String Builder class		

UNIT-IV	Concurrency in Java and I/O Stream	8 hours
Threads: Introduction and Types, Creating Threads, Thread Life-Cycle, Thread Priorities, Daemon Thread, Runnable Class, Synchronizing Threads. I/O Stream: Introduction and Types, Common I/O Stream Operations, Interaction with I/O Streams Classes. Annotations: Introduction, Custom Annotations and Applying Annotations.		
UNIT-V	**GUI Programming, Generics and Collections**	8 hours
GUI Programming: Introduction and Types, Applet, Life Cycle of Applet, AWT, Components and Containers, Layout Managers and User-Defined Layout and Event Handling. Generics and Collections: Introduction, Using Method References, Using Wrapper Class, Using Lists, Sets, Maps and Queues, Working with Generics.		

Text Books

Text Books:
(1) Herbert Schildt,” Java - The Complete Reference”, McGraw Hill Education 12th edition
(2) Herbert Schildt,” Java: A Beginner’s Guide”, McGraw-Hill Education 2nd edition
(3) James Rumbaugh et. al, “Object Oriented Modeling and Design”, PHI 2nd Edition
Reference Books:
(4) Cay S. Horstmann, “Core Java Volume I – Fundamentals”, Prentice Hall
(5) Joshua Bloch,” Effective Java”, Addison Wesley
(6) E Balagurusamy, “Programming with Java A Primer”, TMH, 4th edition.

Branch-wise Applications

Java can be used :

- Data import and export.
- Cleaning data.
- Statistical analysis.
- Machine learning and Deep learning.
- Deep learning.
- Text analytics (also known as Natural Language Processing or NLP).
- Data visualization.

Course Objectives

- The objective of this course is to understand the object-oriented methodology and its techniques to design and develop conceptual models and demonstrate the standard concepts of object-oriented techniques modularity, I/O. and other standard language constructs.

- The basic objective of this course is to understand the fundamental concepts of object-oriented programming in Java language and also implement the Multithreading concepts, GUI based application and collection framework.

Course Outcomes

Course outcomes : After completion of this course students will be able to		
CO 1	Identify the concepts of object oriented programming and relationships among them needed in modeling.	K2
CO 2	Demonstrate the Java programs using OOP principles and also implement the concepts of lambda expressions.	K3
CO 3	Implement packages with different protection level resolving namespace collision and evaluate the error handling concepts for uninterrupted execution of Java program.	K3, K5
CO 4	Implement Concurrency control, I/O Streams and Annotations concepts by using Java program.	K3
CO 5	Design and develop the GUI based application, Generics and Collections in Java programming language to solve the real-world problem.	K6

Program Outcomes

1. Engineering knowledge:
2. Problem analysis:
3. Design/development of solutions:
4.Conduct investigations of complex problems
5. Modern tool usage
6. The engineer and society
7. Environment and sustainability
8. Ethics:
9. Individual and team work
10. Communication:
11. Project management and finance
12. Life-long learning

CO-PO Mapping

Mapping of Course Outcomes and Program Outcomes:

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO.1	**3**	**3**	**3**	**2**	**1**				**2**			**3**
CO.2	3	3	3	2	1				2			3
CO.3	3	3	3	2	1				2			3
CO.4	3	3	3	2	1				2			3
CO.5	3	3	3	2	1				2			3

Program Specific Outcomes

On successful completion of graduation degree the Engineering graduates will be able to:

PSO1: The ability to identify, analyze real world problems and design their ethical solutions using artificial intelligence, robotics, virtual/augmented reality, data analytics, block chain technology, and cloud computing.

PSO2: The ability to design and develop the hardware sensor devices and related interfacing software systems for solving complex engineering problems.

PSO3: The ability to understand inter disciplinary computing techniques and to apply them in the design of advanced computing.

PSO4: The ability to conduct investigation of complex problem with the help of technical, managerial, leadership qualities, and moder engineering tools provided by industry sponsored laboratories.

CO-PSO Mapping

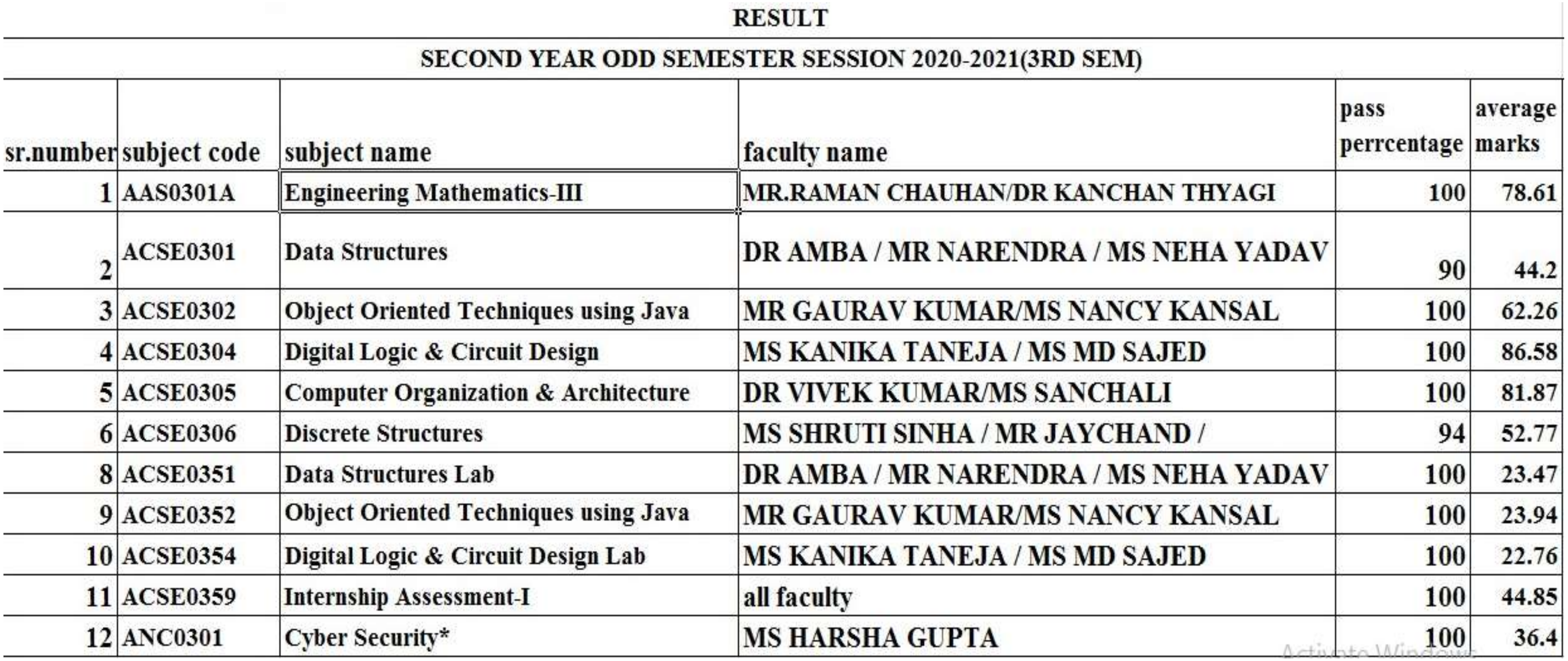
Mapping of Course Outcomes and Program Specific Outcomes:

	PSO1	PSO2	PSO3	PSO4
Co.1	**2**	**3**	**2**	**1**
CO.2	**2**	**3**	**2**	**1**
CO.3	**2**	**3**	**2**	**1**
CO.4	2	3	2	1
CO.5	**2**	**3**	**2**	**1**

Program Educational Objectives

PEO1: To have an excellent scientific and engineering breadth so as to comprehend, analyze, design and provide sustainable solutions for real-life problems using state-of-the-art technologies.
PEO2: To have a successful career in industries, to pursue higher studies or to support entrepreneurial endeavors and to face global challenges.
PEO3: To have an effective communication skills, professional attitude, ethical values and a desire to learn specific knowledge in emerging trends, technologies for research, innovation and product development and contribution to society.
PEO4: To have life-long learning for up-skilling and re-skilling for successful professional career as engineer, scientist, entrepreneur and bureaucrat for betterment of society

Result Analysis

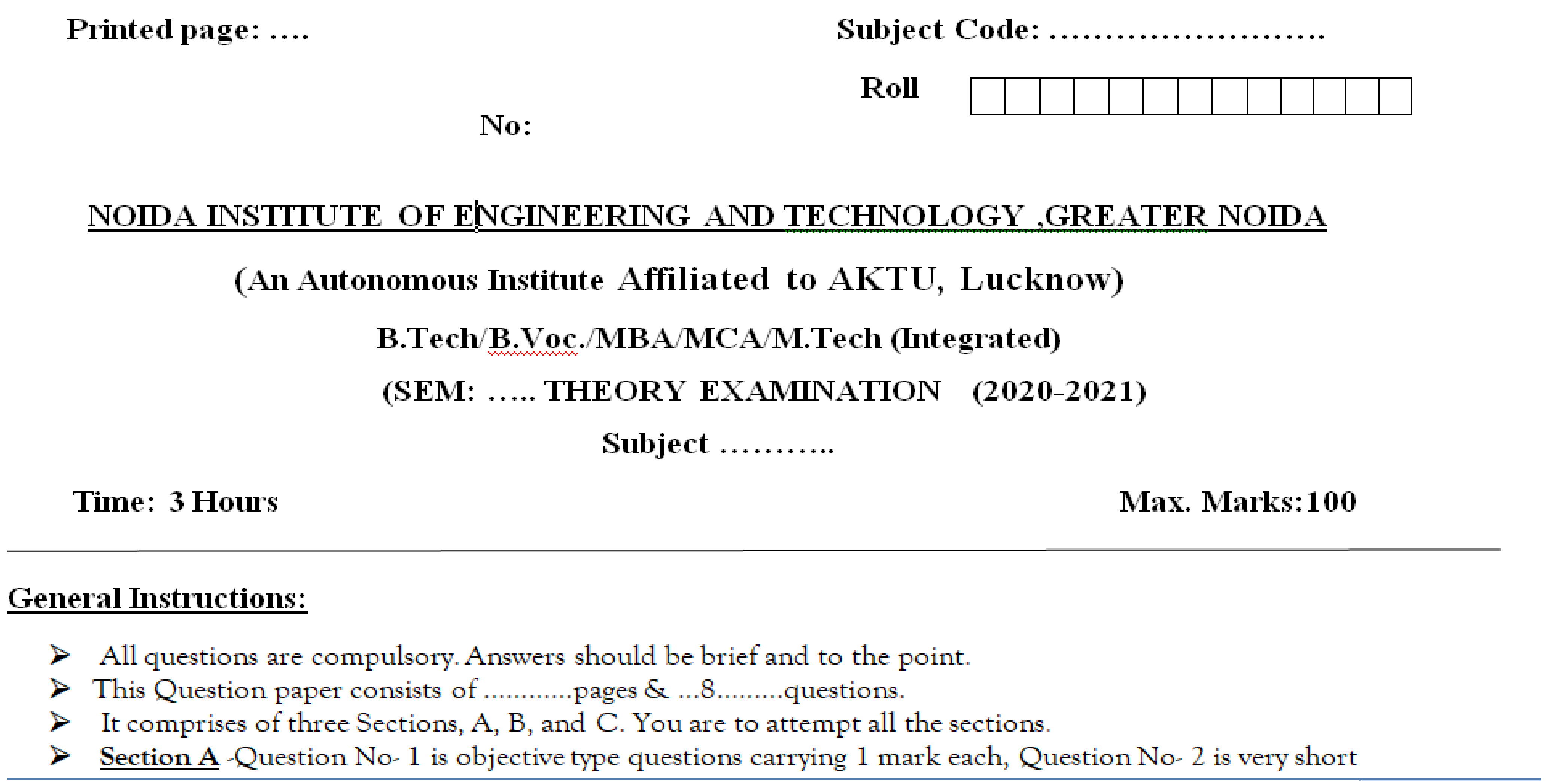


RESULT

SECOND YEAR ODD SEMESTER SESSION 2020-2021(3RD SEM)

sr.number	subject code	subject name	faculty name	pass perrcentage	average marks
1	AAS0301A	Engineering Mathematics-III	MR.RAMAN CHAUHAN/DR KANCHAN THYAGI	100	78.61
2	ACSE0301	Data Structures	DR AMBA / MR NARENDRA / MS NEHA YADAV	90	44.2
3	ACSE0302	Object Oriented Techniques using Java	MR GAURAV KUMAR/MS NANCY KANSAL	100	62.26
4	ACSE0304	Digital Logic & Circuit Design	MS KANIKA TANEJA / MS MD SAJED	100	86.58
5	ACSE0305	Computer Organization & Architecture	DR VIVEK KUMAR/MS SANCHALI	100	81.87
6	ACSE0306	Discrete Structures	MS SHRUTI SINHA / MR JAYCHAND /	94	52.77
8	ACSE0351	Data Structures Lab	DR AMBA / MR NARENDRA / MS NEHA YADAV	100	23.47
9	ACSE0352	Object Oriented Techniques using Java	MR GAURAV KUMAR/MS NANCY KANSAL	100	23.94
10	ACSE0354	Digital Logic & Circuit Design Lab	MS KANIKA TANEJA / MS MD SAJED	100	22.76
11	ACSE0359	Internship Assessment-I	all faculty	100	44.85
12	ANC0301	Cyber Security*	MS HARSHA GUPTA	100	36.4

End Semester Question Paper Templates (Offline Pattern/Online Pattern



Printed page: ....

Subject Code: ....

Roll No:

NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY ,GREATER NOIDA

(An Autonomous Institute Affiliated to AKTU, Lucknow)

B.Tech/B.Voc./MBA/MCA/M.Tech (Integrated)

(SEM: ..... THEORY EXAMINATION (2020-2021)

Subject ....

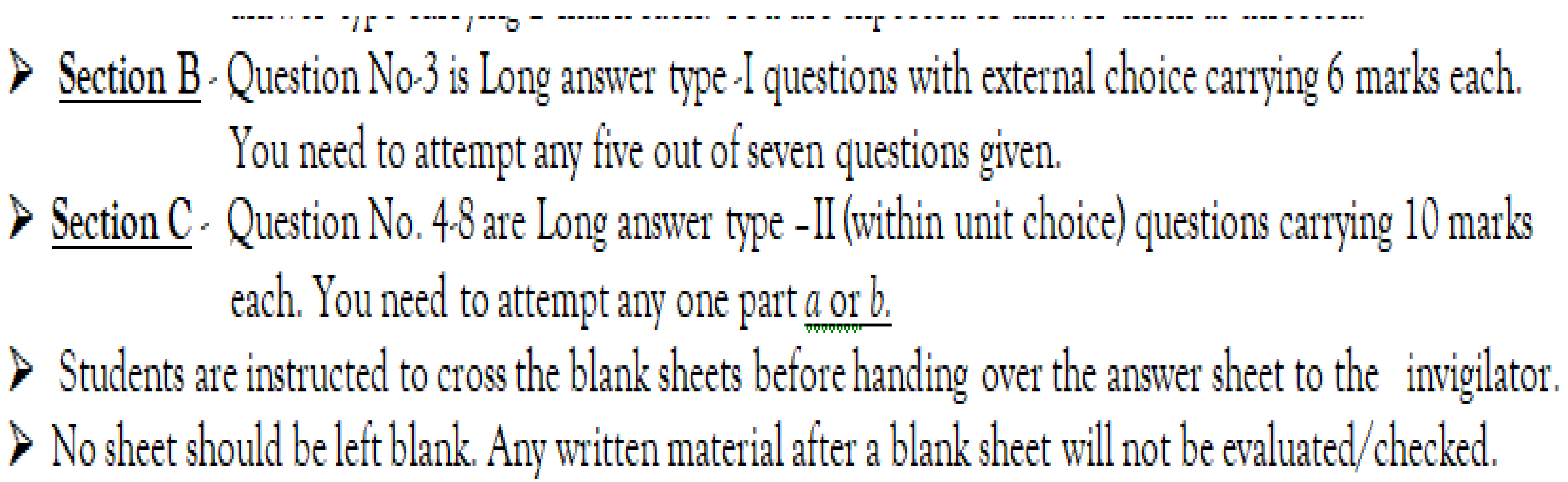
Time: 3 Hours

Max. Marks:100

General Instructions:

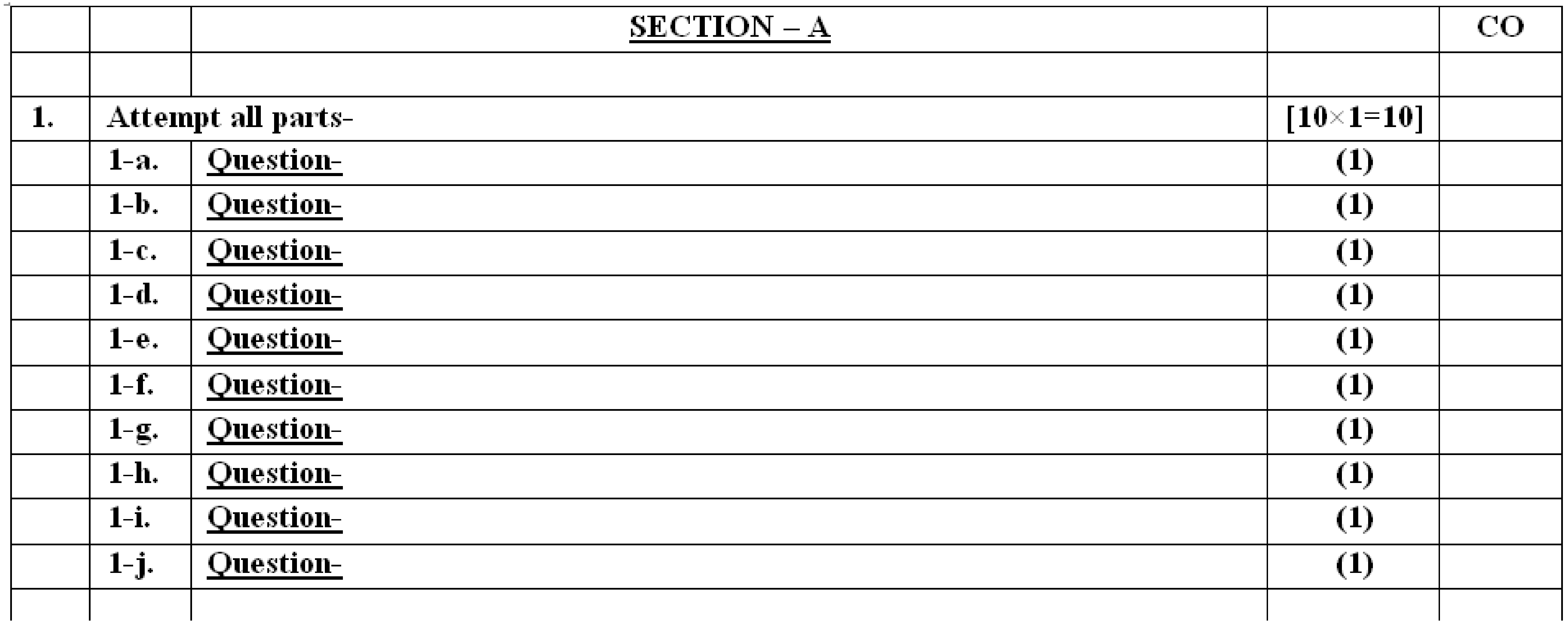
- All questions are compulsory. Answers should be brief and to the point.
- This Question paper consists of ....pages & ...8.........questions.
- It comprises of three Sections, A, B, and C. You are to attempt all the sections.
- Section A -Question No- 1 is objective type questions carrying 1 mark each, Question No- 2 is very short

End Semester Question Paper Templates (Offline Pattern/Online Pattern



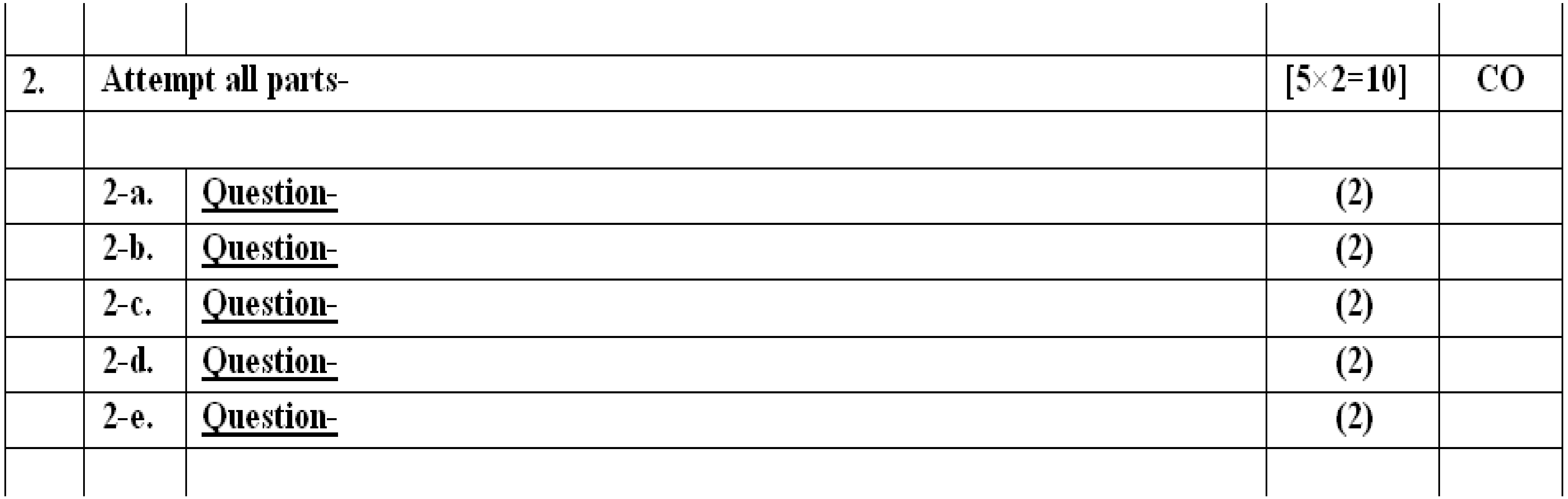
- **<u>Section B</u>** - Question No-3 is Long answer type -I questions with external choice carrying 6 marks each. You need to attempt any five out of seven questions given.
- **<u>Section C</u>** - Question No. 4-8 are Long answer type –II (within unit choice) questions carrying 10 marks each. You need to attempt any one part *a or b.*
- Students are instructed to cross the blank sheets before handing over the answer sheet to the invigilator.
- No sheet should be left blank. Any written material after a blank sheet will not be evaluated/checked.

End Semester Question Paper Templates (Offline Pattern/Online Pattern



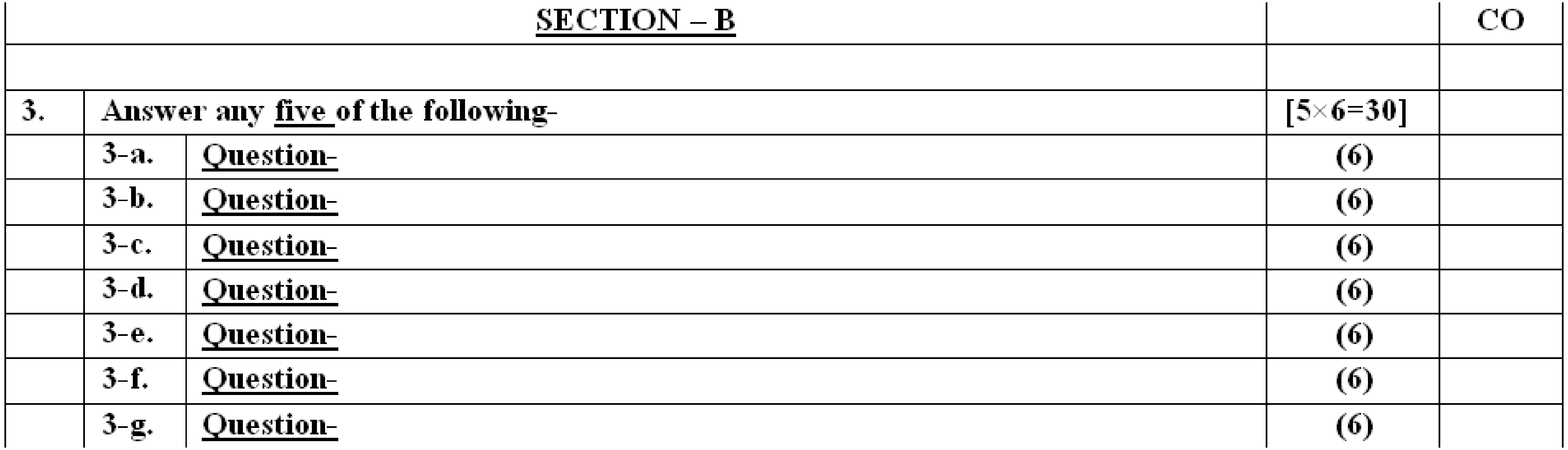
		SECTION – A		**CO**
1.	**Attempt all parts-**		**[10×1=10]**	
	1-a.	**Question-**	**(1)**	
	1-b.	**Question-**	**(1)**	
	1-c.	**Question-**	**(1)**	
	1-d.	**Question-**	**(1)**	
	1-e.	**Question-**	**(1)**	
	1-f.	**Question-**	**(1)**	
	1-g.	**Question-**	**(1)**	
	1-h.	**Question-**	**(1)**	
	1-i.	**Question-**	**(1)**	
	1-j.	**Question-**	**(1)**	

End Semester Question Paper Templates (Offline Pattern/Online Pattern



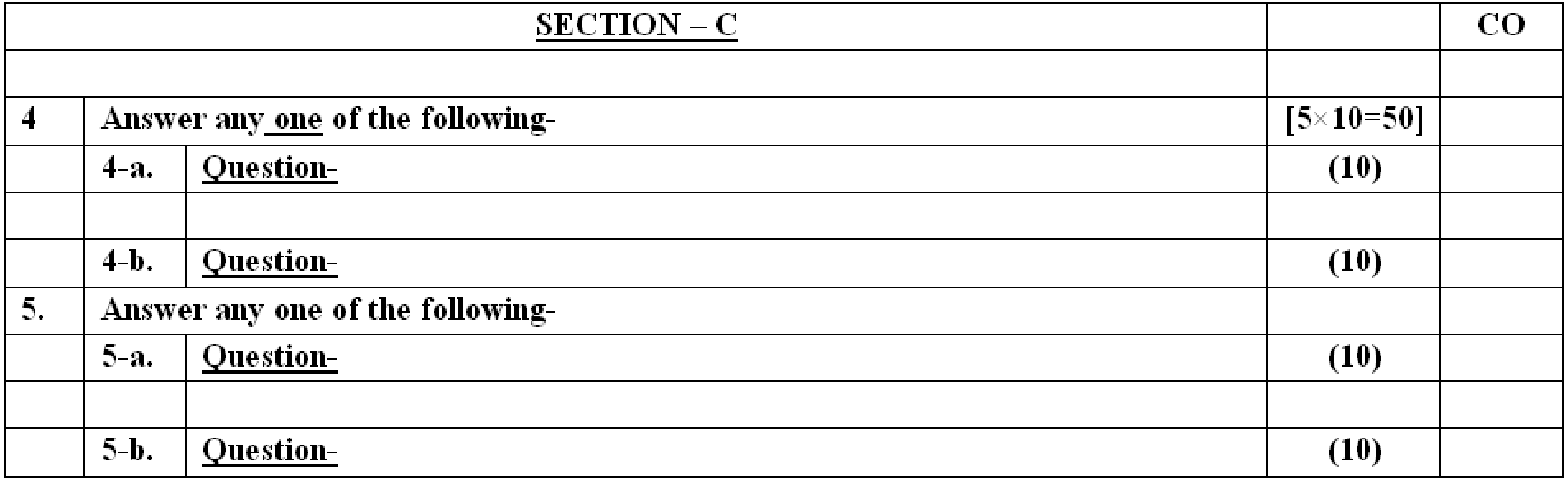
2.	Attempt all parts-		[5×2=10]	CO
	2-a.	<u>**Question-**</u>	(2)	
	2-b.	<u>**Question-**</u>	(2)	
	2-c.	<u>**Question-**</u>	(2)	
	2-d.	<u>**Question-**</u>	(2)	
	2-e.	<u>**Question-**</u>	(2)	

End Semester Question Paper Templates (Offline Pattern/Online Pattern



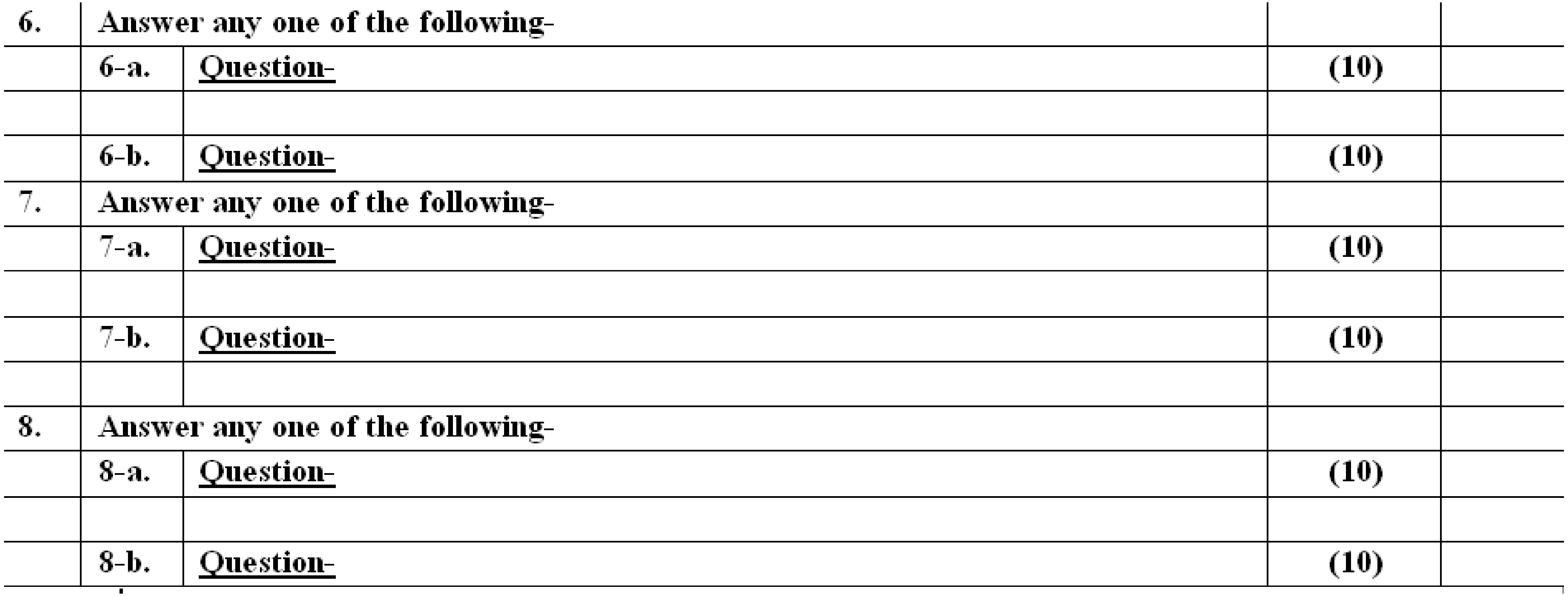
SECTION – B				CO
3.	Answer any five of the following-		[5×6=30]	
	3-a.	Question-	(6)	
	3-b.	Question-	(6)	
	3-c.	Question-	(6)	
	3-d.	Question-	(6)	
	3-e.	Question-	(6)	
	3-f.	Question-	(6)	
	3-g.	Question-	(6)	

End Semester Question Paper Templates (Offline Pattern/Online Pattern



SECTION – C				CO
4	Answer any one of the following-		[5×10=50]	
	4-a.	Question-	(10)	
	4-b.	Question-	(10)	
5.	Answer any one of the following-			
	5-a.	Question-	(10)	
	5-b.	Question-	(10)	

End Semester Question Paper Templates (Offline Pattern/Online Pattern



6.	**Answer any one of the following-**			
	6-a.	**<u>Question-</u>**	**(10)**	
	6-b.	**<u>Question-</u>**	**(10)**	
7.	**Answer any one of the following-**			
	7-a.	**<u>Question-</u>**	**(10)**	
	7-b.	**<u>Question-</u>**	**(10)**	
8.	**Answer any one of the following-**			
	8-a.	**<u>Question-</u>**	**(10)**	
	8-b.	**<u>Question-</u>**	**(10)**	

Prerequisite/Recap

- Student must know at least the basics of how to use a computer, and should be able to start a command line shell.
- Knowledge of basic programming concepts, as covered in ‘Programming Basic' course is necessary.
- Students must have basic understanding of computer programming and related programming paradigms

NIET Greater Noida GET FUTURE READY Estd 2001 AN AUTONOMOUS INSTITUTE

About the Subject with videos

- OOTS refers to languages that uses objects in programming.
- OOTS aims to implement real-world entities like inheritance, hiding, polymorphism etc in programming.
- The main aim of OOTS is to bind together the data and the functions that operate on them so that no other part of the code can access this data except that function

Link:	
Unit 1	https://www.youtube.com/watch?v=r59xYe3Vyks&list=PLS1QulWo1RIbfTjQvTdj8Y6yyq4R7g-Al
Unit 2	https://www.youtube.com/watch?v=ZHLdVRXIuC8&list=PLS1QulWo1RIbfTjQvTdj8Y6yyq4R7g-Al&index=18,
Unit 3	https://www.youtube.com/watch?v=hBh_CC5y8-s https://www.youtube.com/watch?v=OjdT2l-EZJA
Unit 4	https://www.youtube.com/watch?v=qQVqfvs3p48
Unit 5	• https://www.youtube.com/watch?v=OjdT2l-EZJA

UNIT CONTENT

- GUI Programming
- Introduction and Types, Applet
- AWT
- Components and Containers
- Layout Managers and User-defined layout
- Event Handling
- Generics and Collections.
- Using method references
- Using Wrapper classes and Using Lists
- Sets, Maps and Queues

Unit Objectives

- The Objective of this unit is to learn about GUI Programming. And Event Handling.
- The Objective of this unit is to understand the concept of Generics and Collections in java.

Topic Objectives

After you have read and studied this topic, you should be able to

- **develop applets for web applications and to design GUI based applications**

Topic mapping with CO

Topic	CO
GUI Programming	CO5
Introduction and Types, Applet	CO5
AWT	CO5
Components and Containers	CO5
Layout Managers and User-defined layout	CO5
Event Handling	CO5
Generics and Collections.	CO5
Using method references	CO5
Using Wrapper classes and Using Lists	CO5
Sets, Maps and Queues	CO5

Lecture 1

- GUI Programming
- Introduction and Types, Applet

Why learn GUIs?

- Learn about *event-driven programming* techniques
- Practice learning and using a large, complex API
- A chance to see how it is designed and learn from it:
  - model-view separation
  - design patterns
  - refactoring vs. re implementing an ailing API
- Because GUIs are neat!
- *Caution*: There is way more here than you can memorize.
  - Part of learning a large API is "letting go."
  - You won't memorize it all; you will look things up as you need them.
  - But you can learn the fundamental concepts and general ideas.

Java GUI History

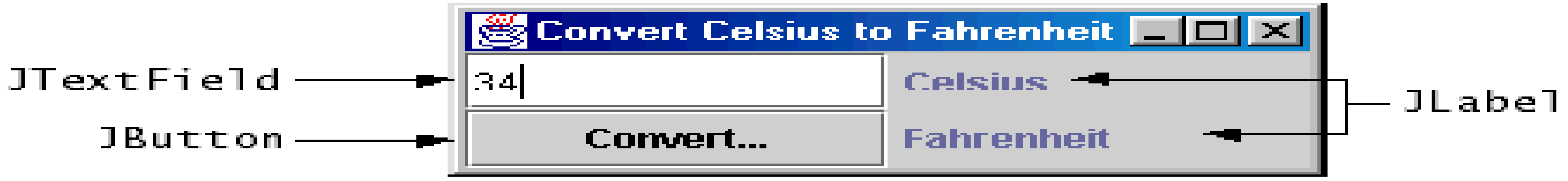
- **Abstract Windowing Toolkit** (**AWT**): Sun's initial effort to create a set of cross-platform GUI classes. *(JDK 1.0 - 1.1)*
  - Maps general Java code to each operating system's real GUI system.
  - *Problems:* Limited to lowest common denominator; clunky to use.

- **Swing**: A newer GUI library written from the ground up that allows much more powerful graphics and GUI construction. *(JDK 1.2+)*
  - Paints GUI controls itself pixel-by-pixel rather than handing off to OS.
  - *Benefits:* Features; compatibility; OO design.
  - *Problem:* Both exist in Java now; easy to get them mixed up; still have to use both in various places.

GUI terminology

- **window**: A first-class citizen of the graphical desktop.
  - Also called a *top-level container*.
  - examples: frame, dialog box, applet
- **component**: A GUI widget that resides in a window.
  - Also called *controls* in many other languages.
  - examples: button, text box, label

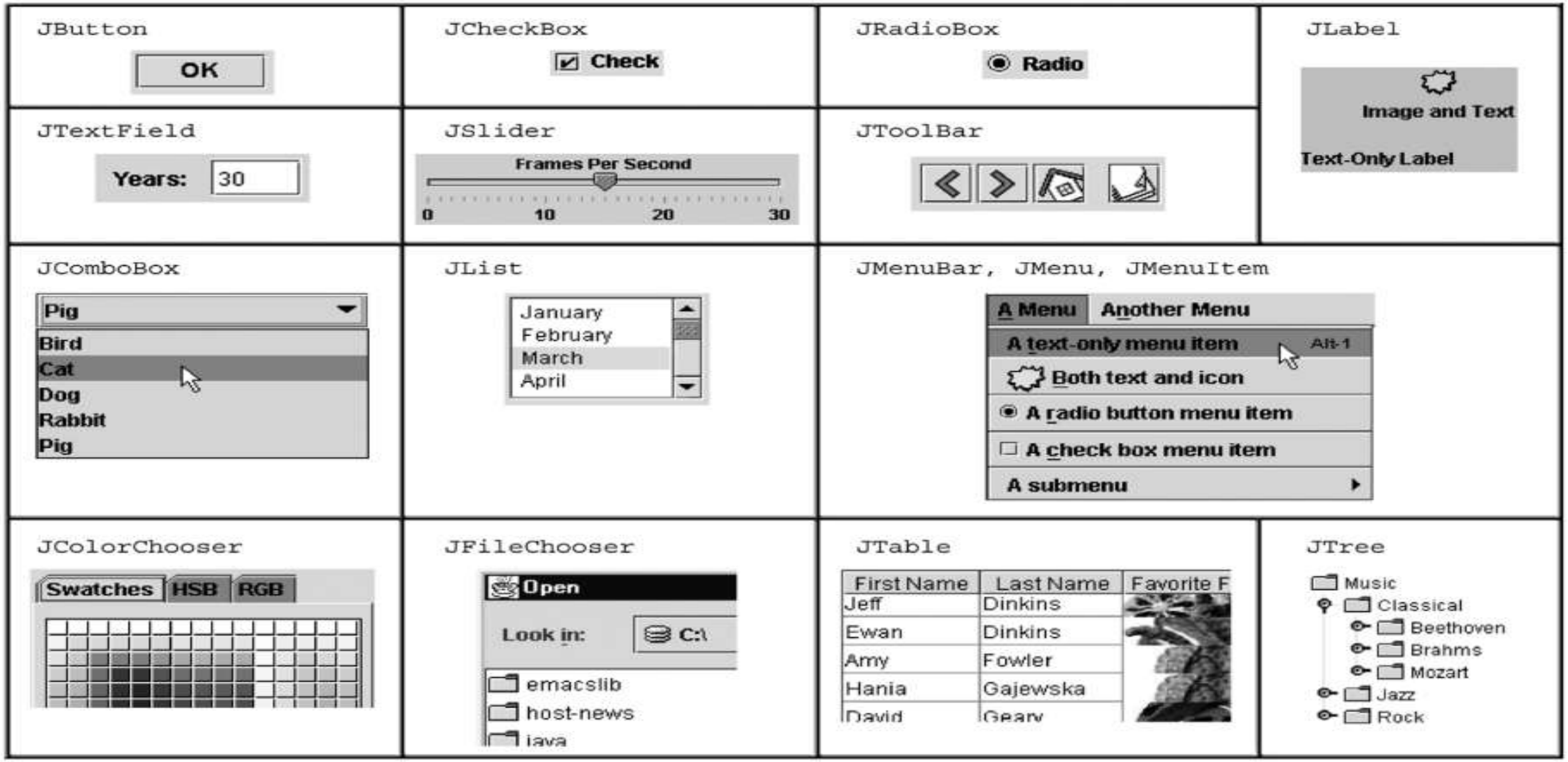
- **container**: A logical grouping for storing components.
  - examples: panel, box



Lecture 2

- AWT
- Components and Containers
- Layout Managers and User-defined layout

Components


JButton
OK
JCheckBox
Check
JRadioBox
Radio
JLabel
Image and Text
Text-Only Label
JTextField
Years:
30
JSlider
Frames Per Second
0
10
20
30
JToolBar
JComboBox
Pig
Bird
Cat
Dog
Rabbit
Pig
JList
January
February
March
April
JMenuBar, JMenu, JMenuItem
A Menu
Another Menu
A text-only menu item
Alt-1
Both text and icon
A radio button menu item
A check box menu item
A submenu
JColorChooser
Swatches
HSB
RGB
JFileChooser
Open
Look in:
C:\
emacslib
host-news
java
JTable
First Name
Last Name
Favorite F
Jeff
Dinkins
Ewan
Dinkins
Amy
Fowler
Hania
Gajewska
David
Geary
JTree
Music
Classical
Beethoven
Brahms
Mozart
Jazz
Rock

Swing inheritance hierarchy

- Component (AWT)
  - Window
    - Frame
      - **JFrame** (Swing)
      - **JDialog**
  - Container
    - JComponent (Swing)

JButton	**JColorChooser**	**JFileChooser**
JComboBox	**JLabel**	**JList**
JMenuBar	**JOptionPane**	**JPanel**
JPopupMenu	**JProgressBar**	**JScrollbar**
JScrollPane	**JSlider**	**JSpinner**
JSplitPane	**JTabbedPane**	**JTable**
JToolbar	**JTree**	**JTextArea**
JTextField	...	

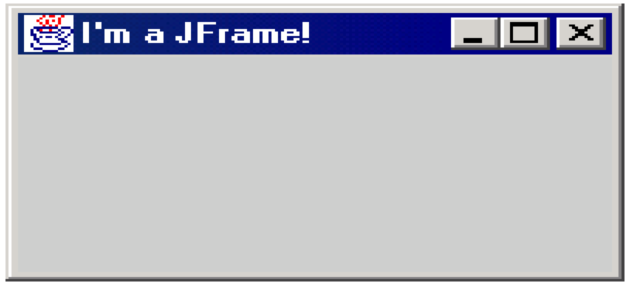
Component Properties

– Each has a get (or is) accessor and a set modifier method.
– examples: getColor, setFont, setEnabled, isVisible

name	type	description
background	`Color`	background color behind component
border	`Border`	border line around component
enabled	`boolean`	whether it can be interacted with
focusable	`boolean`	whether key text can be typed on it
font	`Font`	font used for text in component
foreground	`Color`	foreground color of component
height, width	`int`	component's current size in pixels
visible	`boolean`	whether component can be seen
tooltip text	`String`	text shown when hovering mouse
size, minimum / maximum / preferred size	`Dimension`	various sizes, size limits, or desired sizes that the component may take

Jframe [CO5]

a graphical window to hold other components

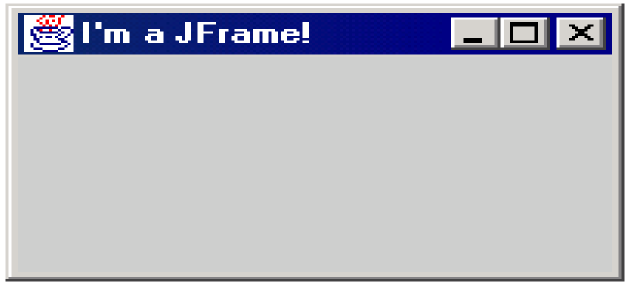


- public JFrame()
  public JFrame(String title)
  Creates a frame with an optional title.

  – Call setVisible(true) to make a frame appear on the screen after creating it.

- public void add(Component comp)
  Places the given component or container inside the frame.

More JFrame



- public void setDefaultCloseOperation(int op)
  Makes the frame perform the given action when it closes.
  - Common value passed: JFrame.EXIT_ON_CLOSE
  - If not set, the program will never exit even if the frame is closed.

- public void setSize(int width, int height)
  Gives the frame a fixed size in pixels.

- public void pack()
  Resizes the frame to fit the components inside it snugly.

JButton

a clickable region for causing actions to occur

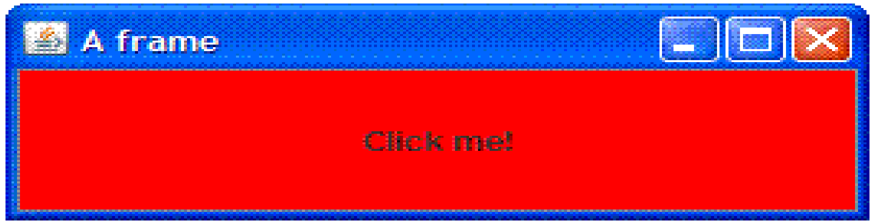
- public JButton(String text)
  Creates a new button with the given string as its text.

- public String getText()
  Returns the text showing on the button.

- public void setText(String text)
  Sets button's text to be the given string.

GUI example

```
import java.awt.*;          // Where is the other button?
import javax.swing.*;

public class GuiExample1 {
    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(new Dimension(300, 100));
        frame.setTitle("A frame");

        JButton button1 = new JButton();
        button1.setText("I'm a button.");
        button1.setBackground(Color.BLUE);
        frame.add(button1);

        JButton button2 = new JButton();
        button2.setText("Click me!");
        button2.setBackground(Color.RED);
        frame.add(button2);

        frame.setVisible(true);
    }
}
```



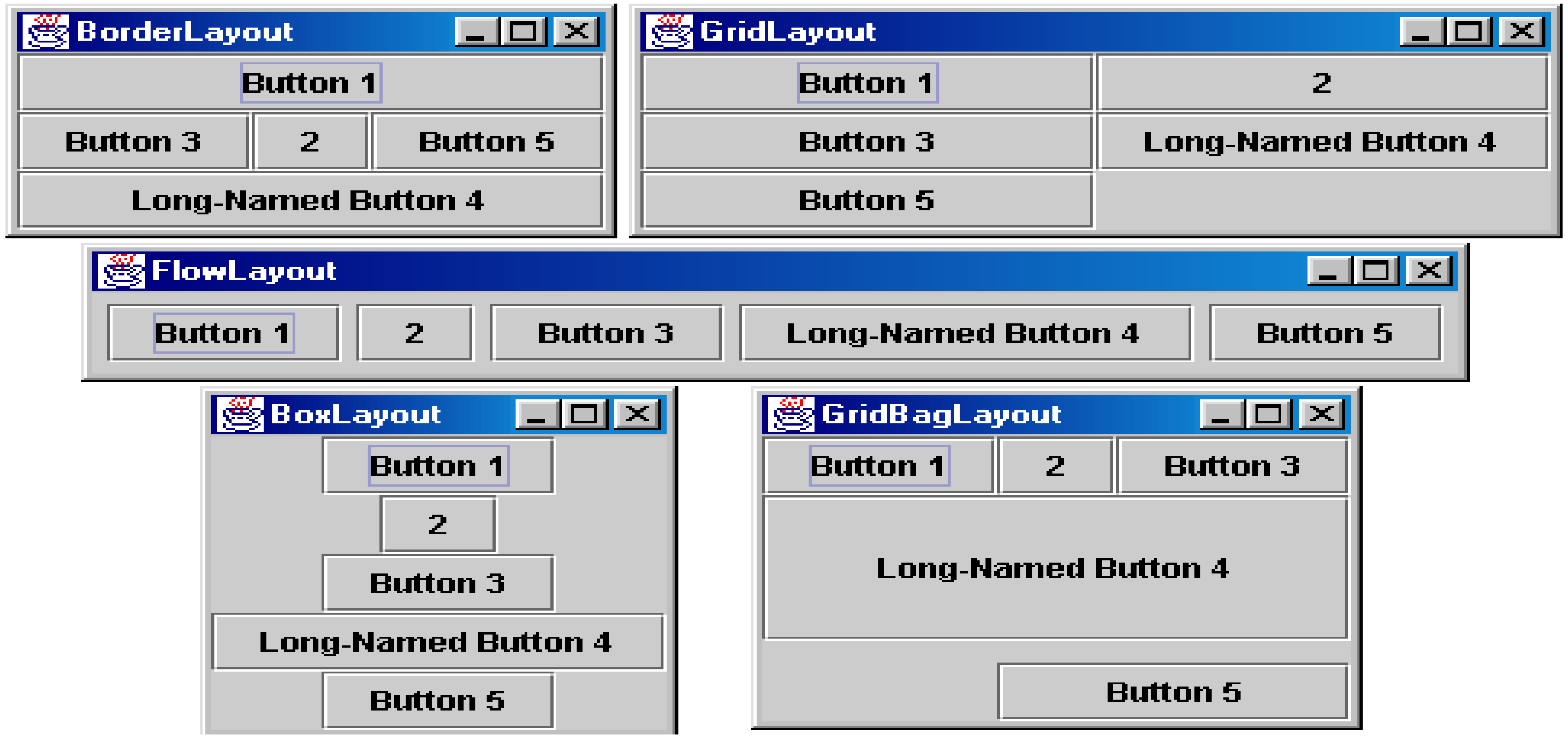
Sizing and positioning

How does the programmer specify where each component appears, how big each component should be, and what the component should do if the window is resized / moved / maximized / etc.?

- **Absolute positioning** (C++, C#, others):
  Programmer specifies exact pixel coordinates of every component.
  - "Put this button at (x=15, y=75) and make it 70x31 px in size."
- **Layout managers** (Java):
  Objects that decide where to position each component based on some general rules or criteria.
  - "Put these four buttons into a 2x2 grid and put these text boxes in a horizontal flow in the south part of the frame."

Containers and layout

- Place components in a *container*; add the container to a frame.
  - **container**: An object that stores components and governs their positions, sizes, and resizing behavior.



JFrame as container

A JFrame is a container. Containers have these methods:

- public void **add**(Component comp)
  public void **add**(Component comp, Object info)
  Adds a component to the container, possibly giving extra information about where to place it.

- public void **remove**(Component comp)

- public void **setLayout**(LayoutManager mgr)
  Uses the given layout manager to position components.

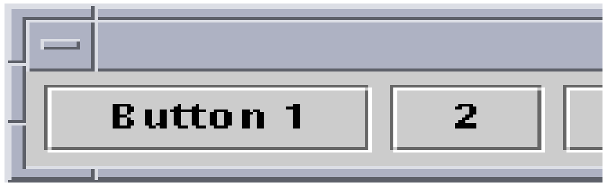
- public void **validate**()
  Refreshes the layout (if it changes after the container is onscreen).

Preferred sizes

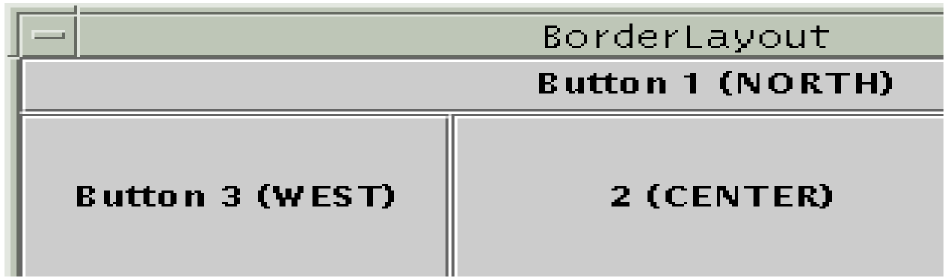
- Swing component objects each have a certain size they would "like" to be: Just large enough to fit their contents (text, icons, etc.).
  – This is called the *preferred size* of the component.

  – Some types of layout managers (e.g. FlowLayout) choose to size the components inside them to the preferred size.
  – Others (e.g. BorderLayout, GridLayout) disregard the preferred size and use some other scheme to size the components.

Buttons at preferred size:



Not preferred size:



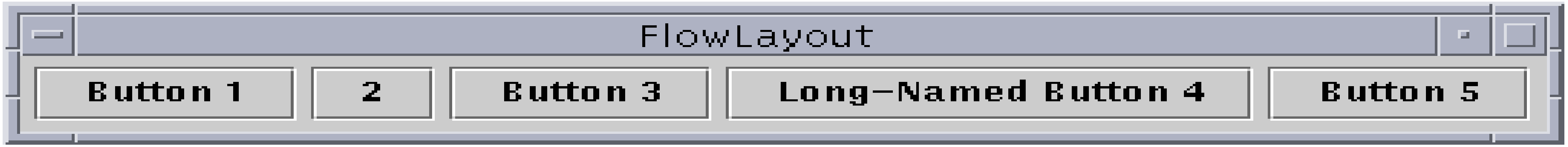
FlowLayout [CO5]

public FlowLayout()

- treats container as a left-to-right, top-to-bottom "paragraph".
  - Components are given preferred size, horizontally and vertically.
  - Components are positioned in the order added.
  - If too long, components wrap around to the next line.

myFrame.setLayout(**new FlowLayout()**);

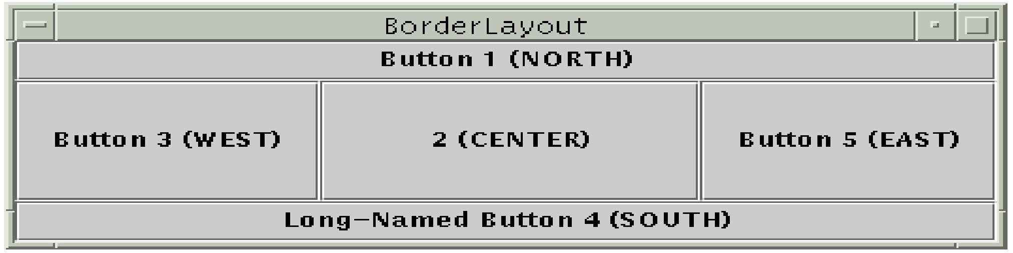
myFrame.add(new JButton("Button 1"));



- The default layout for containers other than JFrame (seen later).

BorderLayout

public BorderLayout()



- Divides container into five regions:
  - NORTH and SOUTH regions expand to fill region horizontally, and use the component's preferred size vertically.
  - WEST and EAST regions expand to fill region vertically, and use the component's preferred size horizontally.
  - CENTER uses all space not occupied by others.

myFrame.setLayout(**new BorderLayout()**);

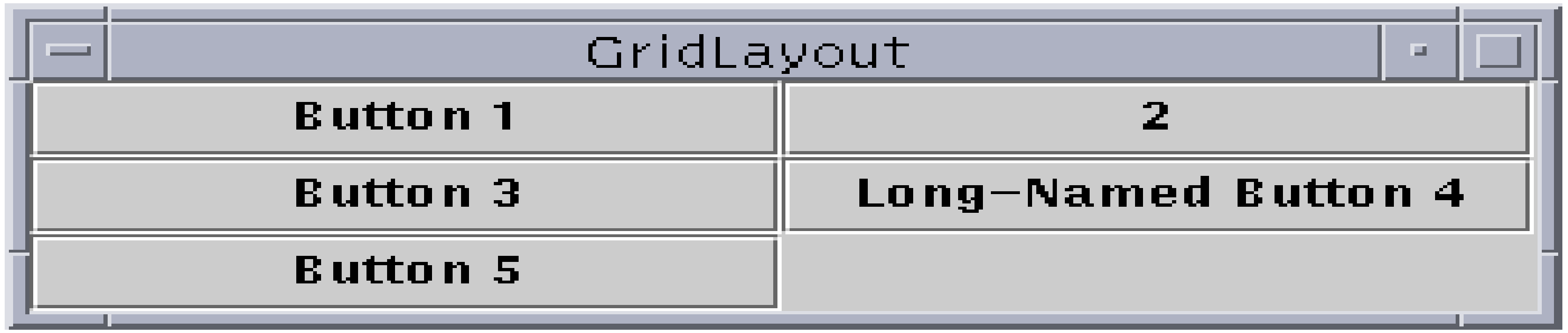
myFrame.add(new JButton("Button 1"), **BorderLayout.NORTH**);

- This is the default layout for a JFrame.

GridLayout

public GridLayout(int rows, int columns)

- Treats container as a grid of equally-sized rows and columns.
- Components are given equal horizontal / vertical size, disregarding preferred size.
- Can specify 0 rows or columns to indicate expansion in that direction as needed.



Lecture 3

- Event Handling

Event Listeners

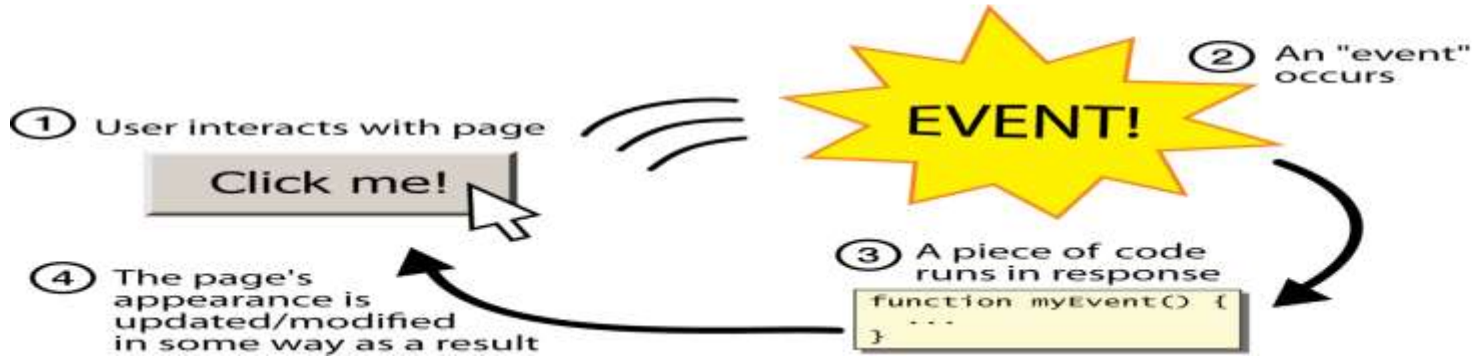
Topic Objective

Student will be able to create event listeners and handlers for User Interface components.

Graphical events

- **event**: An object that represents a user's interaction with a GUI component; can be "handled" to create interactive components.

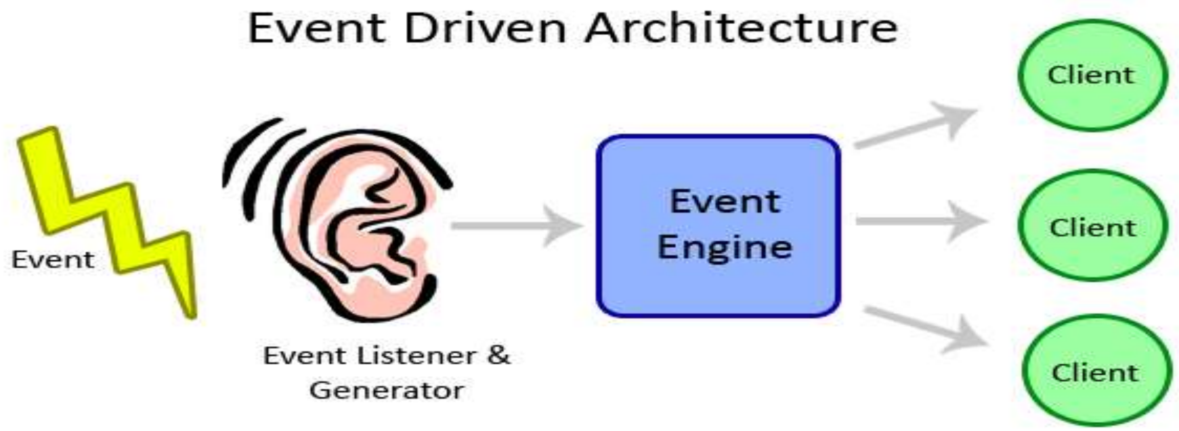
- **listener**: An object that waits for events and responds to them.
  - To handle an event, attach a *listener* to a component.
  - The listener will be notified when the event occurs (e.g. button click).



1 User interacts with page
Click me!
EVENT!
2 An "event" occurs
3 A piece of code runs in response
function myEvent() { ... }
4 The page's appearance is updated/modified in some way as a result

Event-driven programming

- **event-driven programming**: A style of coding where a program's overall flow of execution is dictated by events.
  - Rather than a central "main" method that drives execution, the program loads and waits for user input events.
  - As each event occurs, the program runs particular code to respond.
  - The overall flow of what code is executed is determined by the series of events that occur, not a pre-determined order.



Event hierarchy [CO5]

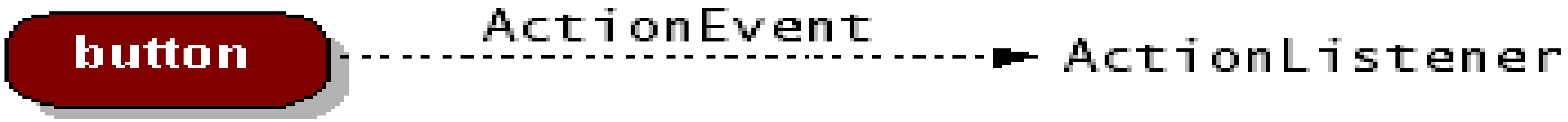
- EventObject
  - AWTEvent (AWT)
    - **ActionEvent**
    - TextEvent
    - ComponentEvent
      - FocusEvent
      - WindowEvent
      - InputEvent
        - KeyEvent
        - MouseEvent

- EventListener
  - AWTEventListener
  - **ActionListener**
  - TextListener
  - ComponentListener
  - FocusListener
  - WindowListener
  - KeyListener
  - MouseListener

Action events

- **action event**: An action that has occurred on a GUI component.
  - The most common, general event type in Swing. Caused by:
    - button or menu clicks,
    - check box checking / unchecking,
    - pressing Enter in a text field, ...

  - Represented by a class named ActionEvent
  - Handled by objects that implement interface ActionListener



Implementing a listener

```
public class **name** implements ActionListener {
    public void actionPerformed(ActionEvent event) {
        **code to handle the event**;
    }
}
```

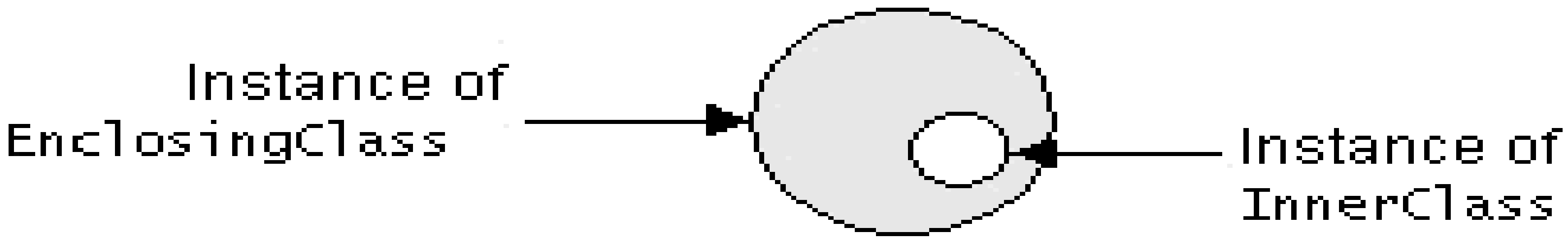
- JButton and other graphical components have this method:
  - public void addActionListener(ActionListener al)
    Attaches the given listener to be notified of clicks and events that occur on this component.

Lecture 4

- Generics and Collections.
- Using method references
- Using Wrapper classes and Using Lists
- Sets, Maps and Queues

Nested classes [CO5]

- **nested class**: A class defined inside of another class.
- Usefulness:
  - Nested classes are hidden from other classes (encapsulated).
  - Nested objects can access/modify the fields of their outer object.
- Event listeners are often defined as nested classes inside a GUI.



Nested class syntax

```
// enclosing outer class
public class name {
    ...

    // nested inner class
    private class name {
        ...
    }
}
```

- Only the outer class can see the nested class or make objects of it.
- Each nested object is associated with the outer object that created it, so it can access/modify that outer object's methods/fields.
  - If necessary, can refer to outer object as **OuterClassName**.this

Static inner classes

```
// enclosing outer class
public class name {
    ...

    // non-nested static inner class
    public static class name {
        ...
    }
}
```

- Static inner classes are *not* associated with a particular outer object.
- They cannot see the fields of the enclosing class.
- *Usefulness:* Clients can refer to and instantiate static inner classes:

  Outer.**Inner name** = new **Outer**.**Inner**(**params**);

Java 2 Collections

- A collection is an object that groups multiple elements into a single unit
- Very useful
  - store, retrieve and manipulate data
  - transmit data from one method to another
  - data structures and methods written by hotshots in the field
    - Joshua Bloch, who also wrote the Collections tutorial

Collections Framework

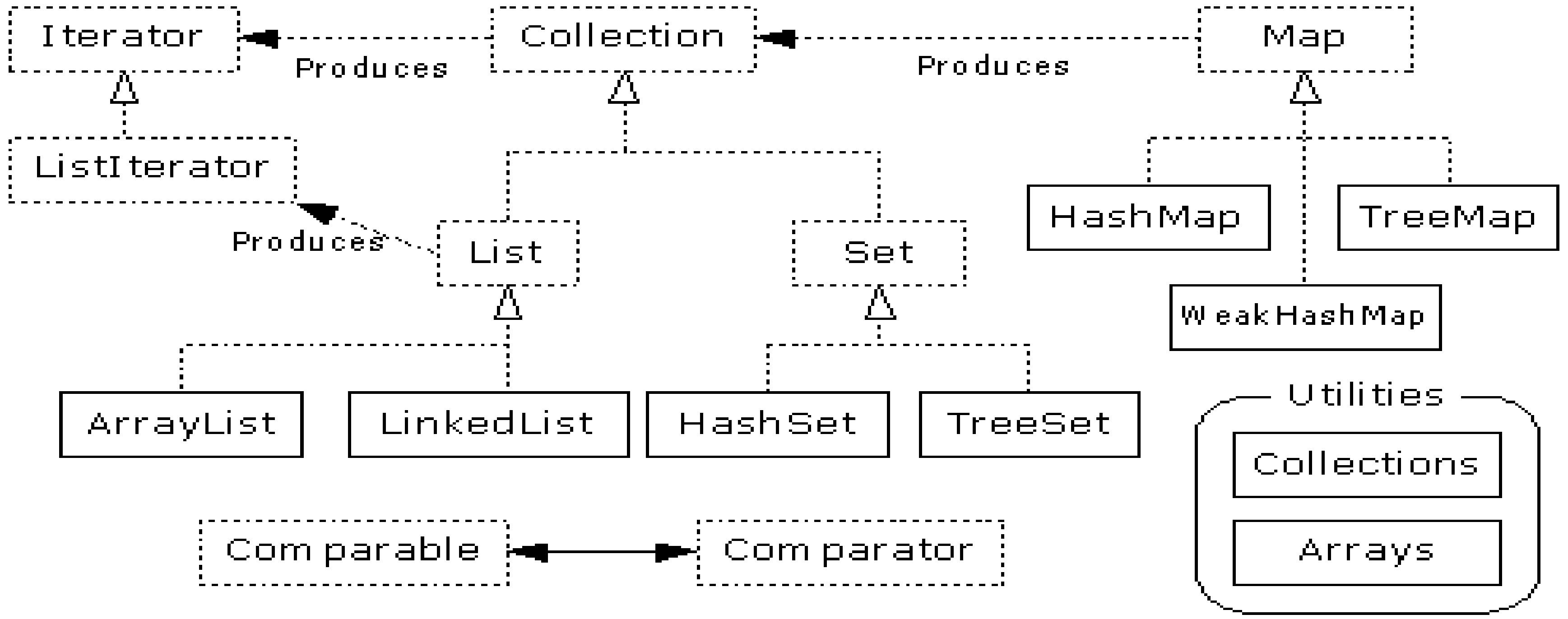
- Unified architecture for representing and manipulating collections.
- A collections framework contains three things

  » Interfaces

  » Implementations

  » Algorithms

Collections Framework Diagram



•Interfaces, Implementations, and Algorithms

Collection Interface

- Defines fundamental methods
  - » **int size();**
  - » **boolean isEmpty();**
  - » **boolean contains(Object element);**
  - » **boolean add(Object element);   // Optional**
  - » **boolean remove(Object element); // Optional**
  - » **Iterator iterator();**
- These methods are enough to define the basic behavior of a collection
- Provides an Iterator to step through the elements in the Collection

Iterator Interface

- Defines three fundamental methods
  - » `Object next()`
  - » `boolean hasNext()`
  - » `void remove()`
- These three methods provide access to the contents of the collection
- An Iterator knows position within collection
- Each call to next() “reads” an element from the collection
  - » Then you can use it or remove it

Iterator Position

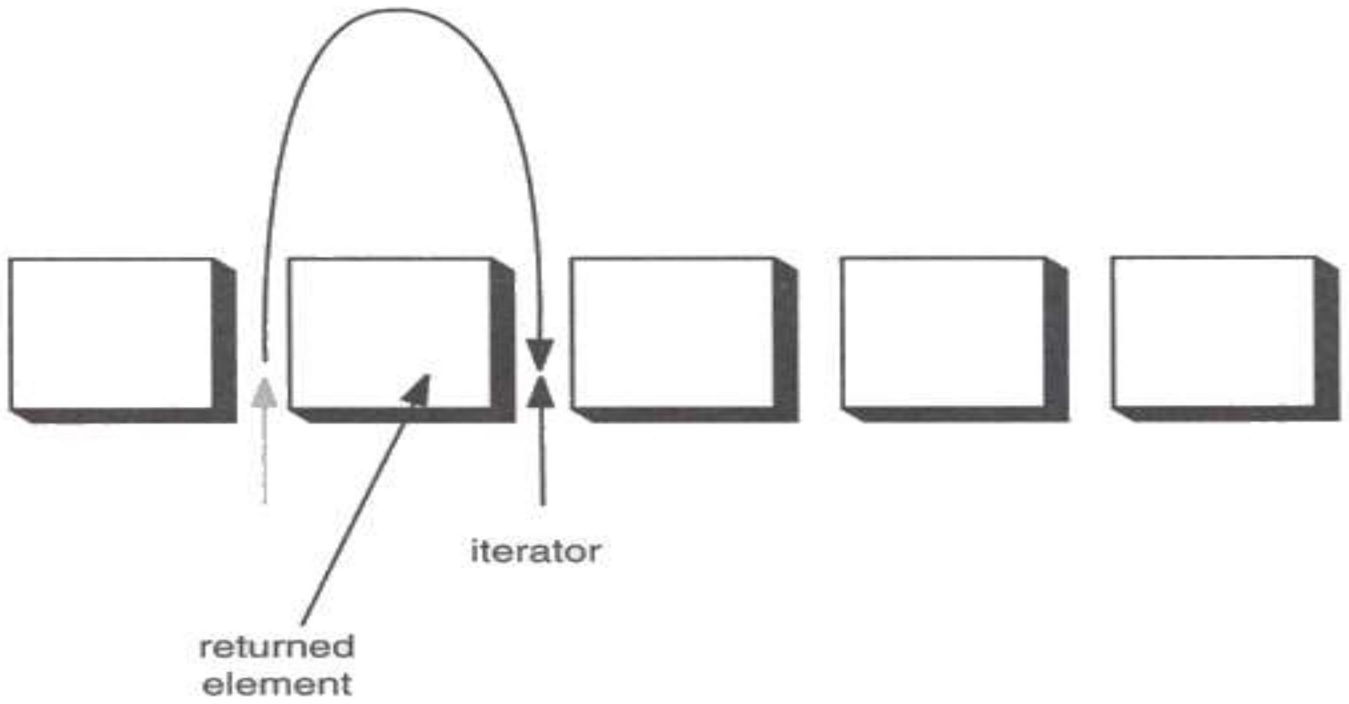
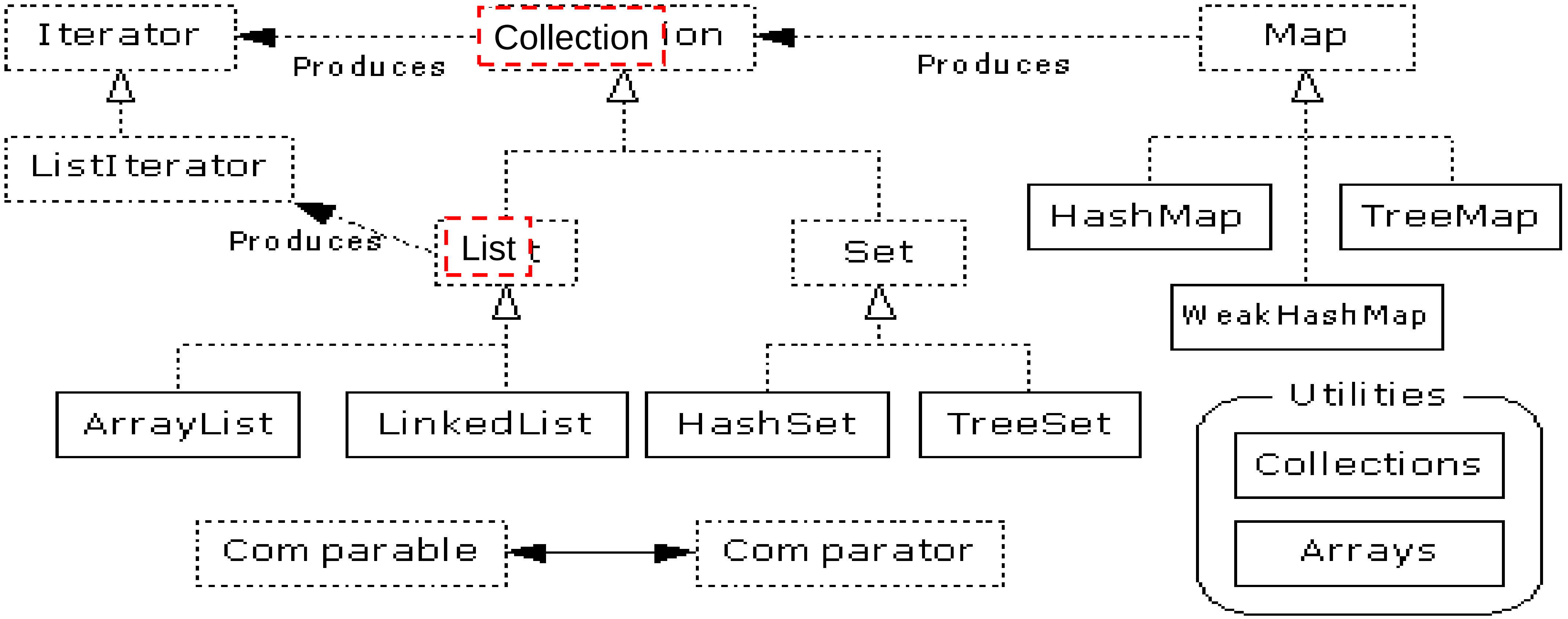


Figure 2–3: Advancing an iterator

Example - SimpleCollection

```
public class SimpleCollection  {
   public static void main(String[] args) {
    Collection c;
    c = new ArrayList();
    System.out.println(c.getClass().getName());
    for (int i=1; i <= 10; i++) {
           c.add(i + " * " + i + " = "+i*i);
    }
    Iterator iter = c.iterator();
    while (iter.hasNext())
           System.out.println(iter.next());
   }
}
```

List Interface Context



List Interface

- The List interface adds the notion of *order* to a collection
- The user of a list has control over where an element is added in the collection
- Lists typically allow *duplicate* elements
- Provides a ListIterator to step through the elements in the list.

ListIterator Interface

- Extends the Iterator interface
- Defines three fundamental methods
  - » **void add(Object o)** - before current position
  - » **boolean hasPrevious()**
  - » **Object previous()**
- The addition of these three methods defines the basic behavior of an ordered list
- A ListIterator knows position within list

Iterator Position - next(), previous()

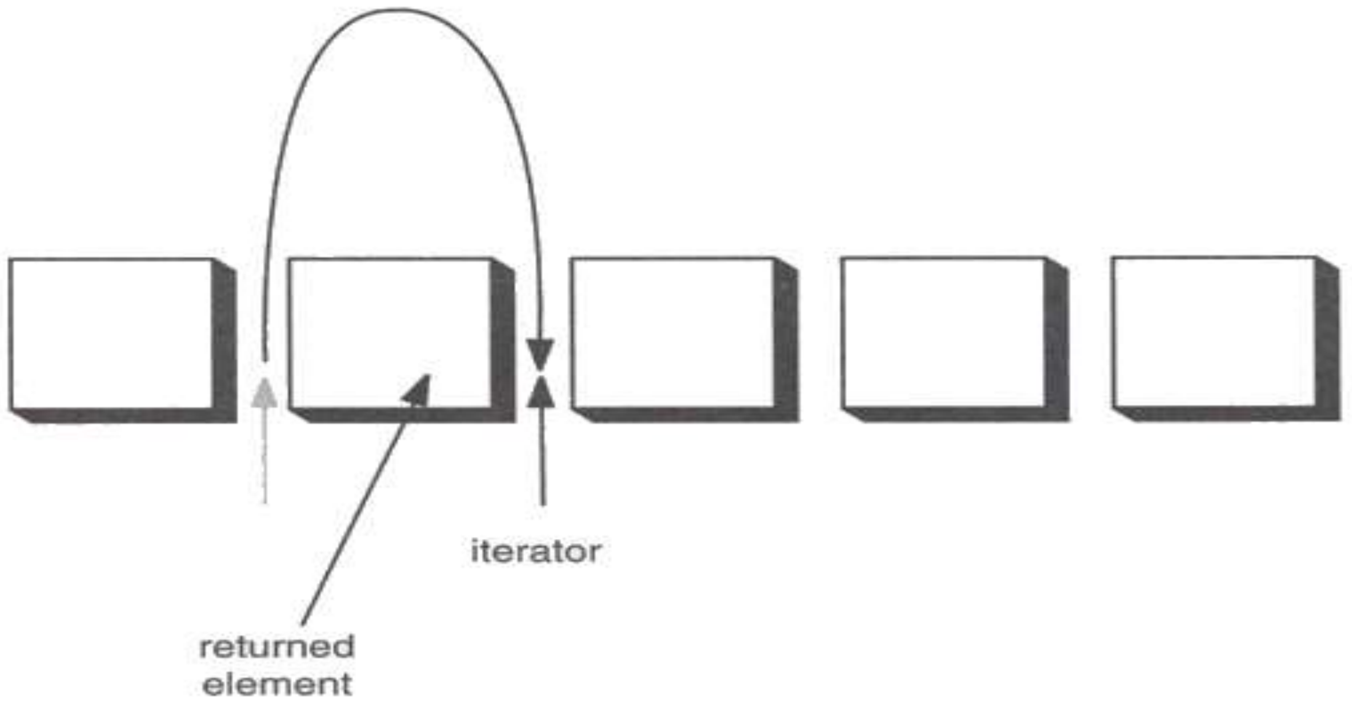
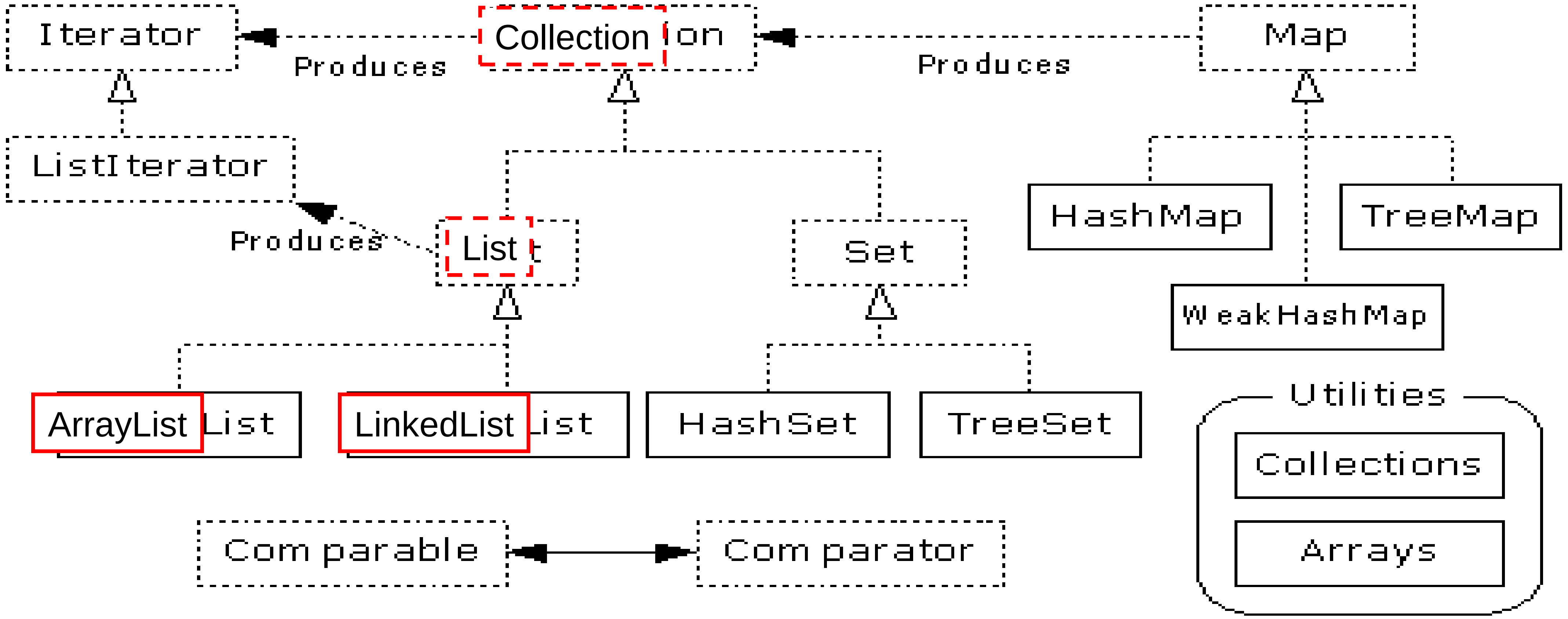


Figure 2–3: Advancing an iterator

ArrayList and LinkedList Context



List Implementations

- ArrayList
  - » low cost random access
  - » high cost insert and delete
  - » array that resizes if need be
- LinkedList
  - » sequential access
  - » low cost insert and delete
  - » high cost random access

ArrayList overview

- Constant time positional access (it’s an array)
- One tuning parameter, the initial capacity

```
public ArrayList(int initialCapacity) {
   super();
   if (initialCapacity < 0)
        throw new IllegalArgumentException(
              "Illegal Capacity: "+initialCapacity);
   this.elementData = new Object[initialCapacity];
}
```

ArrayList methods

- The indexed get and set methods of the List interface are appropriate to use since ArrayLists are backed by an array
  - » **Object get(int index)**
  - » **Object set(int index, Object element)**
- Indexed add and remove are provided, but can be costly if used frequently
  - » **void add(int index, Object element)**
  - » **Object remove(int index)**
- May want to resize in one shot if adding many elements
  - » **void ensureCapacity(int minCapacity)**

LinkedList overview

- Stores each element in a node
- Each node stores a link to the next and previous nodes
- Insertion and removal are inexpensive
  - » just update the links in the surrounding nodes
- Linear traversal is inexpensive
- Random access is expensive
  - » Start from beginning or end and traverse each node while counting

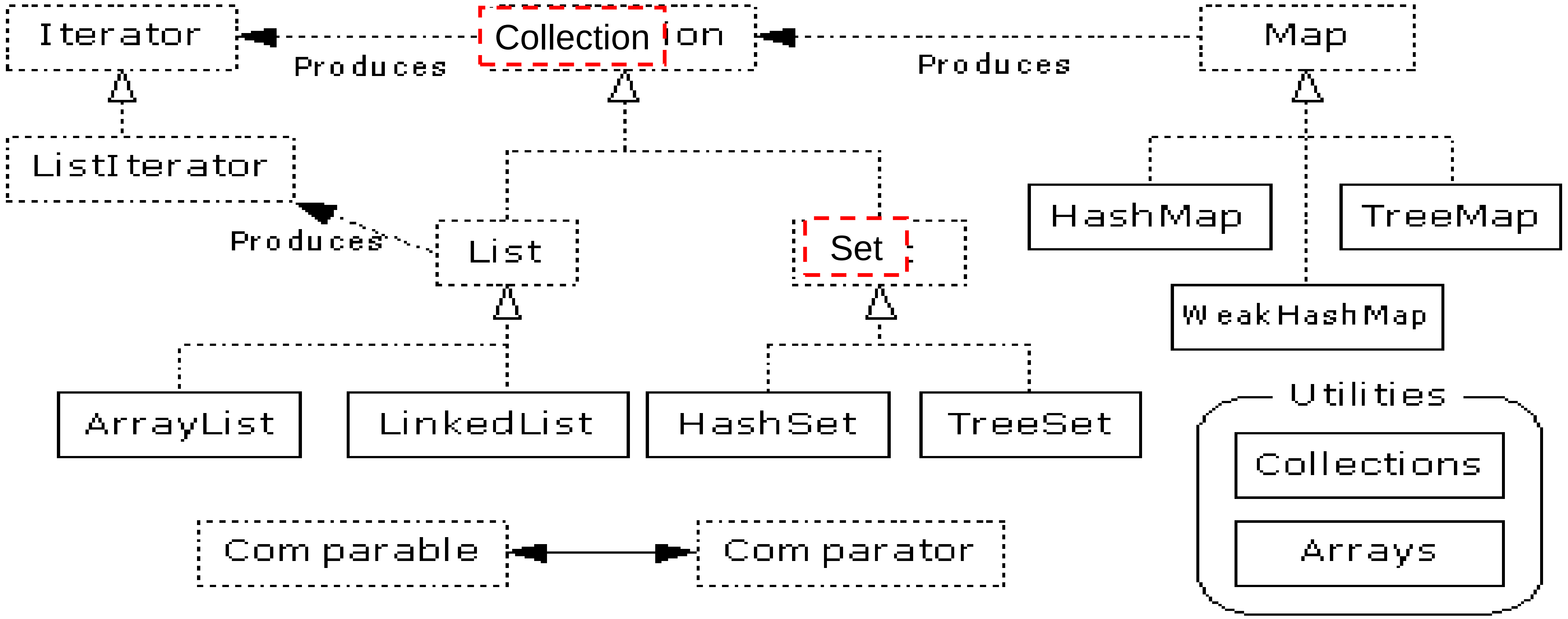
LinkedList entries

```
private static class Entry {
   Object element;
   Entry next;
   Entry previous;

   Entry(Object element, Entry next, Entry previous) {
       this.element = element;
       this.next = next;
       this.previous = previous;
   }
}

private Entry header = new Entry(null, null, null);

public LinkedList() {
   header.next = header.previous = header;
}
```

LinkedList methods

- The list is sequential, so access it that way
  - » **ListIterator listIterator()**
- ListIterator knows about position
  - » use **add()** from ListIterator to add at a position
  - » use **remove()** from ListIterator to remove at a position
- LinkedList knows a few things too
  - » **void addFirst(Object o), void addLast(Object o)**
  - » **Object getFirst(), Object getLast()**
  - » **Object removeFirst(), Object removeLast()**

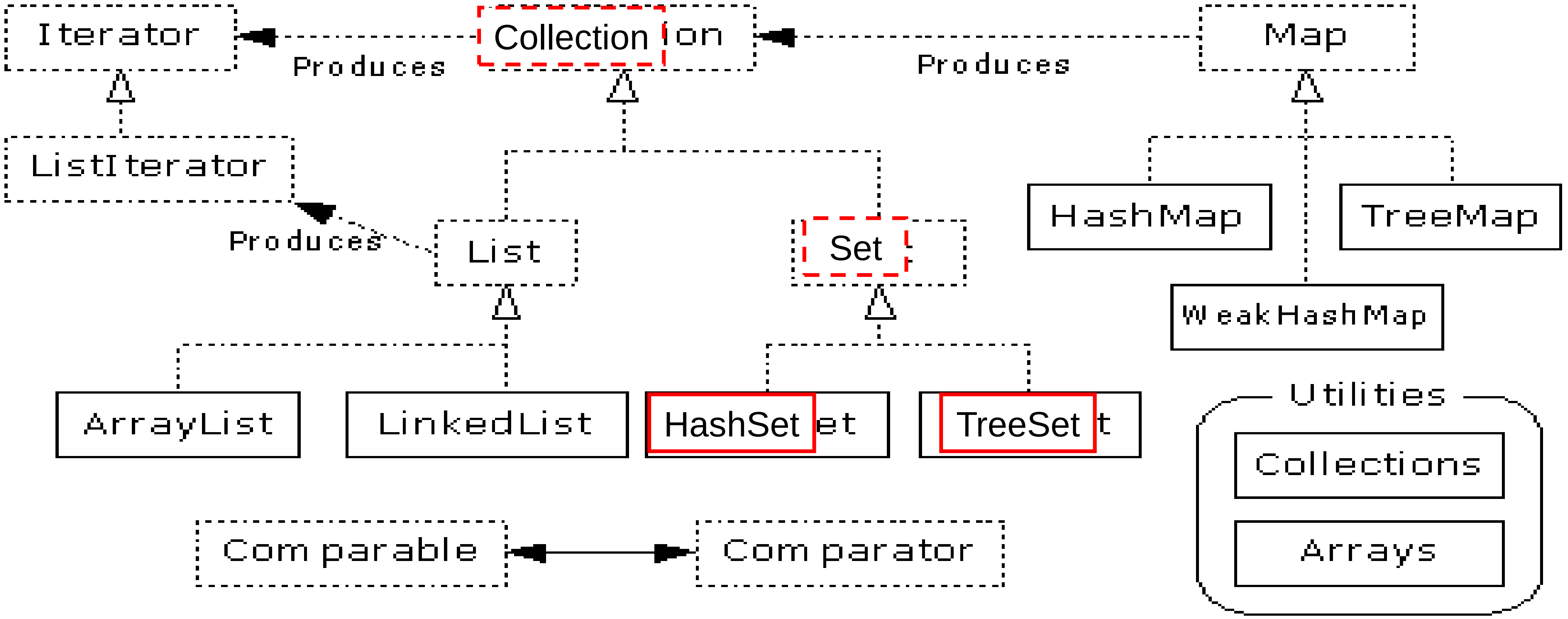
Set Interface Context



Set Interface

- Same methods as Collection
  - » different contract - no duplicate entries
- Defines two fundamental methods
  - » `boolean add(Object o)` - reject duplicates
  - » `Iterator iterator()`
- Provides an Iterator to step through the elements in the Set
  - » No guaranteed order in the basic Set interface
  - » There is a SortedSet interface that extends Set

HashSet and TreeSet Context [CO5]



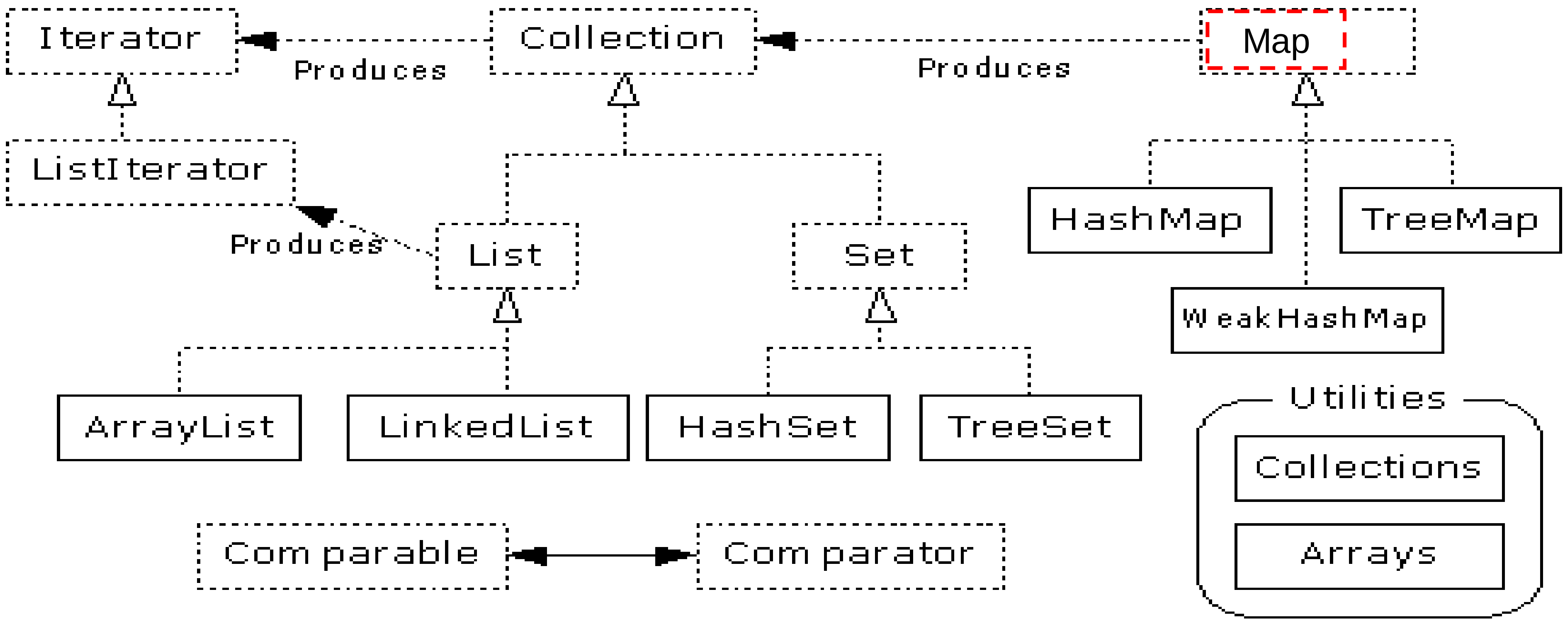
HashSet

- Find and add elements very quickly
  - » uses hashing implementation in HashMap
- Hashing uses an array of linked lists
  - » The **`hashCode()`** is used to index into the array
  - » Then **`equals()`** is used to determine if element is in the (short) list of elements at that index
- No order imposed on elements
- The **`hashCode()`** method and the **`equals()`** method must be compatible
  - » if two objects are equal, they must have the same **`hashCode()`** value

TreeSet

- Elements can be inserted in any order
- The TreeSet stores them in order
  - » Red-Black Trees out of Cormen-Leiserson-Rivest
- An iterator always presents them in order
- Default order is defined by natural order
  - » objects implement the Comparable interface
  - » TreeSet uses **`compareTo(Object o)`** to sort
- Can use a different Comparator
  - » provide Comparator to the TreeSet constructor

Map Interface Context



Map Interface [CO5]

- Stores key/value pairs
- Maps from the key to the value
- Keys are unique
  - » a single key only appears once in the Map
  - » a key can map to only one value
- Values do not have to be unique

Map methods

Object put(Object key, Object value)

Object get(Object key)

Object remove(Object key)

boolean containsKey(Object key)

boolean containsValue(Object value)

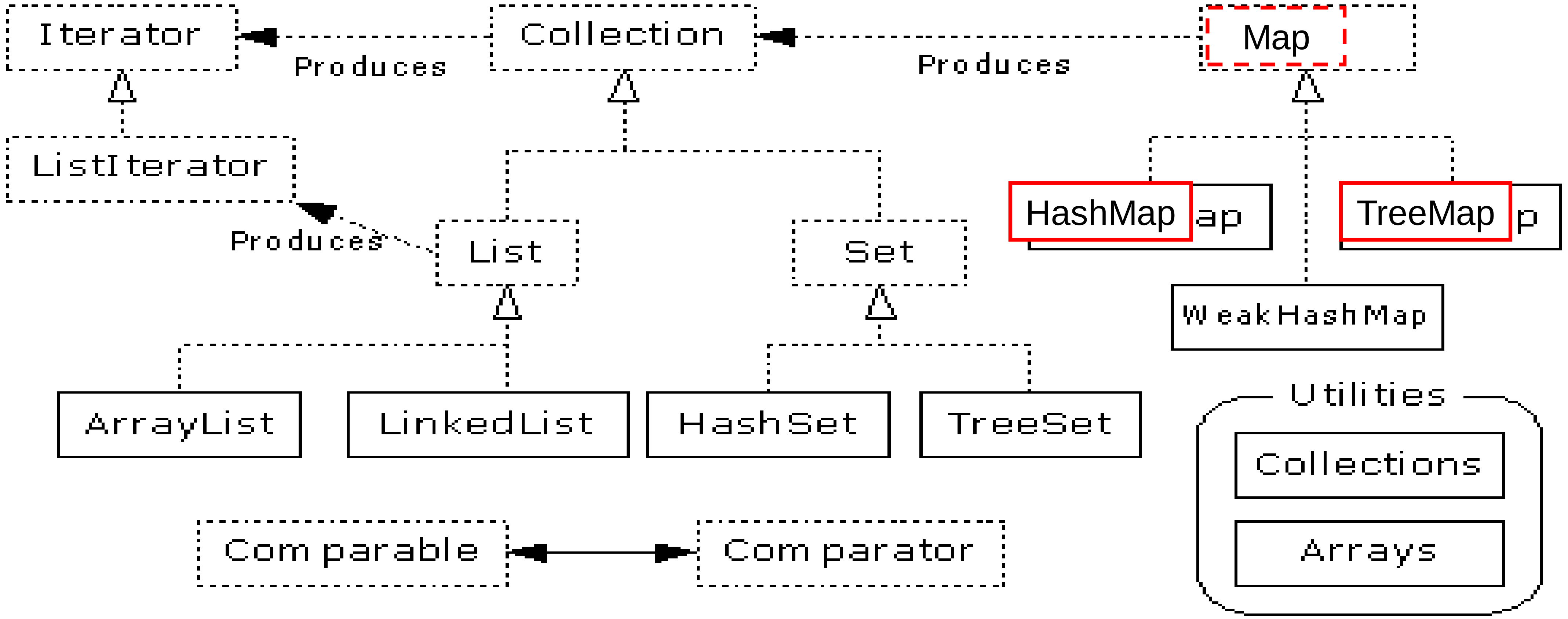
int size()

boolean isEmpty()

Map views

- A means of iterating over the keys and values in a Map
- **Set keySet()**
  - » returns the Set of keys contained in the Map
- **Collection values()**
  - » returns the Collection of values contained in the Map. This Collection is not a Set, as multiple keys can map to the same value.
- **Set entrySet()**
  - » returns the Set of key-value pairs contained in the Map. The Map interface provides a small nested interface called Map.Entry that is the type of the elements in this Set.

HashMap and TreeMap Context



HashMap and TreeMap

- HashMap
  - » The keys are a set - unique, unordered
  - » Fast

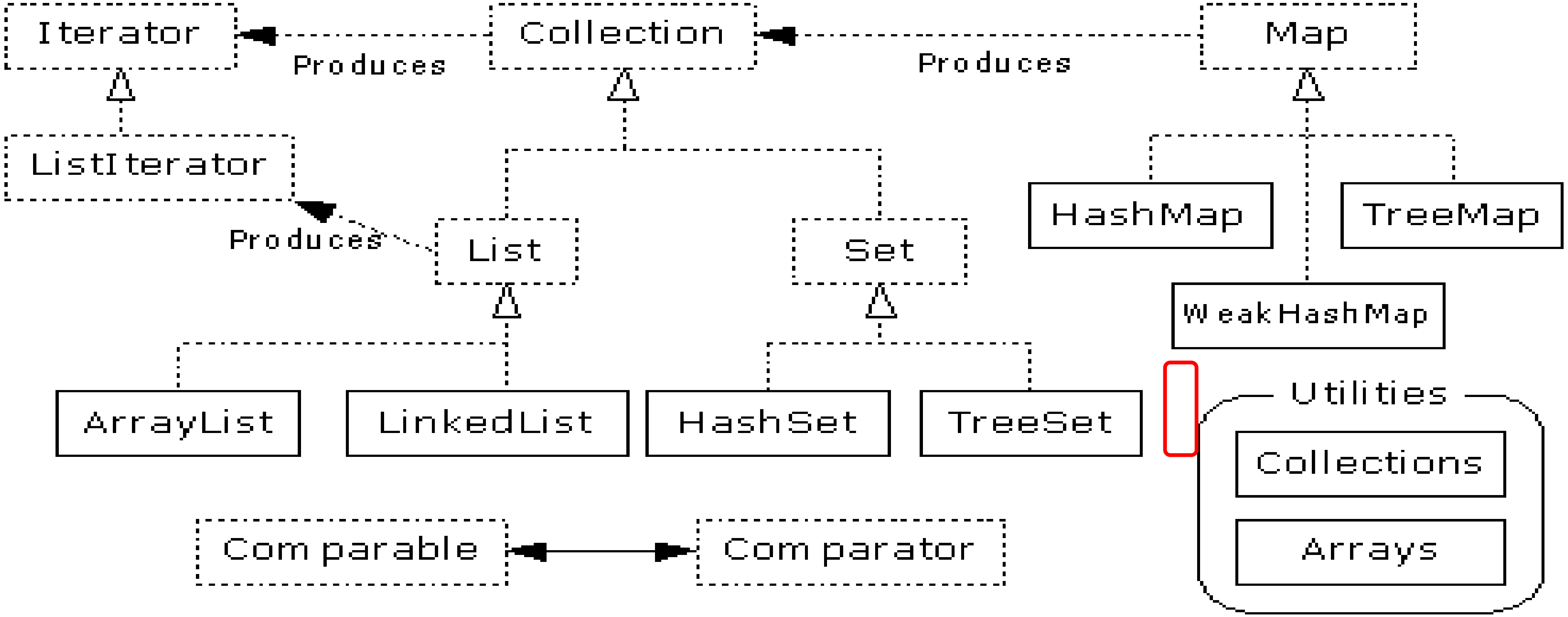
- TreeMap
  - » The keys are a set - unique, ordered
  - » Same options for ordering as a TreeSet
    - *Natural order (Comparable, compareTo(Object))*
    - *Special order (Comparator, compare(Object, Object))*

Bulk Operations

- In addition to the basic operations, a Collection may provide “bulk” operations

```
boolean containsAll(Collection c);
boolean addAll(Collection c);    // Optional
boolean removeAll(Collection c); // Optional
boolean retainAll(Collection c); // Optional
void clear();                    // Optional
Object[] toArray();
Object[] toArray(Object a[]);
```

Utilities Context



Utilities

Lecture 5

- Using Wrapper classes and Using Lists
- Sets, Maps and Queues

Utilities

- The Collections class provides a number of static methods for fundamental algorithms
- Most operate on Lists, some on all Collections
  - » Sort, Search, Shuffle
  - » Reverse, fill, copy
  - » Min, max
- Wrappers
  - » synchronized Collections, Lists, Sets, etc
  - » unmodifiable Collections, Lists, Sets, etc

More Legacy classes

- Vector
  - » use ArrayList
- Stack
  - » use LinkedList
- BitSet
  - » use ArrayList of boolean, unless you can’t stand the thought of the wasted space
- Properties
  - » legacies are sometimes hard to walk away from …
  - » see next few pages

Properties class

- Located in java.util package
- Special case of Hashtable
  - » Keys and values are Strings
  - » Tables can be saved to/loaded from file

System properties

- Java VM maintains set of properties that define system environment
  - » Set when VM is initialized
  - » Includes information about current user, VM version, Java environment, and OS configuration

```
Properties prop = System.getProperties();
Enumeration e = prop.propertyNames();
while (e.hasMoreElements()) {
   String key = (String) e.nextElement();
   System.out.println(key + " value is " +
        prop.getProperty(key));
}
```

DAILY QUIZ

1) What is the Collection framework in Java? [CO5]
2) What are the main differences between array and collection? [CO5]
3) Explain various interfaces used in Collection framework? [CO5]
4) What is the difference between ArrayList and Vector? [CO5]
5) What is the difference between ArrayList and LinkedList? [CO5]
6) What is the difference between Iterator and ListIterator? [CO5]
7) What is the difference between Iterator and Enumeration? [CO5]
8) What is the difference between List and Set? [CO5]
9) What is the difference between Set and Map? [CO5]
10) What is the difference between HashSet and HashMap? [CO5]

WEEKLY ASSIGNMENT

1. Explain importance of container in java with example. [CO5]
2. Explain the different types of layout managers. [CO5]
3. Compare HashMap and Hashtable. Explain with the help of an examples. [CO5]
4. Analyze Generics and list the advantages Of Using Generics.[CO5]
5. How does HashMap handle collisions in Java? [CO5]
6. List down the primary interfaces provided by Java Collections Framework? [CO5]
7. Illustrate properties class. [CO5]
8. Explain importance of container in java with example. [CO5]
9. Elaborate Array and List. Write a program for ArrayList. [CO5]
10. Illustrate event-driven programming. [CO5]

TOPIC LINKS

- https://nptel.ac.in/courses/106/105/106105191/

- https://nptel.ac.in/noc/courses/noc20/SEM1/noc20-cs08/

MCQs

1. Which of these is a correct way of defining of a generic method?[CO5]
A. name(T1, T2, …, Tn) { /* … */ }
B. public name { /* … */ }
C. class name[T1, T2, …, Tn] { /* … */ }
D. name{T1, T2, …, Tn} { /* … */ }

Answer: B

2. Which of these package is used for graphical user interface?[CO5]
java.applet
java.awt
java.awt.image
java.io

Answer: B

MCQs

3. Where are the following four methods commonly used?[CO5]

1) public void add(Component c)
2) public void setSize(int width,int height)
3) public void setLayout(LayoutManager m)
4) public void setVisible(boolean)

a. Graphics class
b. Component class
c. Both A & B
d. None of the above

Answer: B

4. Which is the container that doesn't contain title bar and MenuBars but it can have other components like button, textfield etc?[CO5]

a. Window
b. Frame
c. Panel
d. Container

Answer : C

MCQs

5. Which package provides many event classes and Listener interfaces for event handling?[CO5]

a. java.awt

b. java.awt.Graphics

c. java.awt.event

d. None of the above

Answer: c

6. Which is a component in AWT that can contain another components like buttons, textfields, labels etc.?[CO5]

a. Window

b. Container

c. Panel

d. Frame

Answer: b

MCQs

6. Which of the following is incorrect statement regarding the use of generics and parameterized types in Java?[CO5]
A. Generics provide type safety by shifting more type checking responsibilities to the compiler
B. Generics and parameterized types eliminate the need for down casts when using Java Collections
C. When designing your own collections class (say, a linked list), generics and parameterized types allow you to achieve type safety with just a single class definition as opposed to defining multiple classes
Answer: C

7. Which of the following allows us to call generic methods as a normal method?[CO5]
A. Type Interface
B. Interface
C. Inner class
D. All of the mentioned
Answer: A

MCQs

8. Why are generics used?[CO5]
A. Generics make code more fast
B. Generics make code more optimised and readable
C. Generics add stability to your code by making more of your bugs detectable at compile time
D. Generics add stability to your code by making more of your bugs detectable at a runtime

Answer: C

9. Which of the following reference types cannot be generic?
A. Anonymous inner class[CO5]
B. Interface
C. Inner class
D. All of the mentioned

Answer: A

MCQs

10. Which of these types cannot be used to initiate a generic type?[CO5]
A. Integer class
B. Float Class
C. Primitive Types
D. Collections

Answer: C

11. Which of these instances cannot be created?[CO5]
A. Integer Instance
B. Generic Class Instance
C. Generic Type Instance
D. Collection Instances

Answer: C

Glossary Questions

1. **Attempt all the parts. Pick the correct option from glossary.** [CO5]

i) Generic type ii) container iii) java.awt iv) Type Interface

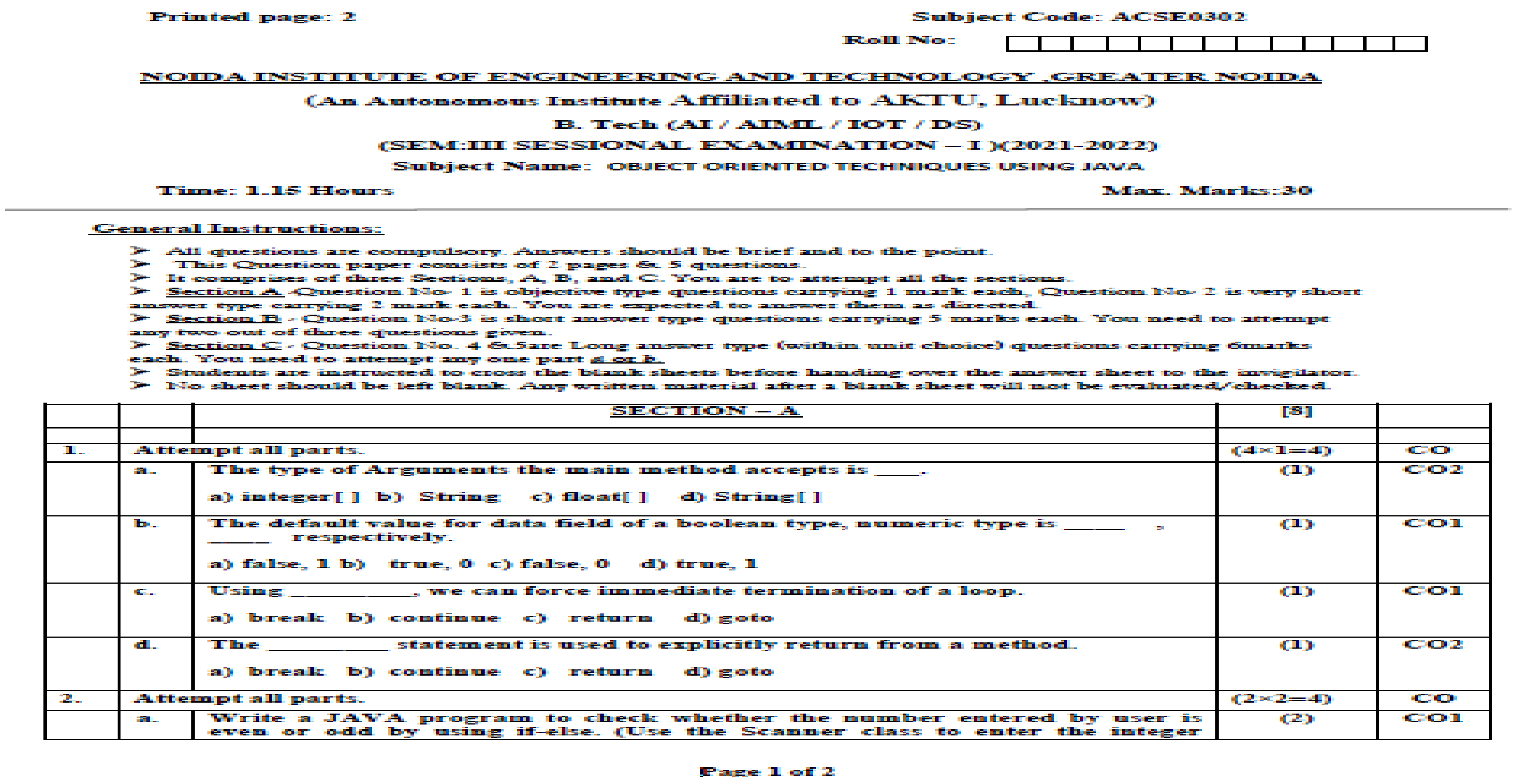
a) __________ AWT component contains another components like buttons, labels, etc.

b)With the help of __________ generic methods can be called as normal methods.

c) we cannot create __________ instances in java.

d) _____package in java provides listener interfaces for event handling.

Sessional Paper-1



Printed page: 2 Subject Code: ACSE0302

Roll No: | | | | | | | | | | | | | |

NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY ,GREATER NOIDA

(An Autonomous Institute Affiliated to AKTU, Lucknow)

B. Tech (AI / AIML / IOT / DS)

(SEM:III SESSIONAL EXAMINATION – I)(2021-2022)

Subject Name: OBJECT ORIENTED TECHNIQUES USING JAVA

Time: 1.15 Hours Max. Marks:30

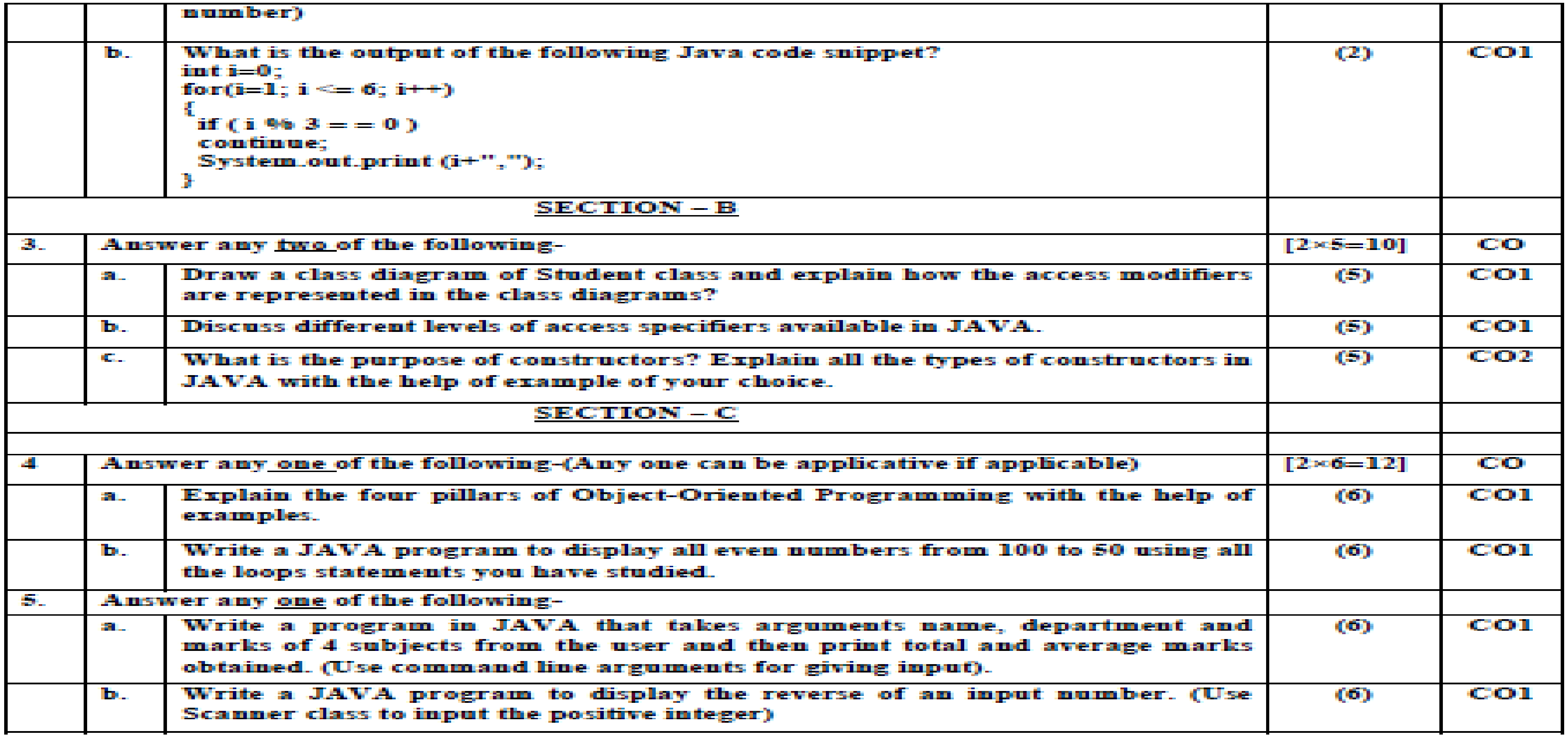
General Instructions:

- All questions are compulsory. Answers should be brief and to the point.
- This Question paper consists of 2 pages & 5 questions.
- It comprises of three Sections, A, B, and C. You are to attempt all the sections.
- Section A -Question No- 1 is objective type questions carrying 1 mark each, Question No- 2 is very short answer type carrying 2 mark each. You are expected to answer them as directed.
- Section B - Question No-3 is short answer type questions carrying 5 marks each. You need to attempt any two out of three questions given.
- Section C - Question No. 4 & 5are Long answer type (within unit choice) questions carrying 6marks each. You need to attempt any one part a or b.
- Students are instructed to cross the blank sheets before handing over the answer sheet to the invigilator.
- No sheet should be left blank. Any written material after a blank sheet will not be evaluated/checked.

		SECTION – A	[8]	
1.	Attempt all parts.		(4×1=4)	CO
	a.	The type of Arguments the main method accepts is ____. a) integer[ ] b) String c) float[ ] d) String[ ]	(1)	CO2
	b.	The default value for data field of a boolean type, numeric type is _____ , _____ respectively. a) false, 1 b) true, 0 c) false, 0 d) true, 1	(1)	CO1
	c.	Using __________, we can force immediate termination of a loop. a) break b) continue c) return d) goto	(1)	CO1
	d.	The __________ statement is used to explicitly return from a method. a) break b) continue c) return d) goto	(1)	CO2
2.	Attempt all parts.		(2×2=4)	CO
	a.	Write a JAVA program to check whether the number entered by user is even or odd by using if-else. (Use the Scanner class to enter the integer	(2)	CO1

Page 1 of 2

Conti….

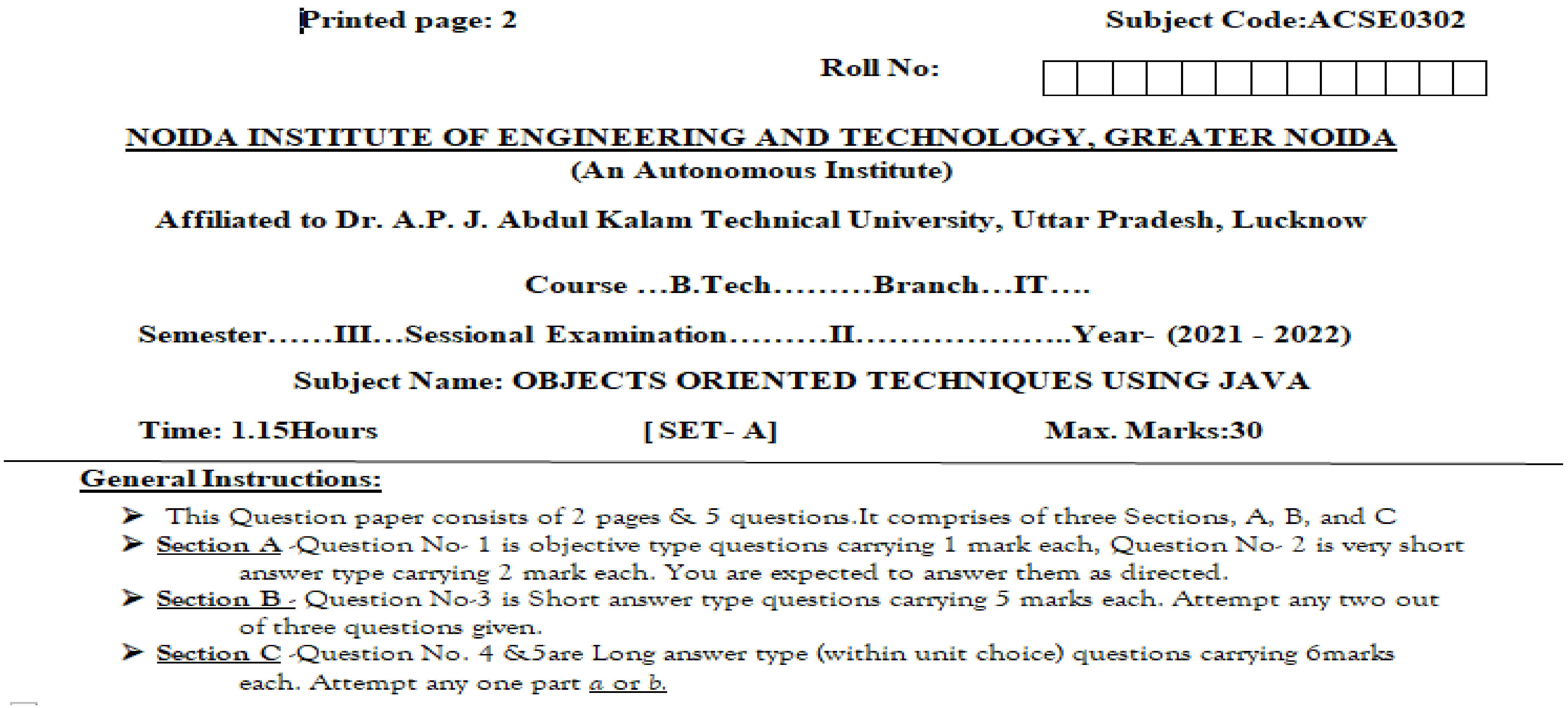


		number)		

	b.	What is the output of the following Java code snippet?
		`int i=0;`
		`for(i=1; i <= 6; i++)`
		`{`
		`  if ( i % 3 = = 0 )`
		`  continue;`
		`  System.out.print (i+",");`
		`}`
			(2)	CO1

SECTION – B				
3.	Answer any two of the following-		[2×5=10]	CO
	a.	Draw a class diagram of Student class and explain how the access modifiers are represented in the class diagrams?	(5)	CO1
	b.	Discuss different levels of access specifiers available in JAVA.	(5)	CO1
	c.	What is the purpose of constructors? Explain all the types of constructors in JAVA with the help of example of your choice.	(5)	CO2
SECTION – C				
4	Answer any one of the following-(Any one can be applicative if applicable)		[2×6=12]	CO
	a.	Explain the four pillars of Object-Oriented Programming with the help of examples.	(6)	CO1
	b.	Write a JAVA program to display all even numbers from 100 to 50 using all the loops statements you have studied.	(6)	CO1
5.	Answer any one of the following-			
	a.	Write a program in JAVA that takes arguments name, department and marks of 4 subjects from the user and then print total and average marks obtained. (Use command line arguments for giving input).	(6)	CO1
	b.	Write a JAVA program to display the reverse of an input number. (Use Scanner class to input the positive integer)	(6)	CO1

Sessional Paper-2



Printed page: 2 Subject Code:ACSE0302

Roll No:

NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY, GREATER NOIDA

(An Autonomous Institute)

Affiliated to Dr. A.P. J. Abdul Kalam Technical University, Uttar Pradesh, Lucknow

Course ...B.Tech.........Branch...IT....

Semester......III...Sessional Examination.........II....................Year- (2021 - 2022)

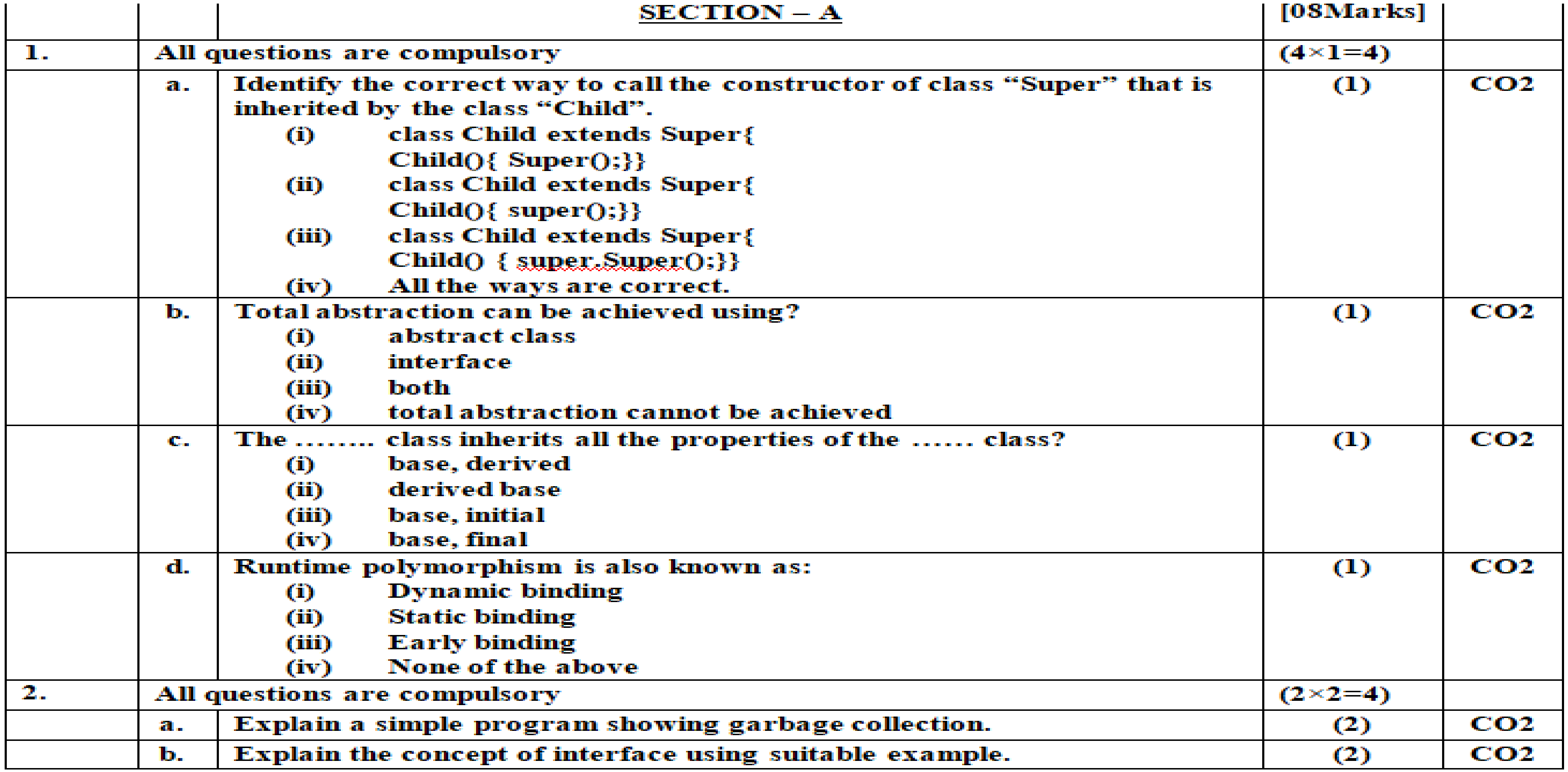
Subject Name: OBJECTS ORIENTED TECHNIQUES USING JAVA

Time: 1.15Hours [SET- A] Max. Marks:30

General Instructions:

- This Question paper consists of 2 pages & 5 questions.It comprises of three Sections, A, B, and C
- Section A -Question No- 1 is objective type questions carrying 1 mark each, Question No- 2 is very short answer type carrying 2 mark each. You are expected to answer them as directed.
- Section B - Question No-3 is Short answer type questions carrying 5 marks each. Attempt any two out of three questions given.
- Section C -Question No. 4 &5are Long answer type (within unit choice) questions carrying 6marks each. Attempt any one part *a or b.*

Conti….

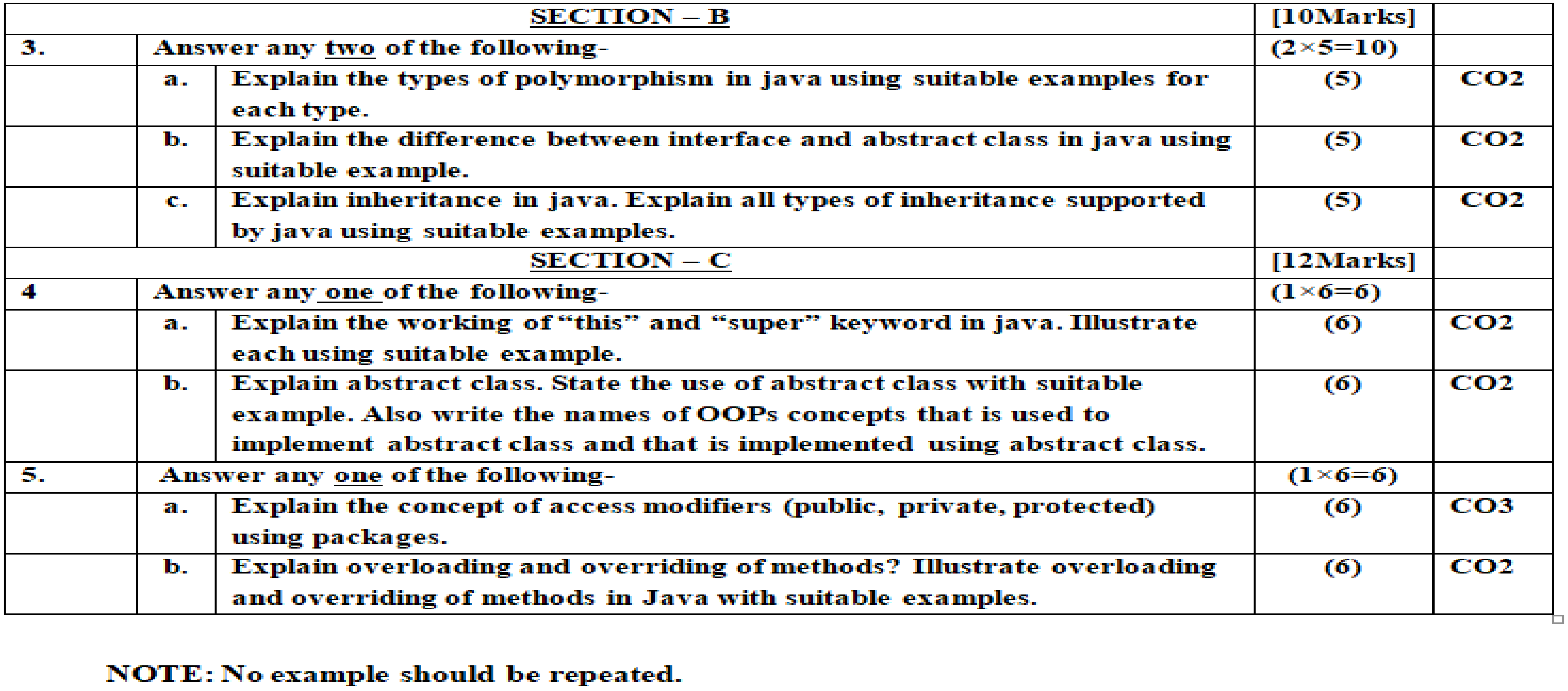


		SECTION – A	[08Marks]	
1.	All questions are compulsory		(4×1=4)	

	a.	Identify the correct way to call the constructor of class "Super" that is inherited by the class "Child".
		(i) class Child extends Super{
		Child(){ Super();}}
		(ii) class Child extends Super{
		Child(){ super();}}
		(iii) class Child extends Super{
		Child() { super.Super();}}
		(iv) All the ways are correct.
			(1)	CO2

	b.	Total abstraction can be achieved using? (i) abstract class (ii) interface (iii) both (iv) total abstraction cannot be achieved	(1)	CO2
	c.	The .... class inherits all the properties of the ...... class? (i) base, derived (ii) derived base (iii) base, initial (iv) base, final	(1)	CO2
	d.	Runtime polymorphism is also known as: (i) Dynamic binding (ii) Static binding (iii) Early binding (iv) None of the above	(1)	CO2
2.	All questions are compulsory		(2×2=4)	
	a.	Explain a simple program showing garbage collection.	(2)	CO2
	b.	Explain the concept of interface using suitable example.	(2)	CO2

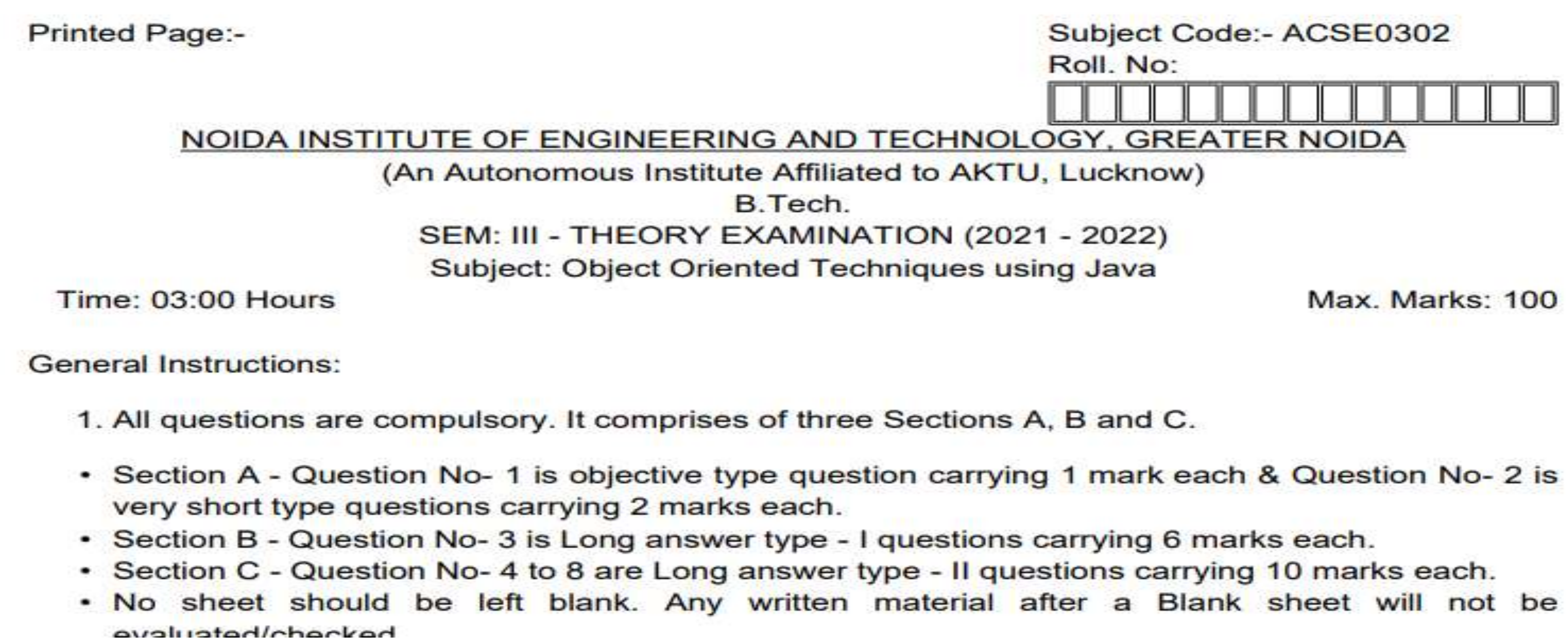
Conti..



		<u>SECTION – B</u>	[10Marks]	
3.	Answer any <u>two</u> of the following-		(2×5=10)	
	a.	Explain the types of polymorphism in java using suitable examples for each type.	(5)	CO2
	b.	Explain the difference between interface and abstract class in java using suitable example.	(5)	CO2
	c.	Explain inheritance in java. Explain all types of inheritance supported by java using suitable examples.	(5)	CO2
		<u>SECTION – C</u>	[12Marks]	
4	Answer any <u>one</u> of the following-		(1×6=6)	
	a.	Explain the working of "this" and "super" keyword in java. Illustrate each using suitable example.	(6)	CO2
	b.	Explain abstract class. State the use of abstract class with suitable example. Also write the names of OOPs concepts that is used to implement abstract class and that is implemented using abstract class.	(6)	CO2
5.	Answer any <u>one</u> of the following-		(1×6=6)	
	a.	Explain the concept of access modifiers (public, private, protected) using packages.	(6)	CO3
	b.	Explain overloading and overriding of methods? Illustrate overloading and overriding of methods in Java with suitable examples.	(6)	CO2

NOTE: No example should be repeated.

Old University Question Paper



Printed Page:- | Subject Code:- ACSE0302

Roll. No:

NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY, GREATER NOIDA

(An Autonomous Institute Affiliated to AKTU, Lucknow)

B.Tech.

SEM: III - THEORY EXAMINATION (2021 - 2022)

Subject: Object Oriented Techniques using Java

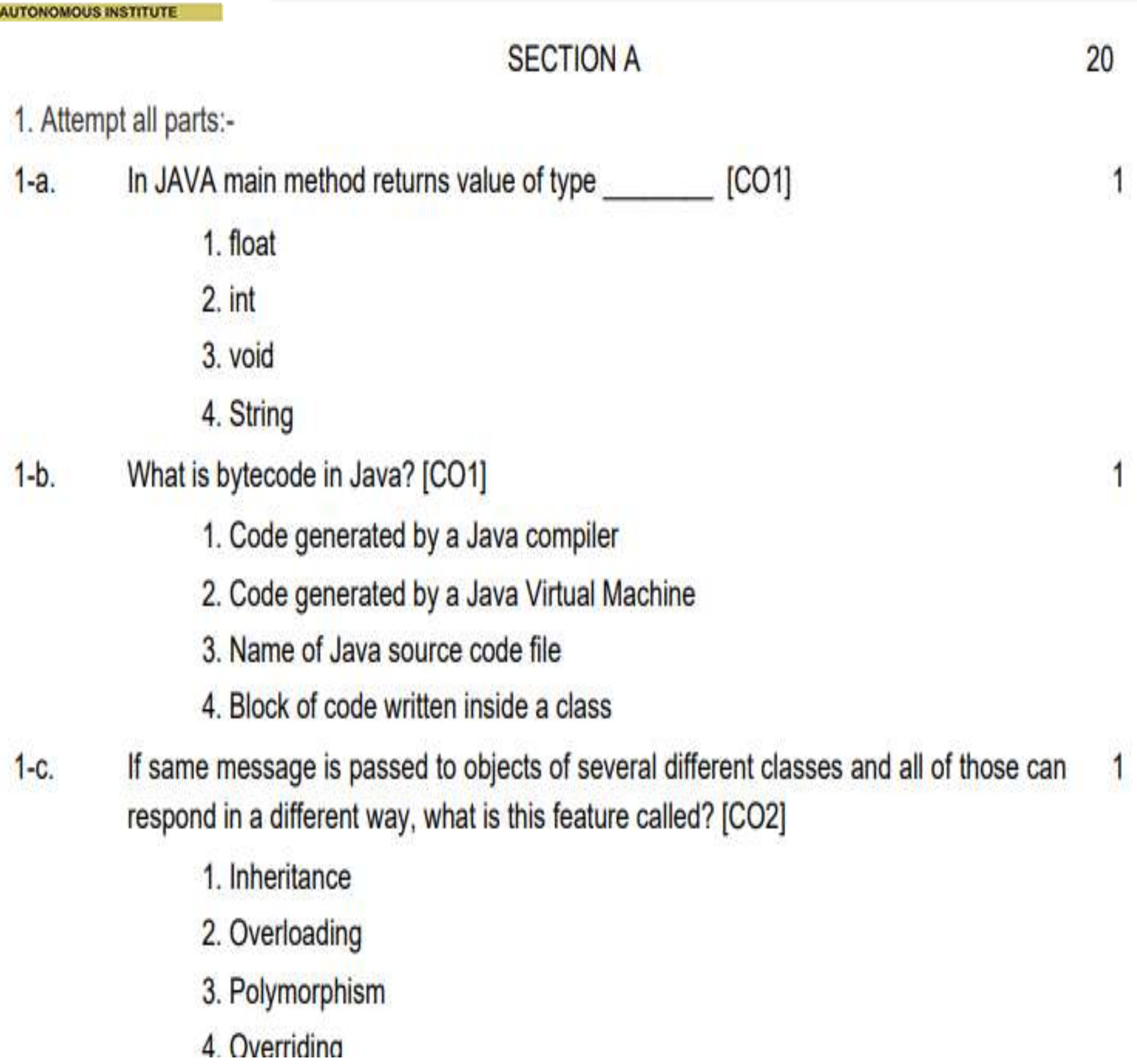
Time: 03:00 Hours | Max. Marks: 100

General Instructions:

1. All questions are compulsory. It comprises of three Sections A, B and C.

- Section A - Question No- 1 is objective type question carrying 1 mark each & Question No- 2 is very short type questions carrying 2 marks each.
- Section B - Question No- 3 is Long answer type - I questions carrying 6 marks each.
- Section C - Question No- 4 to 8 are Long answer type - II questions carrying 10 marks each.
- No sheet should be left blank. Any written material after a Blank sheet will not be evaluated/checked

Old University Question Paper



SECTION A 20

1. Attempt all parts:-

1-a. In JAVA main method returns value of type _______ [CO1] 1

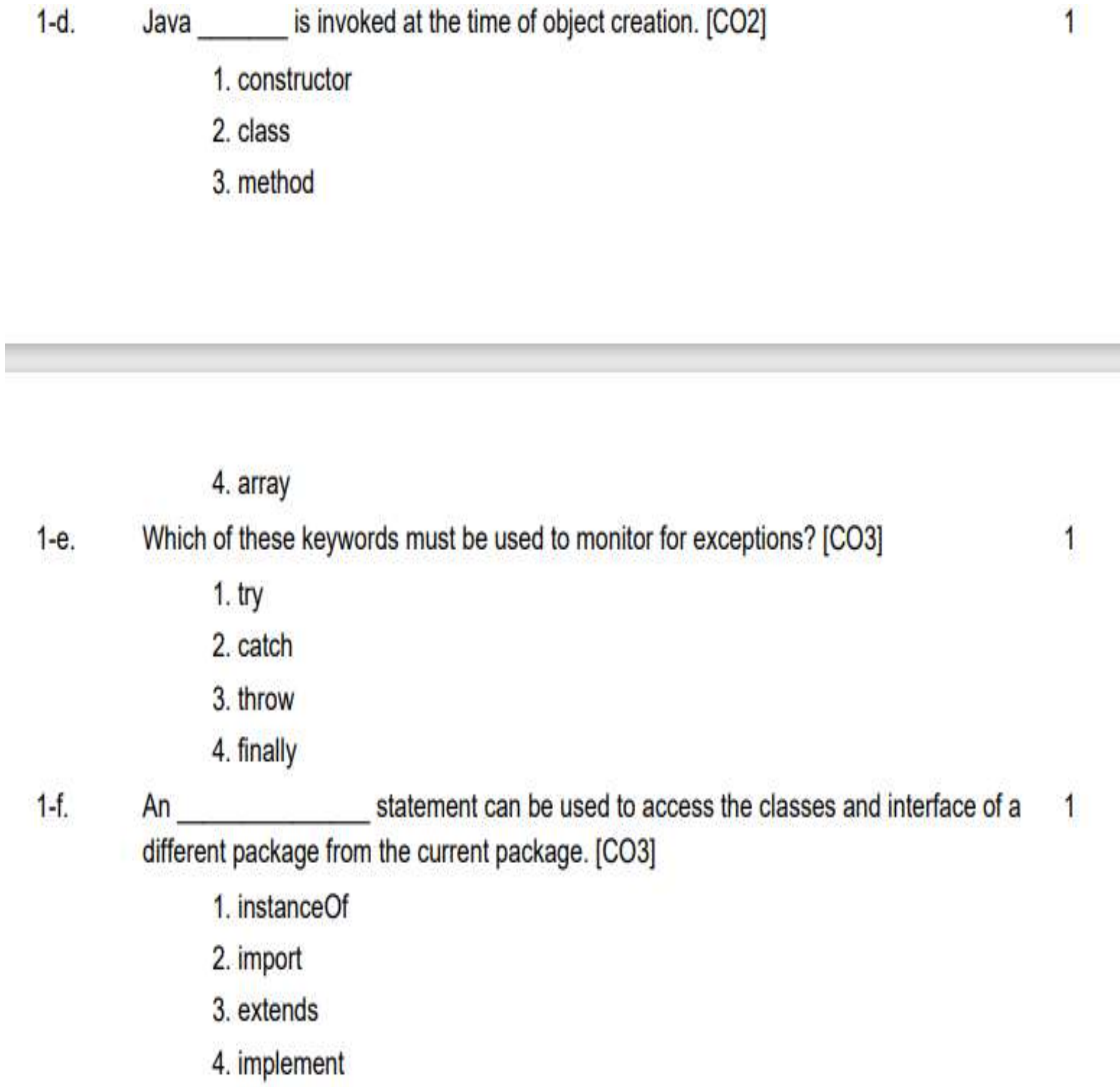
1. float
2. int
3. void
4. String

1-b. What is bytecode in Java? [CO1] 1

1. Code generated by a Java compiler
2. Code generated by a Java Virtual Machine
3. Name of Java source code file
4. Block of code written inside a class

1-c. If same message is passed to objects of several different classes and all of those can respond in a different way, what is this feature called? [CO2] 1

1. Inheritance
2. Overloading
3. Polymorphism
4. Overriding



1-d. Java ______ is invoked at the time of object creation. [CO2] 1

1. constructor
2. class
3. method
4. array

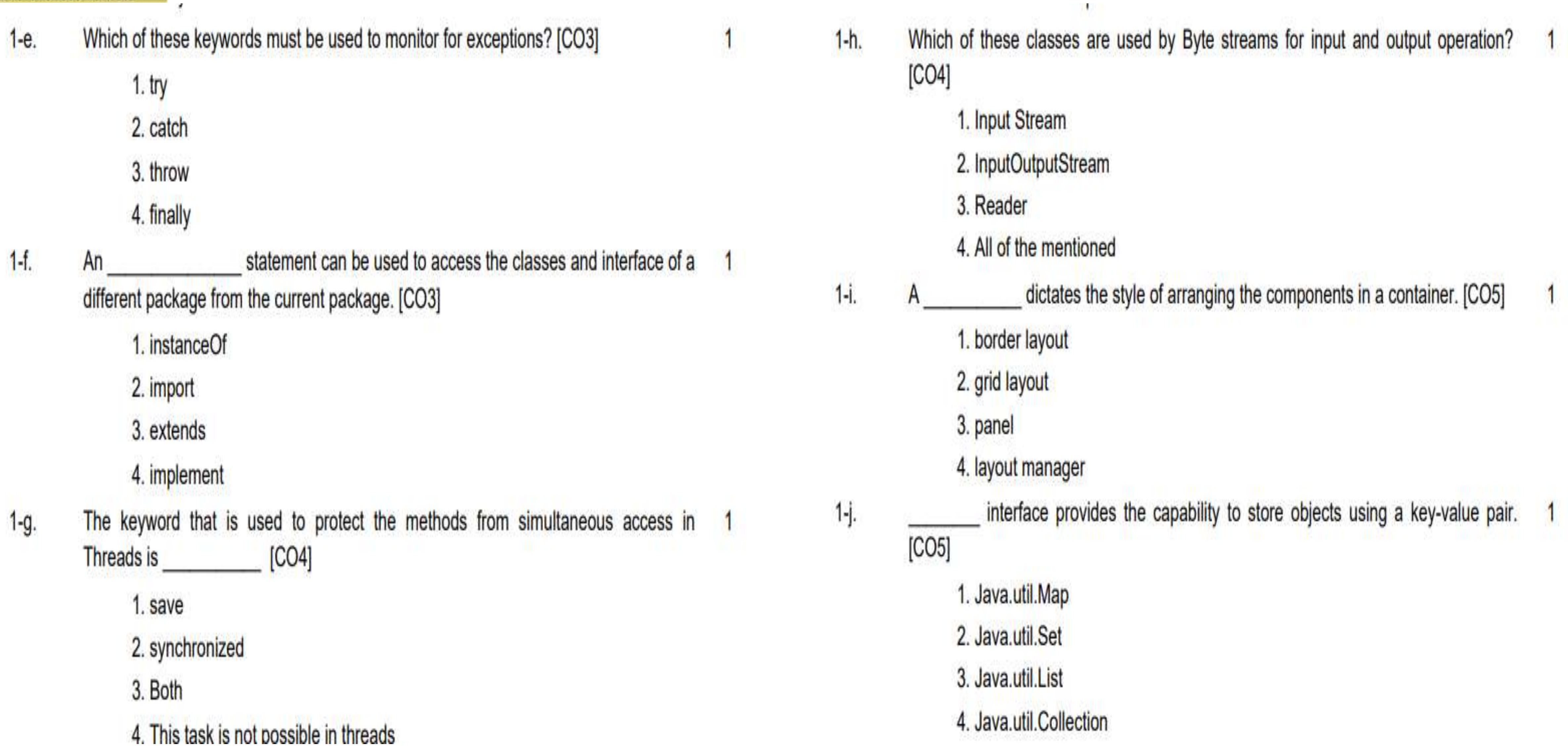
1-e. Which of these keywords must be used to monitor for exceptions? [CO3] 1

1. try
2. catch
3. throw
4. finally

1-f. An ____________ statement can be used to access the classes and interface of a different package from the current package. [CO3] 1

1. instanceOf
2. import
3. extends
4. implement

Old University Question Paper



1-e. Which of these keywords must be used to monitor for exceptions? [CO3] 1

1. try
2. catch
3. throw
4. finally

1-f. An ____________ statement can be used to access the classes and interface of a different package from the current package. [CO3] 1

1. instanceOf
2. import
3. extends
4. implement

1-g. The keyword that is used to protect the methods from simultaneous access in Threads is _________ [CO4] 1

1. save
2. synchronized
3. Both
4. This task is not possible in threads

1-h. Which of these classes are used by Byte streams for input and output operation? [CO4] 1

1. Input Stream
2. InputOutputStream
3. Reader
4. All of the mentioned

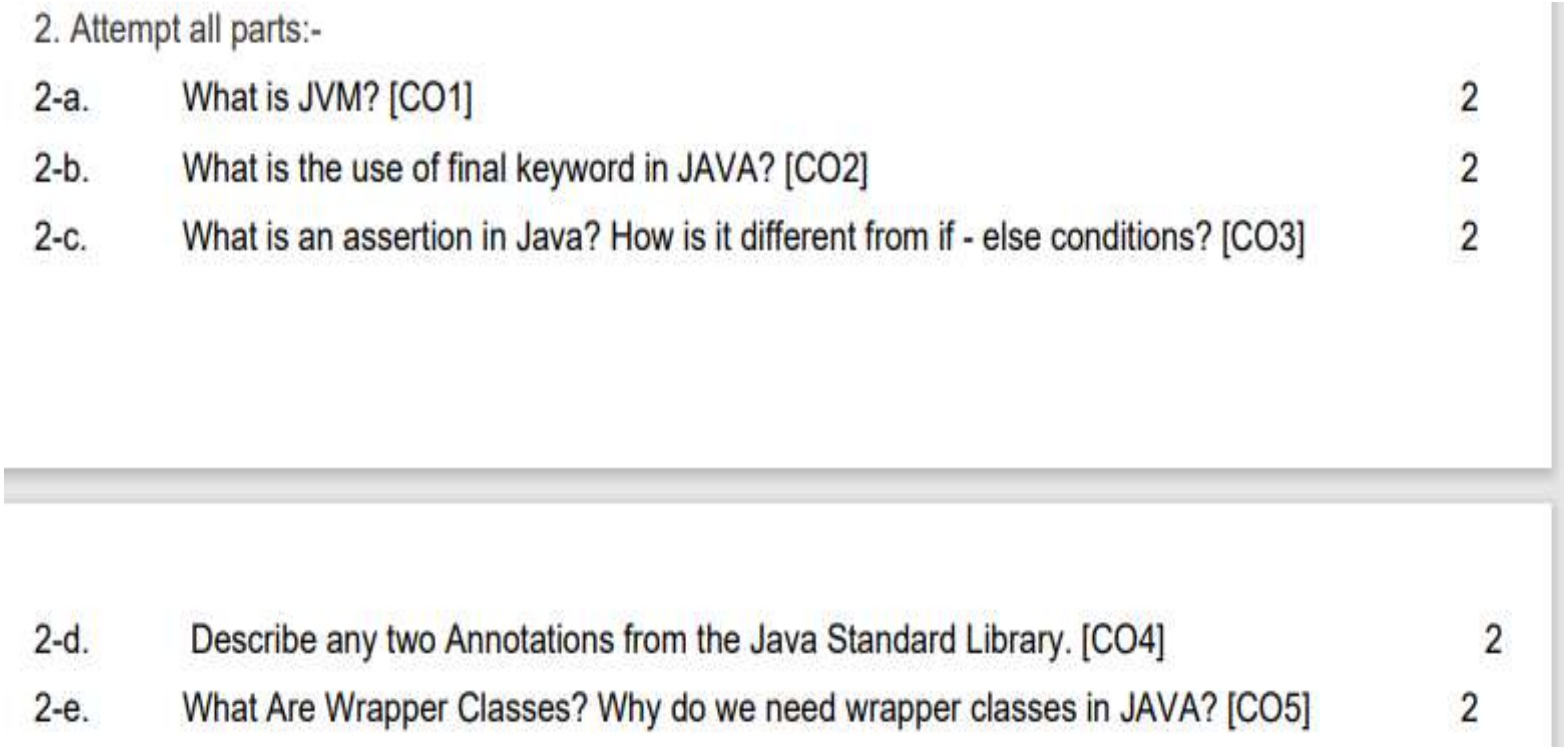
1-i. A _________ dictates the style of arranging the components in a container. [CO5] 1

1. border layout
2. grid layout
3. panel
4. layout manager

1-j. ______ interface provides the capability to store objects using a key-value pair. [CO5] 1

1. Java.util.Map
2. Java.util.Set
3. Java.util.List
4. Java.util.Collection

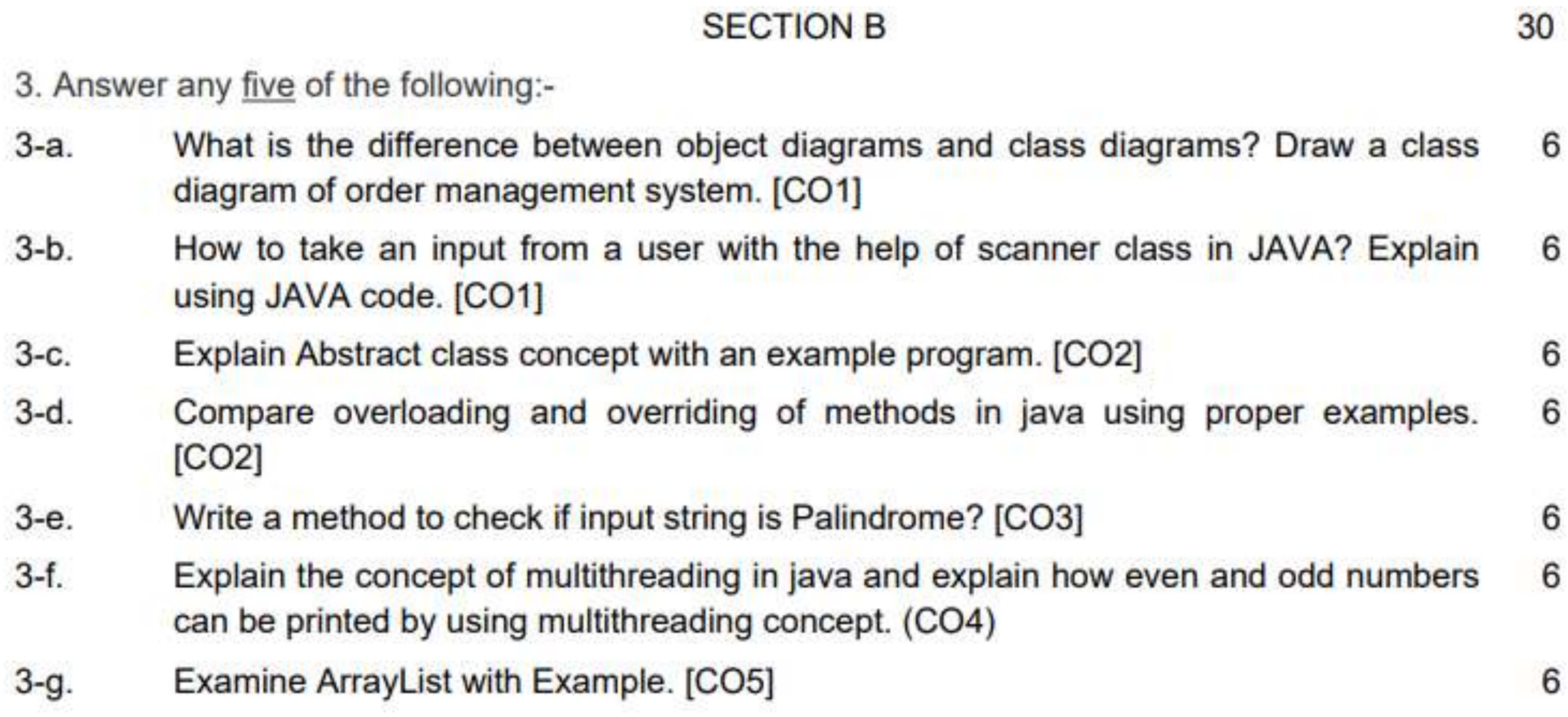
Old University Question Paper



2. Attempt all parts:-

2-a.	What is JVM? [CO1]	2
2-b.	What is the use of final keyword in JAVA? [CO2]	2
2-c.	What is an assertion in Java? How is it different from if - else conditions? [CO3]	2
2-d.	Describe any two Annotations from the Java Standard Library. [CO4]	2
2-e.	What Are Wrapper Classes? Why do we need wrapper classes in JAVA? [CO5]	2

Old University Question Paper

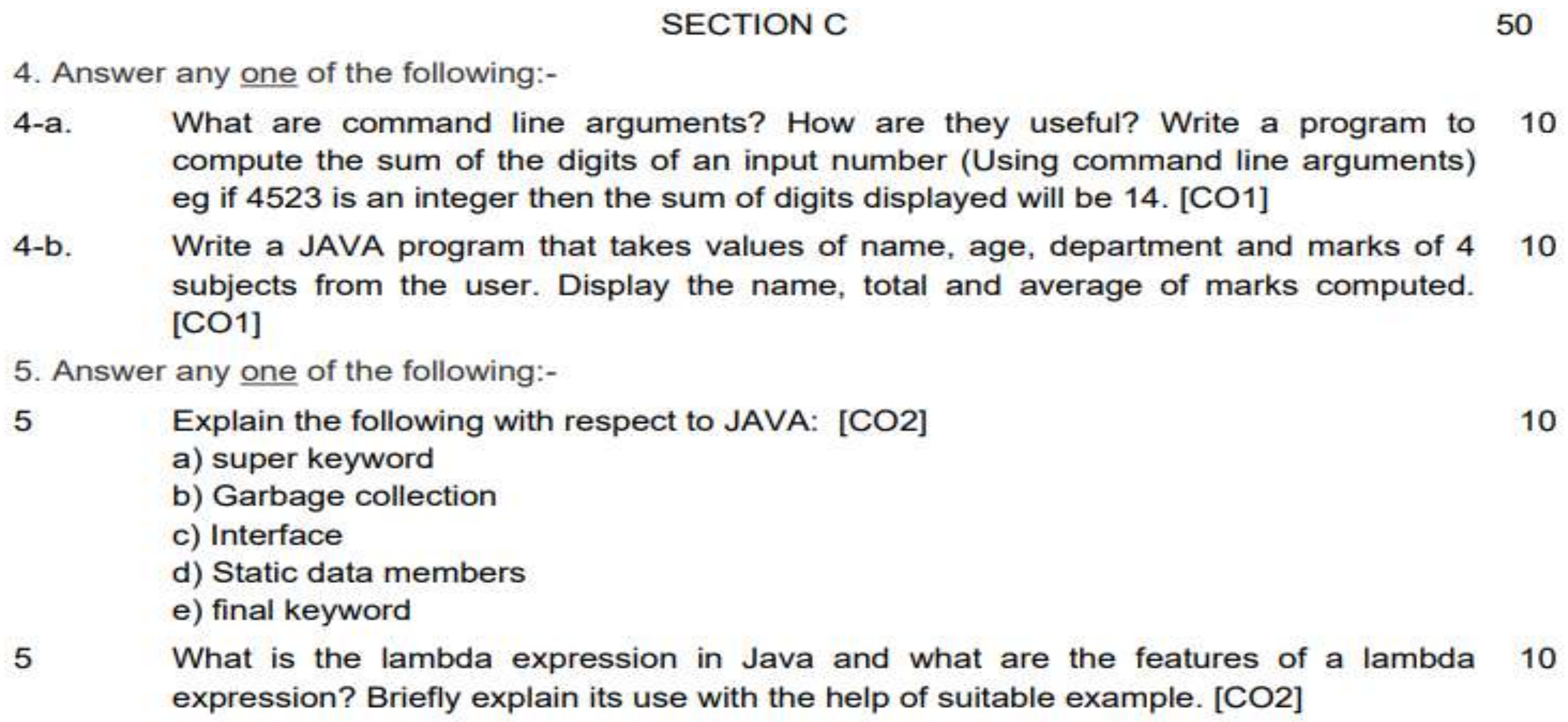


SECTION B 30

3. Answer any five of the following:-

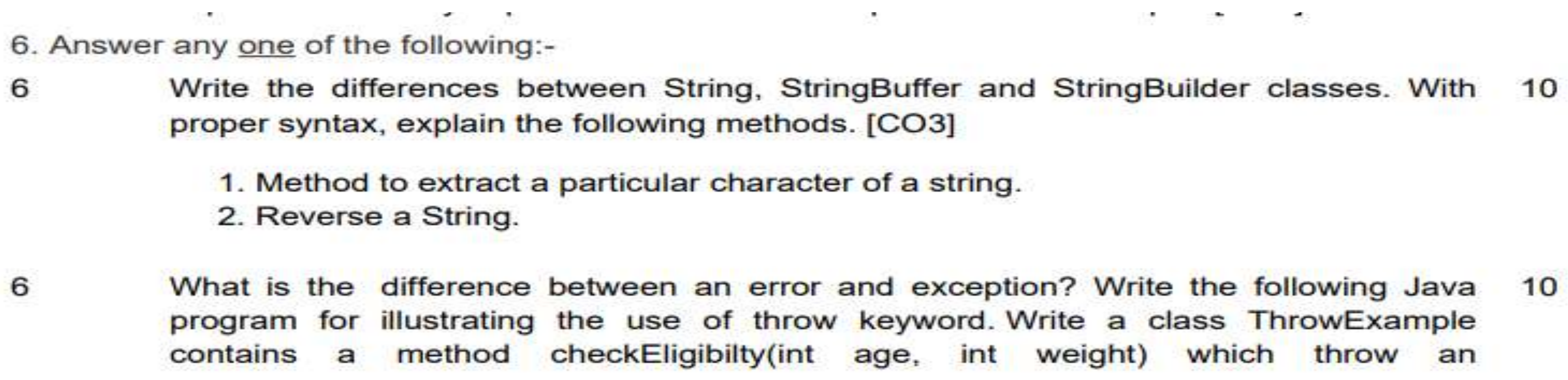
3-a.	What is the difference between object diagrams and class diagrams? Draw a class diagram of order management system. [CO1]	6
3-b.	How to take an input from a user with the help of scanner class in JAVA? Explain using JAVA code. [CO1]	6
3-c.	Explain Abstract class concept with an example program. [CO2]	6
3-d.	Compare overloading and overriding of methods in java using proper examples. [CO2]	6
3-e.	Write a method to check if input string is Palindrome? [CO3]	6
3-f.	Explain the concept of multithreading in java and explain how even and odd numbers can be printed by using multithreading concept. (CO4)	6
3-g.	Examine ArrayList with Example. [CO5]	6

Old University Question Paper



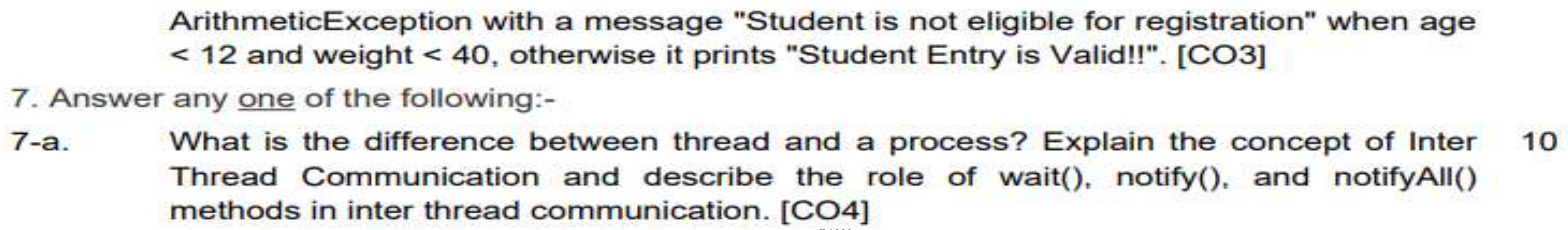
	SECTION C	50
4. Answer any one of the following:-		
4-a.	What are command line arguments? How are they useful? Write a program to compute the sum of the digits of an input number (Using command line arguments) eg if 4523 is an integer then the sum of digits displayed will be 14. [CO1]	10
4-b.	Write a JAVA program that takes values of name, age, department and marks of 4 subjects from the user. Display the name, total and average of marks computed. [CO1]	10
5. Answer any one of the following:-		
5	Explain the following with respect to JAVA: [CO2] a) super keyword b) Garbage collection c) Interface d) Static data members e) final keyword	10
5	What is the lambda expression in Java and what are the features of a lambda expression? Briefly explain its use with the help of suitable example. [CO2]	10

Old University Question Paper



6. Answer any one of the following:-

6	Write the differences between String, StringBuffer and StringBuilder classes. With proper syntax, explain the following methods. [CO3] 1. Method to extract a particular character of a string. 2. Reverse a String.	10
6	What is the difference between an error and exception? Write the following Java program for illustrating the use of throw keyword. Write a class ThrowExample contains a method checkEligibilty(int age, int weight) which throw an	10

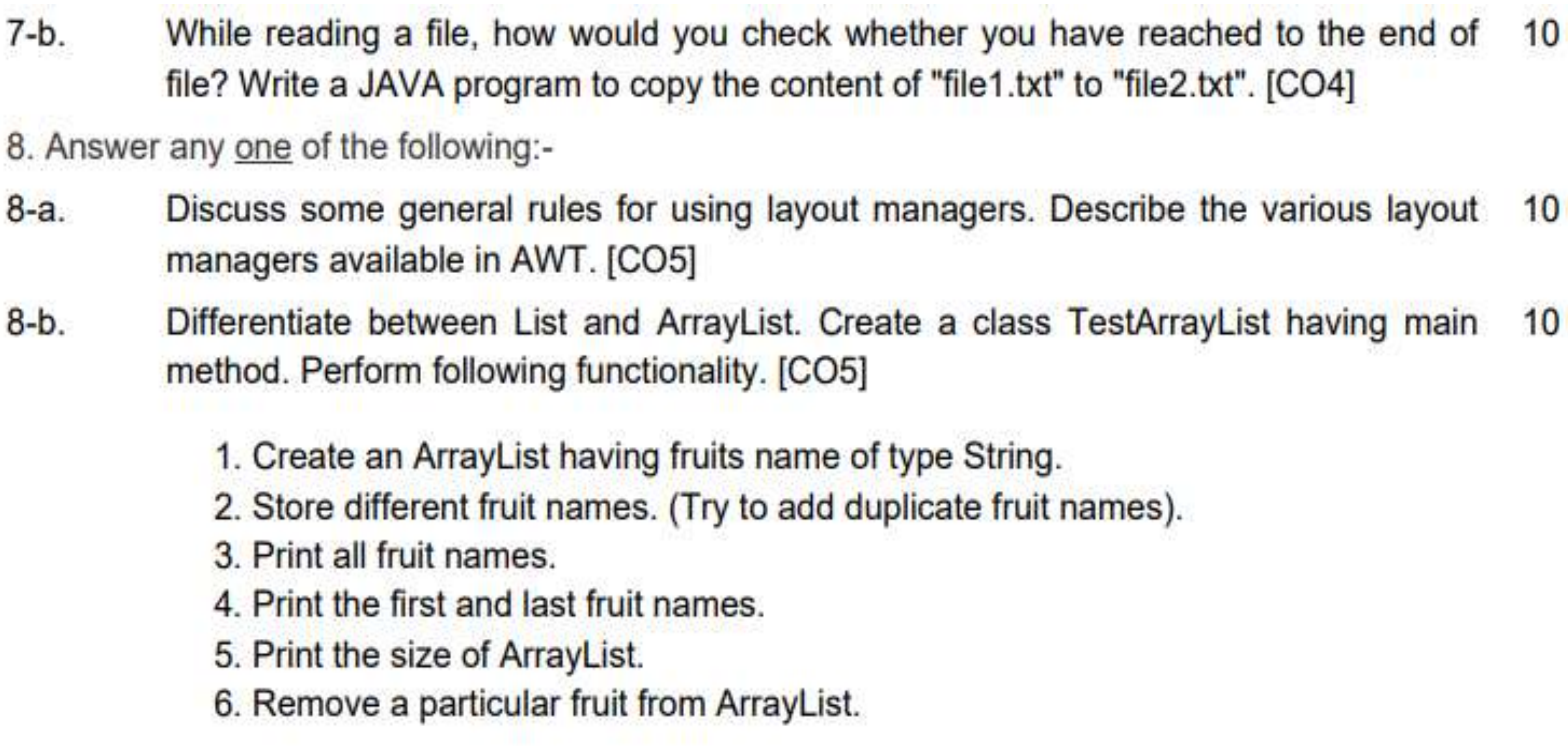


	ArithmeticException with a message "Student is not eligible for registration" when age < 12 and weight < 40, otherwise it prints "Student Entry is Valid!!". [CO3]	

7. Answer any one of the following:-

7-a.	What is the difference between thread and a process? Explain the concept of Inter Thread Communication and describe the role of wait(), notify(), and notifyAll() methods in inter thread communication. [CO4]	10

Old University Question Paper



7-b.	While reading a file, how would you check whether you have reached to the end of file? Write a JAVA program to copy the content of "file1.txt" to "file2.txt". [CO4]	10

8. Answer any one of the following:-

8-a.	Discuss some general rules for using layout managers. Describe the various layout managers available in AWT. [CO5]	10
8-b.	Differentiate between List and ArrayList. Create a class TestArrayList having main method. Perform following functionality. [CO5]	10

1. Create an ArrayList having fruits name of type String.
2. Store different fruit names. (Try to add duplicate fruit names).
3. Print all fruit names.
4. Print the first and last fruit names.
5. Print the size of ArrayList.
6. Remove a particular fruit from ArrayList.

Expected Questions

1) What is the difference between Collection and Collections? [CO5]
2) What is the difference between Comparable and Comparator? [CO5]
3) What is the advantage of the generic collection? [CO5]
4) What is the difference between Array and ArrayList? [CO5]
5) How to reverse ArrayList? [CO5]
6) What is a Container Class? [CO5]
7) What is GUI in Java? [CO5]
8) Give differences between Collection and Collections. [CO5]
9) Differentiate between Comparable and Comparator. [CO5]
10) Discuss the advantage of the generic collection. [CO5]
11) Distinguish between Array and ArrayList. [CO5]
12) How to reverse ArrayList? [CO5]
13) Discuss the use of Container Class with an example. [CO5]
14) Elaborate the significance of GUI in Java. [CO5]

Old Question Papers

- https://www.iare.ac.in/sites/default/files/IARE_JAVA_MODEL_QP.pdf
- https://www.manaresults.co.in/jntuh/download.php?subcode=133BM

Recap of Unit

- GUI Programming in Java.
- Applets and Lifecycle of Applet.
- AWT, components and containers.
- Event Handling.
- **Generics and Collections: Wrapper classes**

References

Text Books:
(1) Herbert Schildt," Java - The Complete Reference", McGraw Hill Education 12th edition
(2) Herbert Schildt," Java: A Beginner's Guide", McGraw-Hill Education 2nd edition
(3) James Rumbaugh et. al, "Object Oriented Modeling and Design", PHI 2nd Edition
Reference Books:
(4) Cay S. Horstmann, "Core Java Volume I – Fundamentals", Prentice Hall
(5) Joshua Bloch," Effective Java", Addison Wesley
(6) E Balagurusamy, "Programming with Java A Primer", TMH, 4th edition.

Thank You