COMSATS University Islamabad Abbottabad Campus
Department of Computer Science

Course: Data Structure Class: BSE-4D
Instructor: N a u m a n Khan Marks: 20 marks

Instructions:

1. Must be submitted at CUOnline only.
2. Last date of submission is **10-09-2025.** Must be submitted inside lab
3. Copying other's assignments will leave a negative impression. Students are advised not to copy material from others.

Lab Task 02 – Java Basics Practice (cont.…)

Objective:

- Practice while and for loops with break and continue.
- Learn input validation using loops.
- Implement and use methods with parameters and return values.
- Apply method overloading.
- Work with arrays to solve problems.

Q1: Pizza Billing System

Instruction

- Ask user for pizza size (small, medium, large) → validate using while(true) until correct.
- Assign base price: small=100, medium=200, large=300.
- Ask if user wants pepperoni: small=+30, medium/large=+50.
- Ask if user wants extra cheese: +20.
- Use continue for invalid inputs, break when valid.
- Show final bill.

Sample Run:

Enter pizza size: mini
Invalid! Try again.
Enter pizza size: small
Do you want pepperoni? yes
Do you want extra cheese? no
Your final bill is: 130 rupees

Q2: Array Analyzer

Instructions:

- Input N numbers into an array.
- Calculate: sum, average, min, max, count of evens/odds.
- Print array in reverse order.

Q3: Utility Method Pack

Instructions:

Implement following methods:

- int add (int a, int b)
- int subtract (int a, int b)
- long multiply (int a, int b)
- double divide (int a, int b) → guard against divide by zero.
- int maxofThree (int a, int b, int c)
- boolean isPrime(int n)

Demonstrate each method with user input.

Q4: Method Overloading: Area Calculator

Instructions:

Implement overloaded methods:

- double area (double radius) → circle
- double area (double length, double width) → rectangle
- double area(double side) → square

Q5: Mini Projects

Project 1: ATM Simulation System

Instructions:

- Predefined PIN = 1234. User has 3 attempts using a for loop.
- If wrong 3 times → locked.
- After login, show menu (while loop):
  a) Deposit
  b) Withdraw
  c) Check Balance
  d) Exit
- Implement methods:
  a) deposit (int amount) → balance increases if amount > 0.
  b) withdraw (int amount) → decreases balance if enough funds.
  c) checkBalance () → prints current balance.
    - Use input validation (no negative deposits/withdrawals).
    - Use continue for invalid options, break to exit.

Sample Result
Enter PIN: 0000
Wrong PIN! Attempts left: 2
Enter PIN: 1234
Login successful!

==== ATM Menu ====
1) Deposit
2) Withdraw
3) Check Balance
4) Exit

Choice: 1
Enter amount to deposit: -50
Invalid amount! Try again.

Choice: 1
Enter amount to deposit: 200
Deposit successful.

Choice: 2
Enter amount to withdraw: 500
Insufficient balance!

Choice: 3
Your balance is: 200

Choice: 4
Thank you for using the ATM. Goodbye!

Project 2: Student Gradebook Manager

Instructions:

- Manage up to 50 students using arrays.
- Store: roll[], name[], marks[].
- Menu (while loop):
  1) Add Student
  2) Display All
  3) Search Student (by Roll / by Name)
  4) Class Average & Topper
  5) Exit
- Methods:
  a) addStudent(int roll, String name, int marks)
  b) displayAll()
  c) search(int roll)
  d) search(String name) → Method overloading
  e) computeAverage(int[] marks, int count)
  f) grade(int marks) → return A/B/C/F
    - Input validation for marks (0–100).
    - Use continue to reject invalid inputs.
    - Use break to exit search loop early.

Sample Run 1

==== Student Gradebook Manager ====
1) Add Student
2) Display All
3) Search Student (by Roll / by Name)
4) Class Average & Topper
5) Exit
Choice: 1

Enter Roll: 12
Enter Name: Ali

Enter Marks (0-100): -5
Invalid marks! Please enter a value between 0 and 100.
Enter Marks (0-100): 92
Student added successfully.

Choice: 1
Enter Roll: 7
Enter Name: Sara
Enter Marks (0-100): 76
Student added successfully.

Choice: 1
Enter Roll: 19
Enter Name: Hassan
Enter Marks (0-100): 88
Student added successfully.

Choice: 1
Enter Roll: 5
Enter Name: Fatima
Enter Marks (0-100): 59
Student added successfully.

Choice: 1
Enter Roll: 3
Enter Name: Ahsan
Enter Marks (0-100): 100
Student added successfully.

Choice: 2

Sample Run 2 — Display All
(Assume grade rule: **A ≥ 85**, **B 70–84**, **C 50–69**, **F < 50**)

Roll	Name	Marks	Grade
12	Ali	92	A
7	Sara	76	B
19	Hassan	88	A
5	Fatima	59	C
3	Ahsan	100	A

Total Students: 5

Sample Run 3 — Search by Roll (int) and by Name (String)
(Shows **overloaded search** + early exit with break)
==== Student Gradebook Manager ====
1) Add Student
2) Display All
3) Search Student (by Roll / by Name)
4) Class Average & Topper
5) Exit

Choice: 3

Search by: 1) Roll 2) Name
1
Enter Roll to search: 19
Found:
Roll: 19
Name: Hassan
Marks: 88
Grade: A

Choice: 3
Search by: 1) Roll 2) Name
2
Enter Name to search: fatima
Found:
Roll: 5
Name: Fatima
Marks: 59
Grade: C

Choice: 3
Search by: 1) Roll 2) Name
1
Enter Roll to search: 99
No record found.
(Press any key to return to menu)
If you implement search with a loop over current size, you can **break** immediately after finding a match.

Sample Run 4 — Class Average & Topper
(Using the same 5 students above: 92, 76, 88, 59, 100)
==== Class Average & Topper ====
Class Average: 83.00
Topper:
Roll: 3
Name: Ahsan
Marks: 100
Grade: A
(Check: 92 + 76 + 88 + 59 + 100 = 415; 415 / 5 = 83.0)

Sample Run 5 — Exit
==== Student Gradebook Manager ====
1) Add Student
2) Display All
3) Search Student (by Roll / by Name)
4) Class Average & Topper
5) Exit
Choice: 5
Goodbye!
Student added successfully.