THE STATE UNIVERSITY OF ZANZIBAR (SUZA)
School of Computing Communication and Media Studies (SCCMS)
Department of Computer Science and IT

IT6003 – Advanced Programming Using Java

Programming Assignment – CycleSheet I

Programme:	MSc. Information Technology	**Total Questions:**	12
Semester:	I	**Total Marks:**	100

Submission: Submit `.java` files with proper naming convention (e.g., `Q1_StudentDetails.java`). Submit all files in a single ZIP folder named: `YourRegNo_CycleSheet1.zip`

General Instructions:

1. Each program must compile and run without errors.
2. Include appropriate comments explaining your logic.
3. Follow Java naming conventions (camelCase for methods/variables, PascalCase for classes).
4. Test your programs with multiple inputs before submission.

AI Usage Policy & Academic Integrity:

The following regulations govern the use of Artificial Intelligence (AI) tools in this assignment:

1. **AI-Assisted Work Declaration:** Students must declare any use of AI tools (ChatGPT, GitHub Copilot, Claude, Gemini, etc.) in their submissions.
2. **Understanding Requirement:** Students must be able to explain every line of code they submit. Random viva/oral examination may be conducted.
3. **Prohibited Actions:**
   - Directly copying AI-generated code without understanding
   - Submitting AI-generated solutions as entirely original work
   - Using AI to complete the entire assignment without learning

-2

4. **Permitted Use:**
   - Using AI to understand concepts and syntax
   - Debugging assistance with proper acknowledgment
   - Learning from AI explanations to write your own code
5. **Plagiarism Detection:** Submissions will be checked using plagiarism detection tools including AI-content detectors.
6. **Penalties:** Violations may result in zero marks, disciplinary action, or failure in the course as per university regulations.

By submitting this assignment, you confirm that the work is substantially your own and you can explain all code submitted.

Section A: Basic Program

Question 1: Student Information System [8 Marks]

Design a Java program to read and display the following details of `n` students using `Scanner` class methods:

- Registration number (String)
- Name (String that may contain first name, middle name, and last name)
- CGPA (Floating point number)
- Programme Name (String)
- School Name (String with multiple words)

- Proctor Name (String that may contain first, middle, and last names)

Requirements:

a) Use appropriate Scanner methods (`nextLine()`, `nextFloat()`, `nextInt()`) correctly.
b) Handle the newline character issue when mixing `nextInt()`/`nextFloat()` with `nextLine()`.
c) Display all student details in a formatted manner after input.

Section B: Arrays

Question 2: Airline Reservation System [10 Marks]

A small airline has just purchased a computer for its new automated reservations system. Develop the new system with the following specifications:

The airline has only one plane with a capacity of 10 seats:

- Seats 1–5: First Class
- Seats 6–10: Economy Class

Requirements:

a) Display menu: "Please type 1 for First Class" and "Please type 2 for Economy".
b) Use a one-dimensional boolean array to represent the seating chart (`false` = empty, `true` = occupied).
c) When a seat is assigned, display a boarding pass showing seat number and section.
d) Never assign an already occupied seat.
e) When the preferred section is full, offer alternative section. If declined, display: "Next flight leaves in 3 hours."

Question 3: Upper Triangular Matrix Checker [8 Marks]

Implement a program using the **labeled break** concept to check if a given square matrix is an Upper Triangular Matrix.

Definition: An upper triangular matrix is one whose entries below the main diagonal are all zero.

Example: $\begin{pmatrix} 1 & 3 & 5 \\ 0 & 9 & 4 \\ 0 & 0 & 8 \end{pmatrix}$ is upper triangular.

Requirements:

a) Read the size and elements of the square matrix from user.
b) Use labeled break to immediately terminate checking when a non-zero element is found below the diagonal.
c) Display appropriate message indicating whether the matrix is upper triangular or not.

Question 4: Course Preference Allocation System [8 Marks]

The following table contains course codes to be offered next semester:

ITA1001	ITA1002	ITA1003	ITA1004	ITA1005	ITA1006
ITA2001	ITA2002	ITA2003	ITA2004	ITA2005	ITA2006
ITE1001	ITE1002	ITE1003	ITE1004	ITE1005	ITE1006
ITE2001	ITE2002	ITE2003	ITE2004	ITE2005	ITE2006
SWE1001	SWE1002	SWE1003	SWE1004	SWE1005	SWE1006
SWE2001	SWE2002	SWE2003	SWE2004	SWE2005	SWE2006

Requirements:

a) Accept up to 5 course preferences via command line arguments.
b) If no preferences provided, randomly select 5 courses from the table.
c) If fewer than 5 preferences given, fill remaining slots randomly.
d) Display the final 5 preferences.

Hint: Generate random numbers 0–35. For number n: row = $n/6$, column = $n\%6$. Example: 17 → row=2, col=5 → ITE1006.

Section C: String Processing

Question 5: Hash Algorithm Implementation [8 Marks]

Implement a hash algorithm that uses rotation and fold shift methods to compute a storage address.

Algorithm Steps:

1. **Rotation:** Move the least significant digit to the most significant position.
2. **Fold Shift:** Divide rotated data into segments of length 2, sum all segments.
3. If sum has more than 2 digits, delete the most significant digit.

Example: Input: 112286 → After rotation: 611228 → Fold shift: 61+12+28=101 → Final: 01

Requirements:

a) Define rotation as a `static` method.
b) Define fold shift as a non-static method.
c) Invoke both methods appropriately from `main()`.

Question 6: Format Specifier Syntax Validator [8 Marks]

Write a program to verify the syntax correctness of `System.out.format()` statements by checking:

Validation Rules:

a) The number of format specifiers must match the number of arguments.
b) The datatype of arguments must match their corresponding format specifiers.

Given variable table:

a	b	s	x
int	float	String	int

Test Cases:

i) `System.out.format("sum is %d"+"avg is %f ", a, b);` → Correct syntax
ii) `System.out.format("name is %s"+"sum is %d avg is %f ", s, a, b);` → Correct syntax
iii) `System.out.format("sum is %d"+"avg is %f ", b, a);` → Wrong syntax

Section D: Classes and Objects

Question 7: Vector Class Implementation [10 Marks]

Define a `Vector` class (not the predefined Java class) with the following specifications:

Instance Variables:

- `x`: magnitude along the x-direction
- `y`: magnitude along the y-direction
- `z`: magnitude along the z-direction

Constructors:

i) Default constructor: Assign 1 for all three components.
ii) Constructor with `x`, `y` parameters for 2D vectors.
iii) Constructor with `x`, `y`, `z` parameters (must reuse code from constructor ii using `this()`).

Methods:

- `dotProduct()`: Compute dot product for 2D vectors. If $\mathbf{U} = a\mathbf{i} + b\mathbf{j}$ and $\mathbf{V} = c\mathbf{i} + d\mathbf{j}$, then $\mathbf{U} \cdot \mathbf{V} = ac + bd$
- Overloaded `dotProduct()`: For 3D vectors. If $\mathbf{U} = a\mathbf{i} + b\mathbf{j} + c\mathbf{k}$ and $\mathbf{V} = d\mathbf{i} + e\mathbf{j} + f\mathbf{k}$, then $\mathbf{U} \cdot \mathbf{V} = ad + be + cf$
- `computeAngle()`: Find angle $\theta = \cos^{-1}\left(\frac{\mathbf{U}\cdot\mathbf{V}}{|\mathbf{U}||\mathbf{V}|}\right)$ where $|\mathbf{U}| = \sqrt{a^2 + b^2 + c^2}$

Create objects `v1`, `v2` (2D) and `v3`, `v4` (3D) to demonstrate all functionality.

Hint: Use `Math.acos()` for inverse cosine.

Question 8: Math Premier League (MPL) Examination System [8 Marks]

Students from Class I to X appear for the MPL examination. Design a class with:

Data Members:

- `standard`: Class standard (I to X)
- `numberOfStudents`: Strength of the class
- `marks[]`: Array to store scores (dynamically sized based on class strength)
- `firstMark`: Highest mark in the class

Requirements:

a) Create an array of 4 MPL objects.
b) Parameterized constructor receives `standard` and `numberOfStudents`, then reads marks and finds `firstMark`.
c) Define `findBestClass()` to display the class with highest first mark.
d) Overload `findBestClass()` to display the class with highest average.

Section E: Inheritance and Interfaces

Question 9: Training Centre Student Tracking System [8 Marks]

A training centre conducts 7 tests. Students can skip tests. There are 25 students in the batch.

Class `TestDetails`:

- 2D float array to store marks (rows = students, columns = tests taken)
- `storeMarks()`: Receive student details and create dynamic columns based on tests taken
- `displayMarks()`: Print all student marks

Class `NoticePeriod` (extends `TestDetails`):

Students are placed in notice period if they:

- Have taken fewer than 3 tests, **OR**
- Have not scored ≥ 50 in at least 3 tests

Requirements:

a) Count and print students in notice period.
b) Print IDs of students in notice period (use array row index as ID).
c) Optimization: Stop checking marks once student has passed 3 tests with ≥ 50.

Question 10: GCD Calculator using Interface [8 Marks]

Define an interface `GCD` with abstract method: `computeGCD(int num1, int num2)`

Implement the interface in two classes:

APPROACH1 — Euclid's Algorithm:

Repeatedly divide larger by smaller until remainder is 0.
Example: GCD(48, 18): $48\%18 = 12 \rightarrow 18\%12 = 6 \rightarrow 12\%6 = 0 \rightarrow$ GCD=6

APPROACH2 — Factor Listing Method:

List all factors of both numbers and find the highest common factor.
Example: Factors of 48: 1,2,3,4,6,8,12,16,24,48 | Factors of 18: 1,2,3,6,9,18 | HCF=6

Section F: Exception Handling

Question 11: Ball Picking Simulation [8 Marks]

A bag contains balls of 4 colors: Red, Green, Blue, and Yellow.

Requirements:

a) Simulate randomly picking a ball 10 times.
b) Create a custom exception: `SameColorBallException`
c) If the same color is picked more than 3 times, throw the exception.
d) When exception occurs, restart the simulation from the beginning.
e) After 10 valid picks, print the count of each color picked.

Hint: Use `Random` class to generate numbers 0–3 representing the 4 colors.

Section G: Multithreading

Question 12: Statistical Mean Calculator [8 Marks]

Calculate the mean of the following marks distribution using multithreading:

Marks (x_i)	1	2	3	4	5	6	7	8	9	10
Count (f_i)	3	4	17	8	23	10	4	6	5	2

Formula: Mean $= \dfrac{\sum(f_i \times x_i)}{\sum f_i}$

Requirements:

a) Thread 1: Calculate $\sum(f_i \times x_i)$ — sum of (frequency × marks)
b) Thread 2: Calculate $\sum f_i$ — sum of frequencies
c) Main Thread: Wait for both threads to complete, then calculate and display the mean.

Hint: Use `thread.join()` to ensure main thread waits for worker threads to complete.

Marking Criteria

Criteria	Weight	Description
Program Compilation	15%	Code compiles without errors
Correctness & Output	30%	Produces expected output for all test cases
Concept Implementation	25%	Proper use of required OOP concepts
Code Quality	15%	Comments, naming conventions, formatting
Edge Cases & Validation	10%	Handles invalid inputs gracefully
Originality & Understanding	5%	Demonstrates understanding during viva
TOTAL	**100%**	

Marks Distribution by Question:

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Total
8	10	8	8	8	8	10	8	8	8	8	8	**100**

Good luck with your assignment!
For queries, contact your course instructor.