NATIONAL UNIVERSITY OF SCIENCE AND TECHNOLOGY
FACULTY OF APPLIED SCIENCE
COMPUTER SCIENCE DEPARTMENT
DECEMBER EXAMINATIONS 2004

SUBJECT: OBJECT ORIENTED PROGRAMMING I
CODE: SCS5101

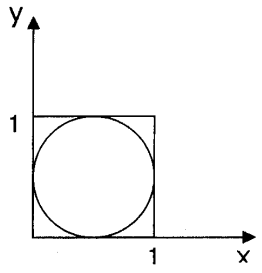
INSTRUCTION TO CANDIDATES

Answer any 5 questions
All questions carry equal marks
Total marks 100

Time: 3 hours

QUESTION ONE

Java includes a pseudo-random number method. It is possible to use this method to obtain an approximation to π. The procedure involves generating a random point in a unit square (by generating two random numbers between 0 and 1), and determining whether this point is inside or outside the circle of diameter 1 with center at $(1/2, 1/2)$. This process is repeated for a large number of points (e.g., 5000). The proportion of 'hits' (that is, points lying inside or on the circle) to the total number of attempts will be roughly equal to the ratio of the area of the circle to that of the square (that is, $\pi/4$). Use this technique in a program for approximating π.



[20]

QUESTION TWO

Write a Java Swing GUI to store the details of a small collection of Student objects using an array. Include the following instance fields for each student: Name, discipline, year, gender and date of birth. Your application should be capable of:

(a) Appending a new student to the collection, [6]
(b) Displaying the student collection by listing the details of the student [6]
(c) Deleting a student from the collection. [8]

All changes to the collection occur as a result of choosing an appropriate menu item from the GUI.

QUESTION THREE

Write a Java application that accepts as input two $n \times n$ matrices A and B. The application should produce the following as outputs:

(a) The matrix sum: $A + B = C$, where $[c_{ij}]_{n \times n} = [a_{ij} + b_{ij}]_{n \times n}$. [6]

(b) The matrix difference: $A - B = D$, where $[\mathrm{d}_{ij}]_{n \times n} = [a_{ij} - b_{ij}]_{n \times n}$ [6]

(c) The matrix product: $AB = P$, where $[\mathrm{p}_{ij}]_{n \times n} = [\sum_{k=1}^{n} a_{ik} b_{kj}]_{n \times n}$ [8]

QUESTION FOUR

Design a set of classes to represent different types of regular two-dimensional shapes. Each concrete class should encapsulate a method to calculate the area of the shape, and fields that represent the dimensions of the shape. Implement a Java program that instantiates each class and tests whether each method works correctly.

[20]

QUESTION FIVE

Define and explain the following:

Interface	[3]
Inheritance	[3]
Class	[3]
Object	[2]
Polymorphism	[3]
Encapsulation	[2]
Overloading	[2]
Overriding	[2]

QUESTION SIX

Write a Java Swing GUI that represents a small calculator. The calculator should operate like a standard non-scientific calculator, but should only incorporate basic arithmetic operations such as addition, subtraction, multiplication and division.

[20]

QUESTION SEVEN

Write a Java program that accepts as input an integer n. The program should print out the following:

If n = 3

```
  *
***
  *
```

If n = 5

```
   *
  ***
*****
  ***
   *
```

[20]

END OF QUESTION PAPER

GOOD LUCK!