COURSE DESCRIPTION:
By the end of the Course, Students should be able to do the following in simple cases:
Given a Problem, devise a Problem Representation and an Algorithm that solves the
Problem. Write a Java Program that Implements this using – Basic Java Language
Elements – Data Types (Primitive and Object type) – Input/output Statements – Control
Structures for Decisions and Iteration – Exception Handling. Understand and use
Inheritance and Polymorphism. Understand and use Recursive Methods.

PREREQUISITE:

TEXTBOOK:
“Java Software Solutions: Foundation of Program,” (9th edition), by Lewis, published by
Pearson.

DEPARTMENT WEBSITE:  http://www.ncas.rutgers.edu/math

THIS COURSE COVERS THE FOLLOWING TOPICS:

Items in Square Brackets are relevant Chapters of the Textbook.

• Basic Instruction on Downloading Java and using a Development Environment.

• Introduction to Java [Chapter 1]. Broad Introduction and Review of Programming
  concepts; Java Program Structure.

• Data and Expressions [Chapter 2]. Character Strings, Variables, Assignment,
  Constants, Primitive Types, The Scanner Class.

• Using classes and objects [Chapter 3]. Using the String, Math, Random
  and DecimalFormat classes.

• Writing classes [Chapter 4]. Encapsulation, constructors, instance data,
accessors and mutators, methods.

- **Conditionals and loops** [Chapters 5 and 6]. Boolean expressions, if, if-else, while, for (including for each), iterators, the ArrayList class, do, switch.

- **Object-oriented design** [Chapter 7]. class relationships, interfaces, method overloading, testing and debugging.

- **Arrays** [Chapter 8]. Declaring and using arrays, arrays of objects, array bounds, initializers, arrays as parameters, command-line arguments.

- **Inheritance** [Chapter 9]. Creating subclasses, protected, super, overriding methods, class hierarchies.

- **Polymorphism** [Chapter 10]. Late binding, polymorphism via inheritance and interfaces, application to searching and sorting.

- **Exceptions** [Chapter 11]. Caught and uncaught exceptions, Exception class, try/catch/finally, I/O exceptions.

- **Recursion** [Chapter 12]. Idea of recursion, recursive programming, simple examples.