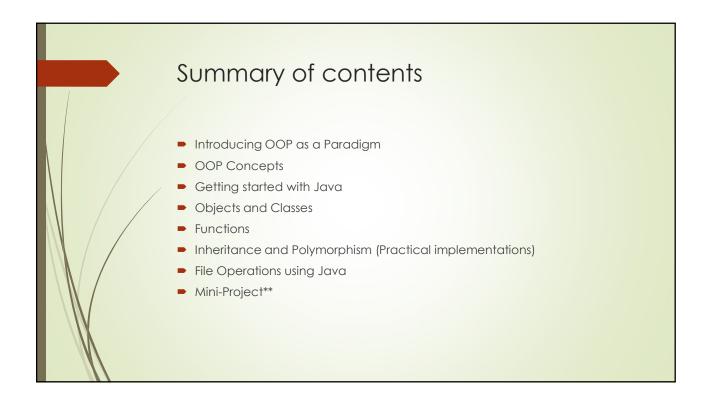
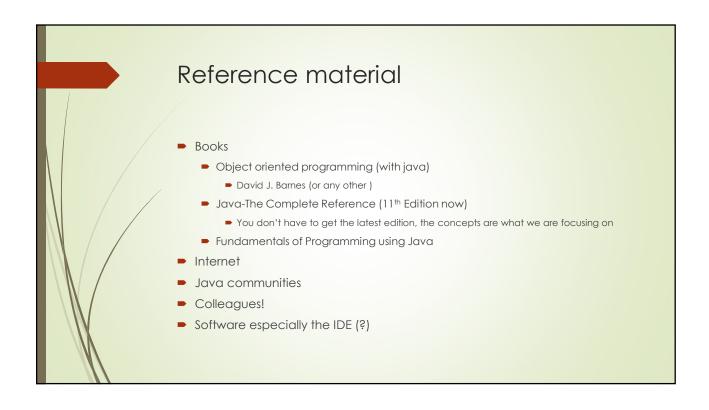


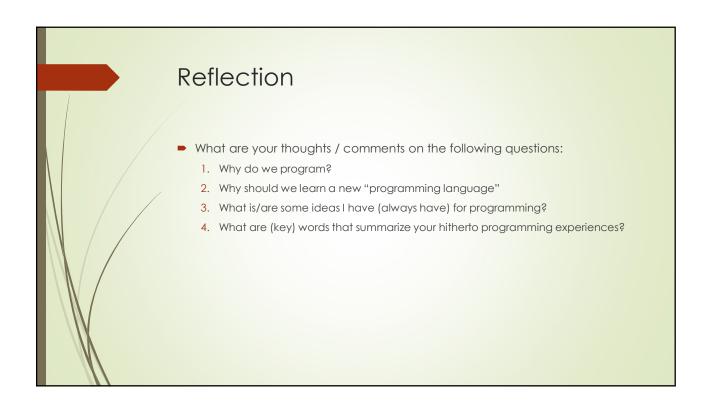
The Course will be offered in a Blended version: Online and face to face Lecturer: Prof. Salesio M, Kiura Consultations: D21, timing: TBA Moodle link: https://elearning.tukenya.ac.ke/ Enrollment Key: OOP2019 Delivery: Lectures Exercises / Activities Assignments, Assessment tests Timing: D23: Monday 7-9, 9-11 / 11-1, 2-4 | | online?: Thursday 7-9

The course will be delivered in a learner-centered way Participants introductions Names, Experiences with programming, prospects or plans in the field Suggestions / Recommendations for the course Expect Participative engagements Learner - centered learning This is a Practical course – practice, practice, practice! Take your time to learn Programming (through practice)

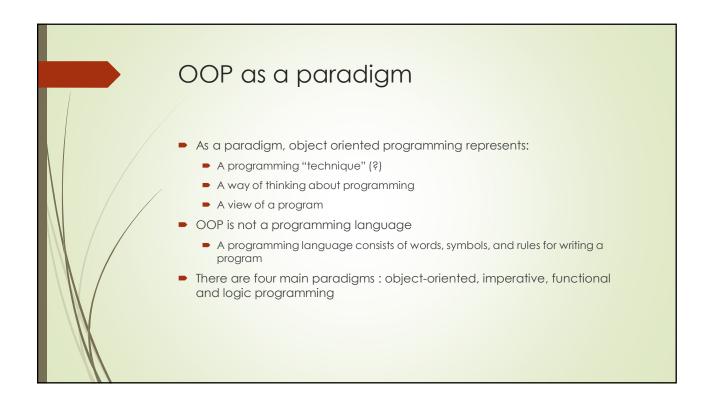
Course objectives At the end of the course, the student should be able to: Demonstrate an in-depth understanding of Object Oriented paradigm and concepts Apply object oriented concepts using a selected language (Java) Implement principles of inheritance, exception handling, abstract classes, packages, etc. Analyze application scenarios (for) and design software systems using object oriented analysis and design.



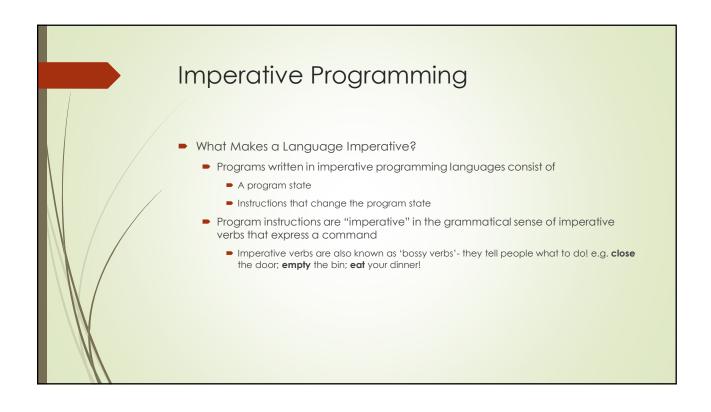




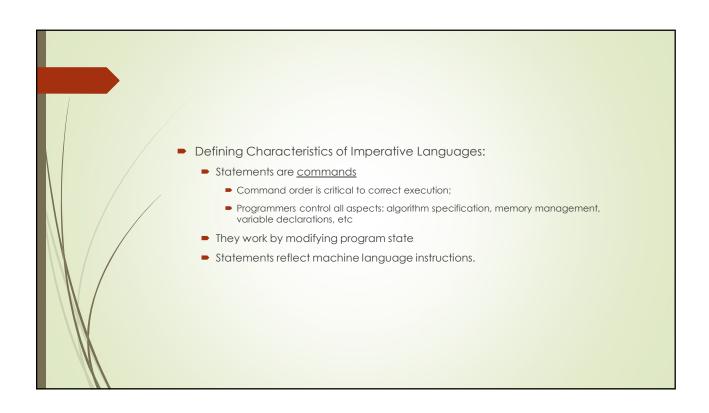
Paradigms What is a paradigm? In science, paradigm describes distinct concepts or thought patterns in any scientific discipline or other epistemological context A programming paradigm is a fundamental style of computer programming Programming Paradigm A way of conceptualizing what it means to perform computation and how tasks to be carried out on the computer should be structured and organized. Computation refers to Implementation of an I/O relation

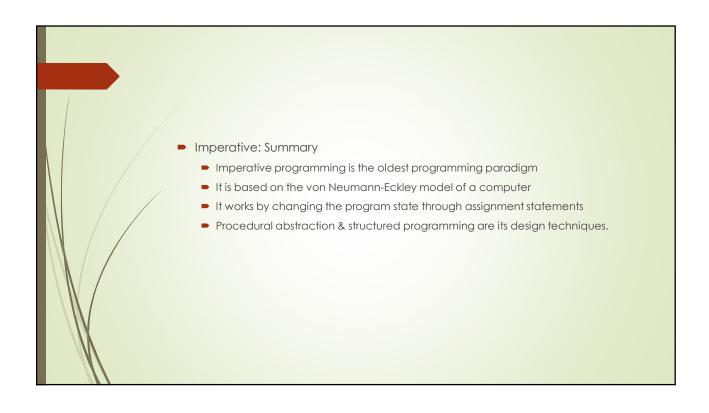


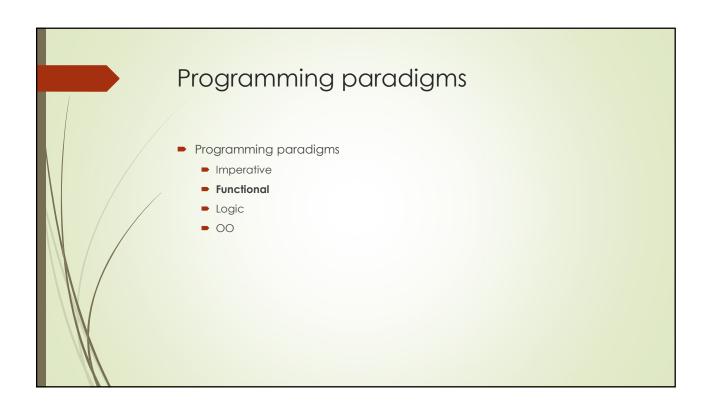
Programming paradigms Imperative Programming program as a collection of statements and procedures affecting data (variables) Object-Oriented Programming program as a collection of classes for interacting objects Functional Programming program as a collection of (math) functions Logic programming Program as a collection of logical sentences

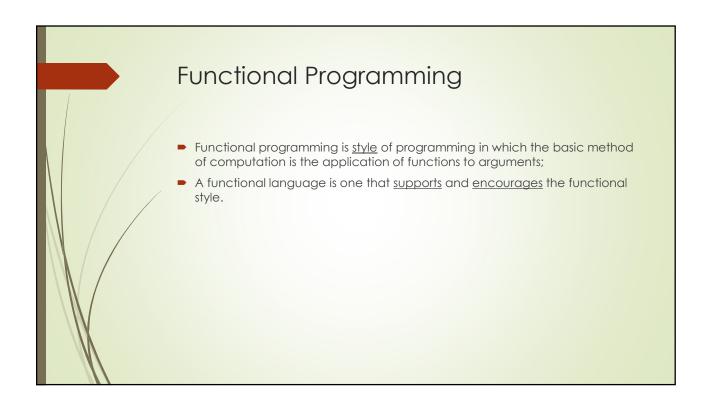


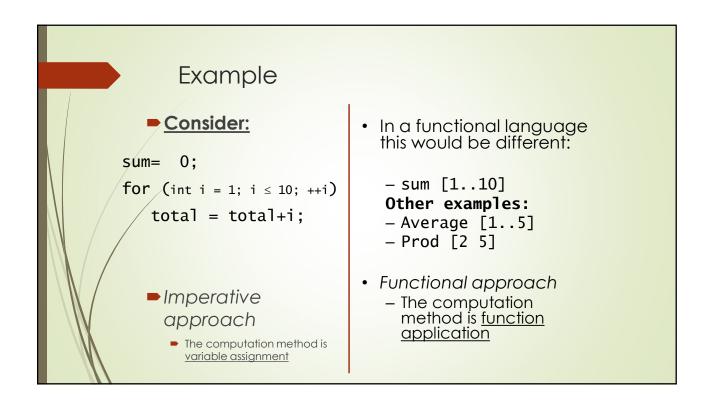
Imperative Languages Commands in an imperative language are similar to the native machine instructions of traditional computer hardware – the von Neumann-Eckley model. Some old Imperative Languages assembly languages 1954-1955: Fortran (FORmula TRANslator) Late 1950's: Algol (ALGOrithmic Language) 1958: Cobol (COmmon Business Oriented Language)

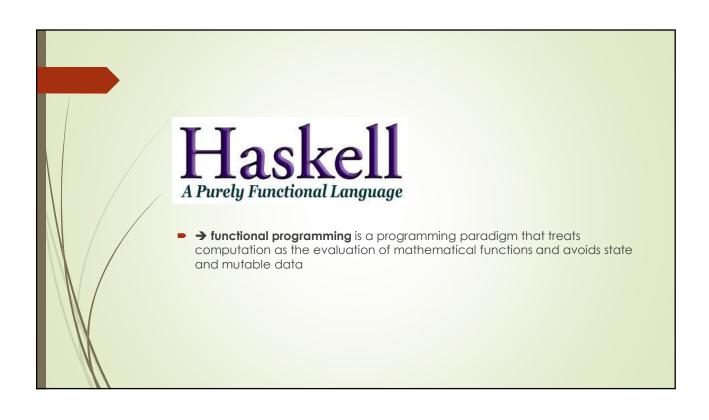


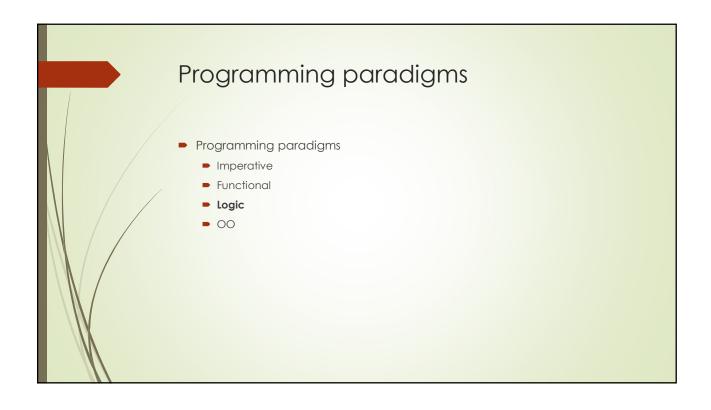


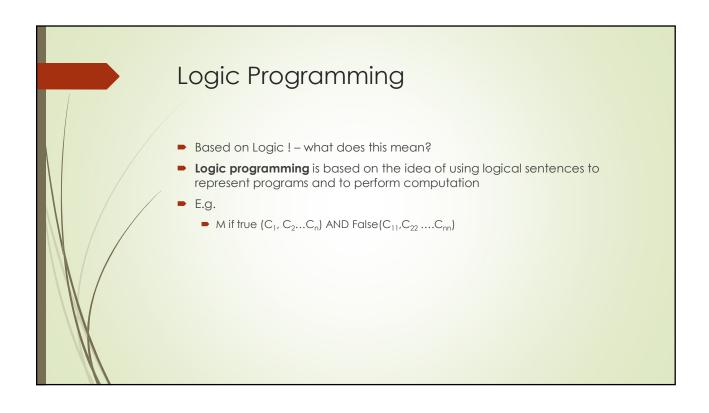


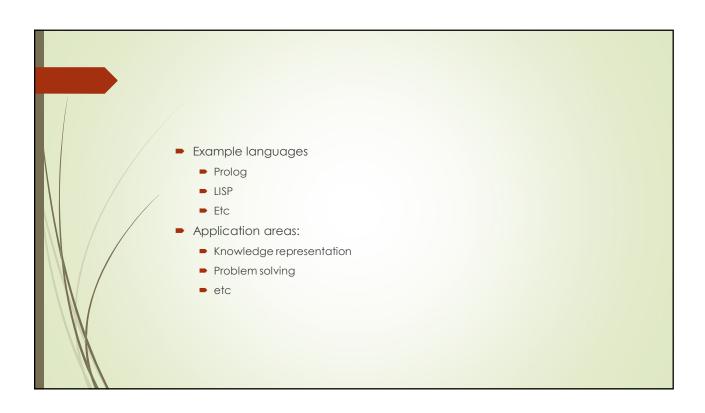


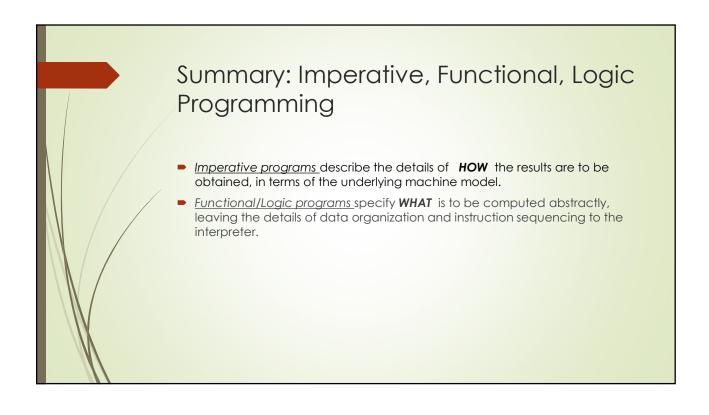


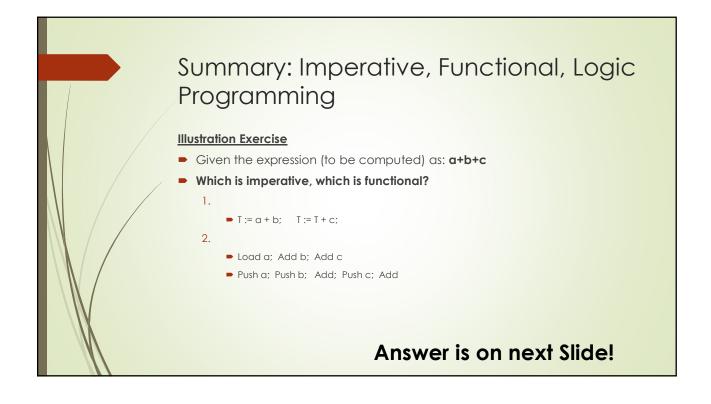




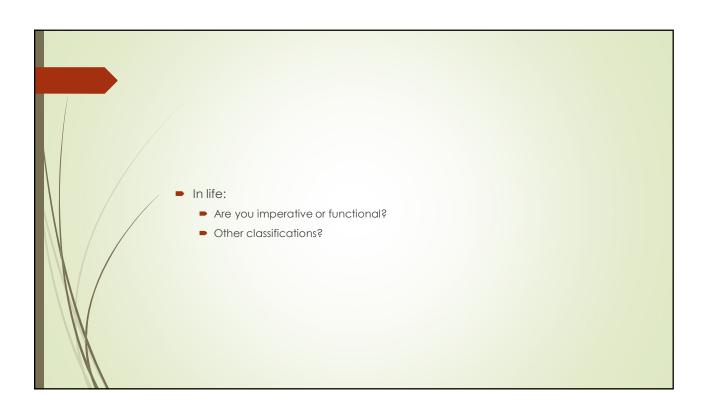


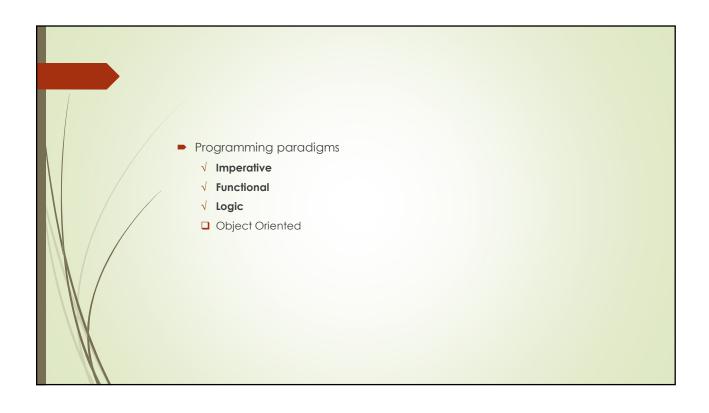


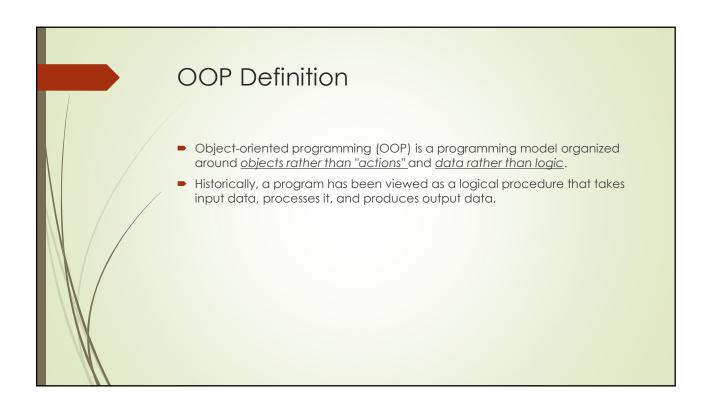


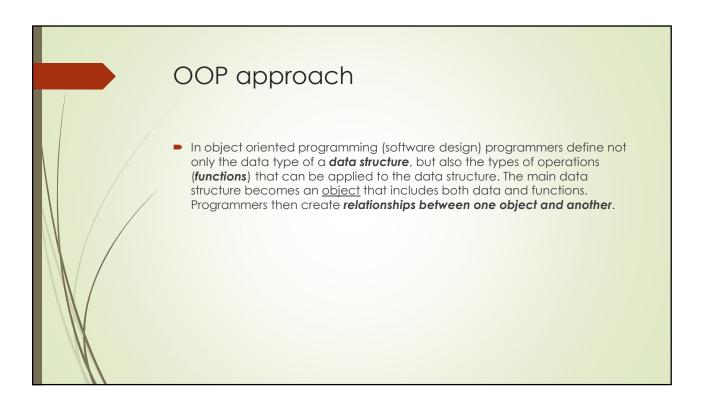


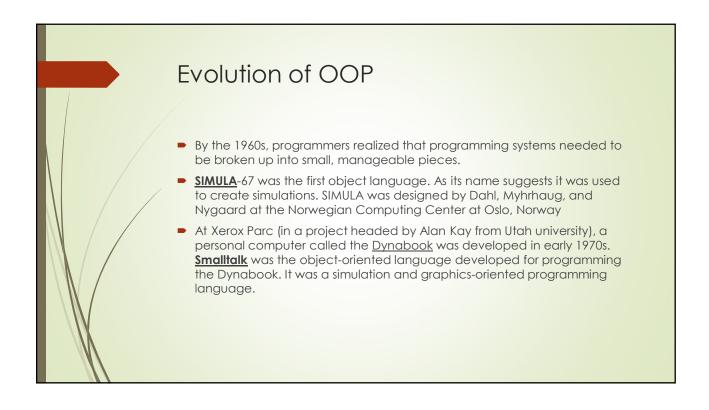




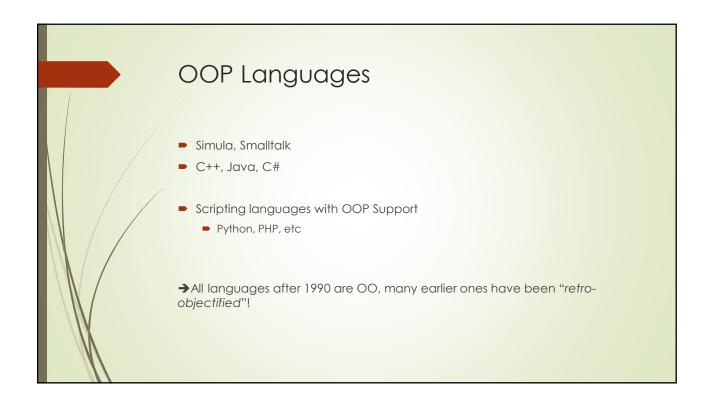




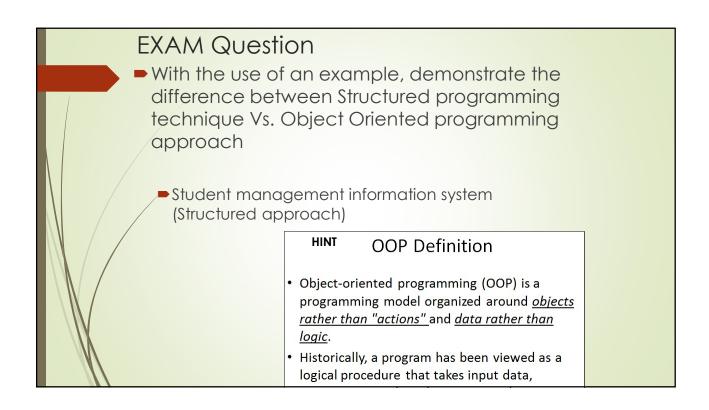


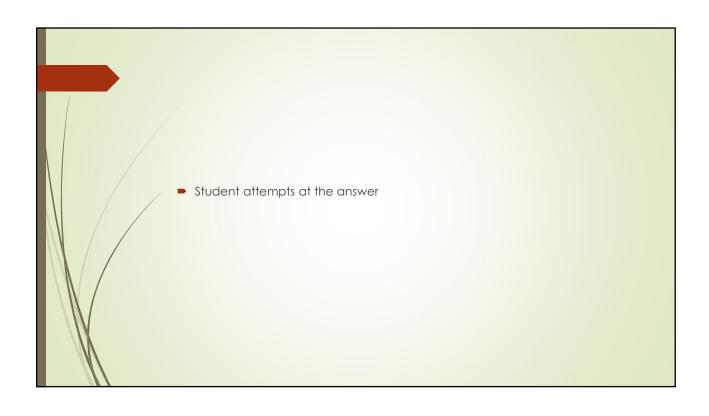


Object-oriented programming gained momentum in the 1970s and in the early 1980s. Bjorn Stroustrup integrated object-oriented programming into the C language. The resulting language was called C++ and it became the first object-oriented language to be widely used commercially. In the early 1990s, at Sun Microsystems a simpler version of C++ was developed called Java that was meant to be a programming language for video-on-demand applications. This project was re-oriented to focus on (and was marketed as being the language for) programming Internet applications. The language gained widespread popularity as the Internet has boomed, although its market penetration has been limited by its (earlier) inefficiency.









Procedural vs Object-Oriented programming Procedural: Emphasis on procedural abstraction. Top-down design; Step-wise refinement. Suited for programming in the small. Object oriented Emphasis on data abstraction. Bottom-up design; Reusable libraries. Suited for programming in the large.

