RAMA UNIVERSITY

www.ramauniversity.ac.in

FACULTY OF ENGINEERING & TECHNOLOGY

Brajesh Mishra
Assistant Professor
Department of Computer Science & Engineering

Topics Covered

Benefits of Encapsulation
Storage Management
Static Storage Allocation

Benefits of Encapsulation

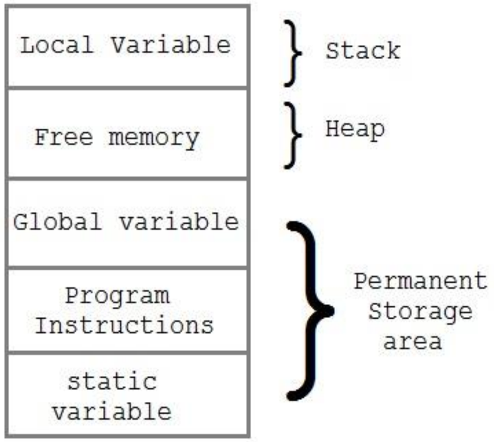
- **Data Hiding:**
  - The user will have no idea about the inner implementation of the class.
  - It will not be visible to the user that how the class is storing values in the variables.
  - He only knows that we are passing the values to a setter method and variables are getting initialized with that value.
- **Increased Flexibility:**
  - We can make the variables of the class as read-only or write-only depending on our requirement.
  - If we wish to make the variables as read-only then we have to omit the setter methods like setName(), setAge() etc. from the above program or if we wish to make the variables as write-only then we have to omit the get methods like getName(), getAge() etc.
- **Reusability:**
  - Encapsulation also improves the re-usability and easy to change with new requirements
- **Testing code is easy:**
  - Encapsulated code is easy to test for unit testing

Storage Management

- It is useful to understand how storage is managed in different programming languages and for different kinds of data.
- Storage is typically divided into 3 areas:
    - Static Storage Allocation
    - Stack-Based Storage Allocation
    - Heap-Based Storage Allocation

Static Storage Allocation

- Static storage allocation is appropriate when the storage requirements are known at compile time.
- For a compiled, linked language, the compiler can include the specific memory address for the variable or constant in the code it generates.
- Examples:
  - code in languages without dynamic compilation
  - all variables in FORTRAN IV
  - global variables in C, Ada, Algol
  - constants in C, Ada, Algol


Local Variable
Free memory
Global variable
Program Instructions
static variable
Stack
Heap
Permanent Storage area