CISC 3115

Midterm Review

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Review Gudie

- List of topics
  - https://huichen-cs.github.io/course/CISC3115/24SP/reviews/
- Sample midterm exam
  - https://bbhosted.cuny.edu/webapps/blackboard/content/listContent.jsp?course_id=_2327615_1&content_id=_84465386_1
- Extra-credit assignment
  - https://bbhosted.cuny.edu/webapps/blackboard/content/listContent.jsp?course_id=_2327615_1&content_id=_84465402_1

Java language idiom

- Identifier naming convention
  - Constant?
  - Variable?
  - Method?
  - Class?
- Indentation and spacing

Problem solving strategies

- Divide-and-conquer
  - Divide the problem into subproblems, each is solved by a class
    - Association relationship
    - Inheritance relationship
  - Divide the problem into subproblems, each is solved by a Java method
- Using known algorithms
- Using Java API classes and classes written by the others
- Known patterns of program design

Defining classes

- Problem solving
  - Model a set of entities in your problem as a Java class (each entity is an object, an instance of the class)
    - Data fields (variables; instance vs. class variables)
    - Behavior (methods; instance vs. class methods)
      - Defining instance methods
      - Defining class/static methods
    - Initializing objects
      - Constructors (default constructor, parameterized constructors)

Using classes

- Problem solving
  - Each entity is an object, an instance of the class
- Referencing an object in your program
  - reference variables vs primitive type variables
- Calling/invoking instance methods

Using methods

- Model behavior of an entity as java methods
  - Methods can take a list of parameters and return a value
    - Passing object arguments to methods
    - Returning objects
  - Arrays of objects
    - As parameter, as a return value
- The this reference variable

Computer and JVM Basics

- Stack vs. heap

Problem solving using classes and objects

- Model a set of entities as a Java class
- Relationship among classes
  - Association (also, composition & aggregation)
    - Realizing association relationship in Java programs
  - Inheritance
    - Realizing inheritance in Java programs
    - Constructors (default constructor, parameterized constructor, constructor chaining, the super keyword)
    - Method overriding
    - Polymorphism and dynamic binding
    - Writing generic methods

Problem solving using classes and objects

- Relationship among objects
    1. Client-server model
    2. Message-passing model

Known patterns of program design

- Using visibility modifier
- Using the final keyword
- Design mutable and immutable objects
- Copy constructors
- Using Java API classes
  - Math, Random, String, StringBuilder, BigDecimal, BigInteger, ArrayList, Arrays, Collections, and Wrapper classes

Arrays vs ArrayList

- Array?
- ArrayList?

Known algorithms and programming patterns

- Mostly from CISC1115, we can categorize them in the following categories
  - Numerical Algorithms
  - Logical Algorithms
  - String Algorithms
  - Data input algorithms
  - Ordered Type Algorithms (numerical and String), e.g.,
  - Sequence of Input and Array/ArrayList Traversal Algorithms
  - Adjacency Algorithms
  - Single array Modification
  - Multiple Array Algorithms

Numerical Algorithms and Patterns

1. Finding absolute value
2. Toggling: 0<-->1 and -1<-->1
3. Parity
4. Divisibility
5. Primality
6. Square root finding (binary search and Newton's algorithm)
7. Histogram construction

String Algorithms and and Patterns

1. Forming initials from String variables or an array, variations on separator
2. Picking out tokens
3. Finding the nth word in a String of text
4. Counting substrings
5. String replacement

Data Input Algorithms and Patterns

- Organized data with header
- End of data (e.g., using sentinel, i.e., trailing token)
- Nested input sequences (either header or sentinel based)
- Input validation

Ordered Type Algorithms and Patterns

- On numerical and String data types, e.g.,
  - Maximum (minimum) of two
  - Maximum (minimum) of three
  - Checking for ascending (descending) order of two
  - Checking for ascending (descending) order of three

Sequence of Input and Array/ArrayList Traversal Algorithms

- Accumulations.
  - addition, subtraction, multiplication, division
- Checking for ascending (descending) order
- Finding extremes (maximum, minimum)
- Counting items having a common property
- Determination of all/some/none of the items having a common property

Adjacency Algorithms and Patterns

1. Identifying runs (e.g., of consecutive numbers)
2. counting or eliminating duplicates

Single Array Modification

1. Element shifting
2. Reversing elements
3. Frequency counts

Array/ArrayList Algorithms

1. element-to-element arithmetic, comparison, string operations
2. copying, selecting
3. Sorting and Searching
4. Selection sort
5. Sequential search
6. Binary search

Questions?