Introduction to CS II – Data Structures
CAS CS 392X1, Spring 2026

(Syllabus)

- **Semester** Spring 2026
- **Teaching Staff**
  - Instructor – Hongwei Xi
    - Office Hours: TBA
    - Location: CDS 727, e-mail: `hwxi@cs.bu.edu`
  - Teaching Fellow – TBA
    - Office Hours: TBA
    - Location: TBA
- **Classroom**: CAS B06B
- **Lecture Times**: TR 3:30-4:45PM
- **Textbook**:
  - Algorithms by R. Sedgewick and K. Wayne, 4th edition. Addison-Wesley, 2011.
- **Homepage**: `https://hwxi.github.io/TEACHING/CS392/2026Sprn`
- **Description**:

  This course starts by quickly revisiting and then building upon basic programming concepts in Java. Then, the main focus of the course is on the design, analysis and implementation of fundamental data structures used throughout computer science. These include linked lists, stacks, queues, trees, hash tables, graphs, as well as specialized methods for searching and sorting. All of our implementations will be in the the object-oriented programming language Java. The emphasis in teaching this course centers around the following:

  - Developing elegant and efficient code from an abstract specification;
  - Literate programming (writing programs that can be read by humans as well as machines);
  - Developing a toolbox of advanced data structures for use in your future programming tasks, and an awareness of various design patterns that recur frequently in advanced programming;
  - Critical thinking about programs and the programming process, which involves:
    - Thinking about the best way to plan out the design using object-oriented design and appropriate features of Java;
    - Methodical and efficient development of the implementation using step-wise refinement and incremental testing and debugging (using appropriate debugging tools);

    * Being able to convince yourself of the correctness of the implementation by mathematical reasoning;
    * Analyzing the running time (efficiency) of programs by inspection and mathematical reasoning; and
    * Evaluating the efficiency and correctness of programs empirically, by using various tools in properly designed experiments.

- **Prerequisites**:

  This course is designed for students who already have a basic level of proficiency in Java. If you do not have significant previous exposure to programming, then you are requested to transfer to CS 111. You are expected to be familiar with a text editor (e.g., VIM and EMACS).

- **Exams**:

  - First midterm exam: TBA
  - Second midterm exam: TBA
  - The final exam date is yet to be anounced.

- **Grades** The final score is calculated using the following formula.

  $$\text{final score} = 25\%\cdot(\text{homework}) + 30\%\cdot(\text{midterm}) + 35\%\cdot(\text{final}) + 10\%\cdot(\text{participation})$$

  The final letter grade is calculated as follows.

  - **A**: final score is 85% or above
  - **B**: final score is 75% or above
  - **C**: final score is 65% or above
  - **D**: final score is 50% or above

- **Program Submission**: Program assignments are to be submitted via the *gsubmit* program. In case you need to learn how to use *gsubmit*, its documentation is available via a link on the homepage of this course.

- **Attendance**: It is expected that you will attend the lectures and the lab sessions for this course.

- **Academic Integrity**: We adhere strictly to the standard BU guidelines for academic integrity. For this course, it is perfectly acceptable for you to discuss the general concepts and principles behind an assignment with other students. However, it is not proper, without prior authorization of the instructor, to arrive at collective solutions. In such a case, each student is expected to develop, write up and hand in an individual solution and, in doing so, gain a sufficient understanding of the problem so as to be able to explain it adequately to the instructor. Under *no* circumstances should a student copy, partly or wholly, the completed solution of another student. If one makes substantial use of certain code that is not written by oneself, then the person must explicitly mention the source of the involved code.