Syllabus: CS 33101 Structure of Programming Languages
Term: Spring 2021

Ms. Aditi Singh
Kent State University
Department of Computer Science
Email: asingh37@kent.edu
Office hours: 2:30 pm – 3:30 pm [Monday /Wednesday] or by appointment
Grader: Venkata Sriram Konijeti
Grader email: vkonijet@kent.edu

Delivery: Remote; scheduled online sessions and online exams.
Dates: January 19 – May 4, 2021

Course Information

Course Description
The motivation behind this course is to prepare you for better programming and writing compilers, and to prepare you for tomorrow since languages keep changing while the basic principle and design philosophy does not alter.

- This course will provide basic understanding of general design issues and behavior of different class of programming languages. No specific programming language syntax will be discussed. Examples will use syntax of popular languages in a specific class of languages.
- In each module, we will learn about the underlying structure for each programming paradigm like imperative programming, functional programming, logic programming, object oriented programming.

Course Times and Location
This is a remote, 15-week course. All assignments have due dates; please refer to the course schedule located within the course.

Prerequisites
CS 331001 Data Structures

Course Goals
By the end of this course, you will:

1. Understand the design issues and pitfalls of different class of programming languages.
2. Understand the implementation issues in programming languages.
3. Understand various middleware level concepts.
4. Be able to learn new programming languages with the aid of a book and manual.

Course Learning Outcomes
By the end of the course, you will be able to:

1. Classify programming languages based on programming paradigms and implementation models.
2. Explain different translation mechanisms used in programming languages to translate high-level language to low-level machine language.
3. Differentiate programming paradigms used based on multitude of problem domains.
4. Recognize the type of grammar used in programming language in terms of syntax and semantics.
5. Distinguish between high-level and assembly level programming based on data and control abstraction.
6. Compare and contrast information exchange methods between program units in different programming languages.
7. Explore memory allocation schemes used in programming languages for optimum use of memory.
8. Evaluate a programming language based on abstract concepts in computation such as binding and scope rules, computation modules, and types used.
9. Select an appropriate programming language that uses the least resources for solving a computation problem.
10. Investigate different paradigms used for a particular programming language.

Learning Materials

- Other reference books (optional):
Topics

Language Pragmatics (Chapter 1)
[Elective]: [2.0 hours]

Topics
- Principles of languages design and orthogonality
- Program components and Structured programming
- Problem domain and programming paradigms
- Attributes of good programming languages
- Control flow diagrams and iterative constructs
- External calls and system libraries
- Software maintenance and Software development cycle

Program Representation and Abstraction (Chapter 2 and Chapter 4: Sec. 4.1 to 4.8)
[Core Tier 2]: [3.0 hours]

Topics
- Von Neuman machine and the required Abstract Computation Concepts
  - Environment and Store
  - Abstract data structures such as stack and heap
  - Finite state machine
  - Hash tables and hashing techniques
- Scope rules: static vs. dynamic scope
- Namespace, types of variables and modules
- Data abstraction
- control abstraction
- Parameter passing, aliasing and side-effects
- Programs that take (other) programs as input such as interpreters, compilers, documentation generators
- Abstract syntax trees (see under syntax analysis and formal semantics)

[Elective]: [1.0 hour]

Topics
- Nondeterministic programming
- Software reuse and Interoperability

Advanced Programming Constructs (Chapter 4: Sec. 4.2 and 4.6; Chapter 9: Sec. 9.5; Chapter 11: Sec. 11.2)
[Elective]: [1.0 hour] (Note: Most of the topics are covered under respective sections)

Topics
- Lazy evaluation (see under functional programming)
• Control Abstractions: Exception Handling (see under Program Representation and Abstractions)
• Object-oriented abstractions: Multiple inheritance, Mixins, Traits, Multimethods (also see under object-oriented programming)
• Module systems
• String manipulation via pattern-matching (also see under Basic Type Systems)
• Dynamic code evaluation (“eval” see under functional programming)

**Grammars and Syntax analysis (Chapter 3: Sec. 3.1 to 3.4)**

[Elective]: [3.0 hours]

**Topics**
• Types of grammar
  • Regular grammar; context-free grammar and context sensitive grammar
• Regular grammar and lexical analysis using regular expressions
• Context-free grammar and Parsing
  • Formal modeling of context-free grammar
  • Backus-Naur form and extended Backus-Naur form representation
  • Syntax diagrams and their transformation to BNF and EBNF
  • Parse-tree and basics of parsing
  • Ambiguity in parsing and resolution

**Formal Semantics and Compiler Semantic Analysis (Chapter 3: Sec. 3.1 and Sec. 3.5)**

[Elective]: [1.5 hours]

**Topics**
• Definition of semantics
• Correspondence between semantics and syntax
• Lambda Calculus (see under Functional Programming)
• Approaches to semantics: Operational, Denotational, Axiomatic and Behavioral
• Abstract Syntax Trees; contrast with concrete syntax
• Scope and binding resolutions
• Attribute grammar
• Formal definitions

**Language Translation and Execution (Sec. 1.4, Chapter 5: Sec. 5.1 – 5.7; Chapter 6: Sec. 6.1 to 6.8; Sec. 12.3)**

[Core Tier 2]: [4.0 hours]

**Topics**
• Abstract Computing Machine for low level program execution
• Interpretation vs. compilation to native code vs. compilation to portable intermediate representation
• Language translation pipeline: parsing, optional type-checking, translation, linking, execution
• Execution as native code or within a virtual machine
• Alternatives like dynamic loading and “just-in-time” code generation
• functions (closures) (see under Functional Programming)
• Run-time layout of memory: call-stack, heap, static data
  • Implementing loops, recursion, and tail calls
• Memory management (cross-listed with Runtime Systems)
• Manual memory management: allocating, de-allocating, and reusing heap memory
• Automated memory management: garbage collection as an automated technique using the notion of reachability

[Elective]: [1.0 hour]

Topics
• Data layout for objects, activation records and recursive data structures
• Implementing different types of parameter passing
  • Data layout for call by value, call by reference, call by value-result
  • code substitution in call by name
  • side-effect and issues with pointers

Code Generation (Sec. 1.4; Chapter 5: Sec. 5.2 to 5.4)
[Elective]: [1.0 hour]

Topics
• Procedure calls and method dispatching (also see under object-oriented programming)
• Separate compilation: linking
• Low level translation of control-flow constructs such as conditional statements, definite and indefinite iteration

Runtime Systems and Dynamic Memory Management (Chapter 6: Sec. 6.1 to 6.8, Sec. 12.3)
[Core Tier 2]: [3.0 hours]

Topics
• Memory management
  • Manual memory management: allocating, de-allocating, and reusing heap memory
  • Automated memory management: garbage collection as an automated technique using the notion of reachability
• Dynamic memory management approaches and techniques: malloc/free, fragmentation, garbage collection (mark-sweep, copying, incremental garbage collection, reference counting and concurrent garbage collection)
• Just-in-time compilation
• Other common features of virtual machines: class loading and security
• Case Study from JVM (Java Virtual Machine)

**Basic Type Systems (Chapter 7: Sec. 7.1 to 7.9)**

[Core-Tier1]: [1.5 hours]

**Topics**
• Advantages of type declaration in error detection and memory optimization
• A type as a set of values together with a set of operations
  • Primitive types (e.g., numbers, Booleans)
  • Compound types built from other types (e.g., records, unions, arrays, lists, functions, references)
• Association of types to variables, arguments, results, and fields
• Type safety and errors caused by using values inconsistently given their intended types
• Goals and limitations of static typing
  • Eliminating some classes of errors without running the program

[Core-Tier2]: [1.5 hours]

**Topics**
• Generic types
  • Definition
  • Polymorphism: parametric, inclusion, ad hoc and coercion
  • Comparison of different types of polymorphism
  • Prescriptive vs Descriptive polymorphism
  • Universal reference types
• Complementary benefits of static and dynamic typing
• Errors early vs. errors late/avoided

[Elective]: [1.0 hour]

**Topics**
• Type equivalence
• Different schemes for the Implementation of polymorphic languages
• Study of type systems and polymorphism including polymorphic programming in Ada, C++, Scala and Ruby

**Concurrent Programming (Chapter 8: Sec. 8.1 to 8.4, Sec. 8.7)**

[Elective]: [2.5 hours]
Topics
- Racing conditions and thread dependencies
- Control dependency
- Data dependency
  - Types of data dependency
  - Producer-consumer relationship
  - Anti-dependency
- Program dependency graph
- Parallelization techniques
  - Synchronization and sequential execution
  - Loop unfolding
- Granularity and execution efficiency
- Program slicing and concurrency
- Task and data parallelism
  - Critical sections and multithreads
  - Integrating task and data parallelism
- Parallel programming language constructs

Functional Programming (Chapter 9: Sec. 9.1 – 9.5)
[Core-Tier1] [2.5 hours]
Topics
- Evaluation of \( \lambda \)-expressions
  - applicative order vs. normal order reduction
- FPS – Functional Programming System
  - Kernel functions
  - Forming complex function using composition, map, apply-all, etc.
- Effect-free programming
  - Function calls have no side effects, facilitating compositional reasoning
  - Variables are immutable, preventing unexpected changes to program data by other code
- Processing structured data (e.g., trees) via functions with cases for each data variant
- Associated language constructs in languages like LISP, Haskell, Scala and Ruby
- Functions defined over compound data in terms of functions applied to the constituent pieces
- First-class functions

[Core-Tier2]: [1.5 hours]
Topics
- Function closures (functions using variables in the enclosing lexical environment)
  - Basic meaning and definition -- creating closures at run-time by capturing the environment
• Canonical idioms: call-backs, arguments to iterators, reusable code via function arguments
• Using a closure to encapsulate data in its environment (see SECD machine)
• Defining higher-order operations on aggregates, especially map, reduce/fold, and filter

[Elective]: [1.0 hours]

Topics
• Implementation Models for functional languages
  • SECD Machine and eager evaluation
  • Graph reduction techniques and strictness analysis
  • Implementing lazy evaluation

Logic Programming (Chapter 10: Sec. 10.1 – 10.5)
[Elective]: [3.0 hours]

Topics
• First-order predicate calculus and correspondence with logic programs
• Clausal representation of data structures and algorithms
• Unification vs. assignment and expression evaluation
• Programming
  • Simple logical program including recursive programs
  • Nondeterministic programming
• Implementation of Logic Programs
  • AND-OR Tree
  • Mapping query reduction to AND-OR tree
  • Backtracking and multiple solution generation
• Shallow vs. deep backtracking
• Warren Abstract Machine
• Role of Trail Stack

Object-oriented Programming (Chapter 11: Sec. 11.1 – 11.6)
Supplemented with the course: object-oriented programming CS 13012
[Core-Tier1]: [2.5 hours]

Topics
• Object-oriented design
  • Decomposition into objects carrying state and having behavior
  • Class-hierarchy design for modeling
• Definition of classes: fields, methods, and constructors
• Subclasses, inheritance, and method overriding
• Dynamic dispatch: definition of method-call
[Core-Tier2]: [2.5 hours]

Topics

- Subtyping
  - Subtype polymorphism; implicit upcasts in typed languages
  - Notion of behavioral replacement: subtypes acting like supertypes
  - Relationship between subtyping and inheritance
- Object-oriented idioms for encapsulation
  - Privacy and visibility of class members
  - Interfaces revealing only method signatures
  - Abstract base classes
- Using collection classes, iterators, and other common library components
  - Implementation of object-oriented languages
    - Run-time representation of core language constructs such as objects, virtual methods tables and first-class objects
  - Storage allocation and deallocation
  - Object allocation and deallocation in heap
  - Attribute tables
  - Implementing casting
  - Implementing multiple-inheritance
  - Implementing virtual entities
  - Runtime behavior

Event-Driven and Reactive Programming (Chapter 13: Sec. 13.1)
[Core-Tier2]: [1.5 hours]

Topics

- Event Model
  - Events and event handlers
  - Defining event handlers/listeners
  - Main event loop not under event-handler-writer’s control
- Event-based programming
  - Externally-generated events and program-generated events

Technology Requirements and Skills

Computer Hardware and Software
Students new to Kent State University should review Information Service’s Technology Viewbook (link available in the Preparing your computer section of the Getting Started in Your Online Course link within the Start Here folder). A personal computer with consistent, reliable Internet access is required:

1. A DSL or cable connection to the Internet; dial-up is not sufficient.
2. Laptop or desktop computer with a minimum of a 2 GHz processor and 2 GB of RAM

You should have one of the following computer operating systems and additional software applications installed on your computer:

1. Windows 7 system operating software or newer for PC computers OR Mac OS X 10.6 or newer for Apple Mac computers.
2. Microsoft Office Suite (Word, Excel, PowerPoint) discounts available at The Microsoft Store, link available in the Preparing your computer section of the Getting Started in Your Online Course link within the Start Here folder.
3. A free version of Microsoft Office is available for students. Instructions and information can be found on support.kent.edu.
4. Antivirus for Windows OS, Microsoft Security Essentials OR Antivirus for Mac OS, Sophos
5. A Blackboard Learn compatible browser, such as the latest version of Mozilla Firefox. Blackboard also supports Chrome and Safari. **Internet Explorer is NOT a supported browser and should not be used.**

**Technology Skills**

Basic understanding of at least one programming language. Navigating a computer operating system, launching and quitting applications, connecting to the Internet, using a web browser to search the World Wide Web, downloading, saving, and uploading files, and sending and replying to email. Also, basic skills in MS Word, MS PowerPoint.

**Blackboard (Bb) Learn**

This class will use Blackboard (Bb) Learn, the official learning management system (LMS) used by Kent State University to deliver course materials to university students. ALL course materials and activities will take place in Bb Learn.

In order to login to the online Bb Learn LMS, students will need a Kent State FlashLine User Name ID and password.

- Students can login to Bb Learn either through a student FlashLine account or via a direct link to the login page: [https://learn.kent.edu](https://learn.kent.edu)

For help using the Blackboard (Bb) Learn system use the “Bb Learn Tutorials for Students” link in the main navigation of your course.

In general, Bb Learn works best using the latest version of most major web browsers, including Firefox (recommended), Chrome, and Safari.

**Technology Help Guidelines**

A. **30-Minute Rule:** When you encounter struggles with technology, give yourself 30 minutes to ‘figure it out.’ If you cannot, then post a message to the discussion board; your peers may have suggestions to assist you. You are also directed to contact the
KSU Helpdesk 24/7. As a last resort, contact me. However, do not expect an immediate reply, and I cannot guarantee that I will be able to help with any and all technology issues.

B. When posting or sending email requesting help with technology issues, whether to the Helpdesk or me, use the following guidelines:

1. Include a descriptive title for the subject field that includes 1) the name of course 2) the issue. Do NOT just simply type “Help” into the subject field or leave it blank.
2. List the steps or describe the circumstance that preceded the technical issue or error. Include the exact wording of the error message.
3. When possible, always include a screenshot(s) demonstrating the technical issue or error message.
4. Also include what you have already tried to do to remedy the issue (rebooting, trying a different browser, etc.).

Policies and Expectations

Online Attendance Policy
Online courses are conducted on the premise that regular attendance requires students to log into the Bb Learn learning management system (LMS). Attendance is measured both by virtual presence in the online course and student interaction with course learning materials, discussions and assignments. Students are expected to check their Kent State e-mail and to log into the system multiple times (at least every other day) during the week.

All actions by students in the Bb Learn LMS can be tracked. At any time during the course, an instructor may generate a report that indicates when and how long individual students have been logged into the LMS, or engaged with course materials or course tools.

Students who anticipate an absence from the online course due to technical or medical reasons should consult with the instructor individually. An absence due to illness or injury requires verification from a medical professional and should be presented to the instructor.

Communication Policy
1. Email course questions and personal concerns, including grading questions, to me privately using your @kent.edu email. Do NOT submit posts of a personal nature to the discussion board.
2. Email will be checked at least twice per day Monday through Friday; Saturday and Sunday, email is checked once per day. During the week, I will respond to all emails within 24 hours; on weekends and holidays, allow up to 48 hours. If there
are special circumstances that will delay my response, I will make an announcement to the class.

3. For questions related to technology, please contact: 330-672-HELP for 24/7 support.

**Online Student Conduct and (N)etiquette**

Communicating appropriately in the online classroom can be challenging. In order to minimize this challenge, it is important to remember several points of “internet etiquette” that will smooth communication for both students and instructors:

1. **Read first, Write later.** Read the ENTIRE set of posts/comments on a discussion board before posting your reply, in order to prevent repeating commentary or asking questions that have already been answered.

2. **Avoid language that may come across as strong or offensive.** Language can be easily misinterpreted in written electronic communication. Review email and discussion board posts BEFORE submitting. Humor and sarcasm may be easily misinterpreted by your reader(s). Try to be as matter-of-fact and professional as possible.

3. **Follow the language rules of the Internet.** Do not write using all capital letters, because it will appear as shouting. Also, the use of emoticons can be helpful when used to convey nonverbal feelings. 😊

4. **Consider the privacy of others.** Ask permission prior to giving out a classmate's email address or other information.

5. **Keep attachments small.** If it is necessary to send pictures, change the size to an acceptable 250kb or less (there are several programs you can use to do this such as: Photoshop, Paint, GIMP, and picresize.com).

6. **No inappropriate material.** Do not forward virus warnings, chain letters, jokes, etc. to classmates or instructors. The sharing of pornographic material is forbidden.

**NOTE:** The instructor reserves the right to remove posts that are not collegial in nature and/or do not meet the Online Student Conduct and Etiquette guidelines listed above.

**University Use of Electronic Email**

A university-assigned student e-mail account is the official university means of communication with all students at Kent State University. Students are responsible for all information sent to them via their university-assigned e-mail account. If a student chooses to forward information in their university e-mail account, he or she is responsible for all information, including attachments, sent to any other e-mail account. To stay current with university information, students are expected to check their official university e-mail account and other electronic communications on a frequent and consistent basis. Recognizing that some communications may be time-critical, the university recommends that electronic communications be checked minimally twice a week.
Assignments and Grades

A detailed breakdown of course assignments and due dates by lesson module is available as a separate .pdf document that can be accessed in Bb Learn by clicking on the Syllabus & Course Schedule link in the course menu.

Writing Assignments
Students should submit all writing assignments via Blackboard. Students should type the answers in the submission area for each assignment.

Discussion Boards
Discussion boards are integral part of the course. Two modules include a discussion and students are required to participate in each discussion.

• Each discussion includes an initial post which should include approximately 7-10 sentences.
• After initial post, students are required to respond to at least one of their classmates’ post.

Exams
Exams will be conducted on ground.

Assessment Feedback
• All assignments and discussions will be submitted and graded on Blackboard.
• The grades will be posted within 7 days of the due date on Blackboard.

Late and Make-up Work Policy
Late homework will be penalized 20% everyday including weekends. In general, you will have adequate time to complete each assignment.

Assignment Distribution and Grading Scale

<table>
<thead>
<tr>
<th>Method of Assessment</th>
<th>Number of Assessments</th>
<th>Weights</th>
<th>Total Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignments (Each 50 points)</td>
<td>9</td>
<td>20%</td>
<td>450</td>
</tr>
<tr>
<td>Exams (Each 100 points)</td>
<td>3</td>
<td>50%</td>
<td>300</td>
</tr>
<tr>
<td>Discussions (Each 50 points)</td>
<td>2</td>
<td>10%</td>
<td>100</td>
</tr>
<tr>
<td>Project (100 point)</td>
<td>1</td>
<td>20%</td>
<td>100</td>
</tr>
<tr>
<td><strong>Total Possible Points</strong></td>
<td><strong>950</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

You can calculate your weighted average grade at any time during the course by using one of the following online tools listed below:
Blacks Domain Weighted Average Grade Calculator:
Mercer University Weighted Average Grade Calculator
http://www.mercer.edu/registrarcalc/weightedaverage.html

<table>
<thead>
<tr>
<th>Percentage of Earned Points</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>85% - 100%</td>
<td>A</td>
</tr>
<tr>
<td>82% - 84%</td>
<td>A-</td>
</tr>
<tr>
<td>78% - 81%</td>
<td>B+</td>
</tr>
<tr>
<td>75% - 77%</td>
<td>B</td>
</tr>
<tr>
<td>72% - 74%</td>
<td>B-</td>
</tr>
<tr>
<td>68% - 71%</td>
<td>C+</td>
</tr>
<tr>
<td>65% - 67%</td>
<td>C</td>
</tr>
<tr>
<td>62% - 64%</td>
<td>C-</td>
</tr>
<tr>
<td>58% - 61%</td>
<td>D+</td>
</tr>
<tr>
<td>55% - 57%</td>
<td>D</td>
</tr>
<tr>
<td>50% - 54%</td>
<td>D-</td>
</tr>
<tr>
<td>Under 50%</td>
<td>F</td>
</tr>
</tbody>
</table>

University Policies

Students are required to be aware of and follow all general and academic policies established by Kent State University. A list of the general academic policies is listed on the Kent State University Policy Register, which can be found in the University policies section of the Getting Started in Your Online Course link within the Start Here folder. Specific policies related to the successful completion of this online course can be located and reviewed in your Blackboard Learn course.

Students with Disabilities

University policy 3342-3-01.3 requires that students with disabilities be provided reasonable accommodations to ensure their equal access to course content. If you have a documented disability and require accommodations, please contact the instructor at the beginning of the semester to make arrangements for necessary classroom adjustments. Please note, you must first verify your eligibility for these through Student Accessibility Services (contact 330-672-3391 or visit www.kent.edu/sas for more information on registration procedures).

The Blackboard accessibility statement can be found in the University policies section of the Getting Started in Your Online Course link within the Start Here folder.
**Course Enrollment and Withdrawal**

University policy requires all students to be officially registered in each class they are attending. Students who are not officially registered for a course by published deadlines should not be attending classes and will not receive credit or a grade for the course. Each student must confirm enrollment by checking his/her class schedule (using Student Tools in FlashLine) prior to the deadline indicated.

If registration errors are not corrected by this date and you continue to attend and participate in classes for which you are not officially enrolled, you are advised now that you will not receive a grade at the conclusion of the semester for any class in which you are not properly registered. Also, it is your responsibility to check the withdrawal dates for each semester.

**Plagiarism and Academic Integrity**

Students enrolled in the university, at all its campuses, are to perform their academic work according to standards set by faculty members, departments, schools and colleges of the university; and cheating and plagiarism constitute fraudulent misrepresentation for which no credit can be given and for which appropriate sanctions are warranted and will be applied.

For more information see the Kent State policy on plagiarism in the University policies section of the Getting Started in Your Online Course link within the Start Here folder.

**Subject to Change Statement**

The syllabus and course schedule may be subject to change. Changes will be communicated via email or the Blackboard Learn announcement tool. It is the responsibility of students to check email messages and course announcements to stay current in their online courses.