Jonathan Trans

https://www.linkedin.com/in/jonathantrans/
jbtrans@ucdavis.edu | 209-319-6752 | Davis, CA.

DETAILED NOTABLE COURSEWORK @ *University of California - Davis*

ECS 32A - Introduction to Programming **Davis, CA**
Python *September 2019 - December 2019*

- Introduction to programming and problem-solving in Python.
  - Algorithms: general concept, development of efficient algorithms
  - Scalar data types, the concept of data type, standard and user-defined scalar types
  - Simple statements, arithmetic, and boolean expressions, assignment statements, simple input and output statements (including file I/O)
  - The flow of control, repetitive statements, conditional statements
  - Data structures: lists, strings
  - Functions: general concept; declaration and calls; parameters
  - Classes
  - Use of an Integrated Development Environment (IDE)

MAT 21A - Calculus: Differential Calculus **Davis, CA**
Calculus *September 2019 - December 2019*

- The goal of this course is to help students develop effective strategies for solving both mathematical and real-world problems. Although students often do not like “word problems” probing applications of their mathematical skills, it is very important that instructors emphasize these types of problems so that students become experts at them. In particular, students should be taught how to create mathematical models, develop effective strategies for solving problems in applied settings and non-routine situations.
  - Students will learn the fundaments of differential calculus. At its most rudimentary level, this is the study of rates of change and motion which enables computations of how systems evolve based on simple laws. Calculus, therefore, forms the foundations of modern science. In this course, students not only learn the basic techniques of calculus but also how to distill the mathematical essence of real-world systems in terms of well-posed problems for the important variables of the problem.
  - This course is an entry point to advance mathematics. Mastery of this course enhances the analytic and problem-solving skills of students. At the same time, the course supports the ability to design mathematical models in a wide range of applied fields.

MAT 21B - Calculus: Integral Calculus **Davis, CA**
Calculus *January 2020 - March 2020*

- The goal of this course is to help students develop effective strategies for solving both mathematical and real-world problems. Although students often do not like “word problems” probing applications of their mathematical skills, it is very important that instructors emphasize these types of problems so that students become experts at them. In particular, students should be taught how to create mathematical models, develop effective strategies for solving problems in applied settings and non-routine situations.
  - Students will learn the fundamentals of integral calculus. The course begins with the integral of a continuous function defined as the limit of its Riemann sum before the Fundamental Theorems of Calculus presents integrals as anti-derivatives. Integrals are applied to many problems in physics, including area, volume, arc length, surface area, and center of gravity. Various techniques of integration are studied and include u-substitution, integration by parts, integration by partial fractions, and trigonometric substitution. After applying integrals to separable differential equations, the course concludes with the calculus of parametric equations.
  - This course is a prerequisite for multi-variable and vector calculus. Mastery of this course would be manifested in improved reading, writing, thinking, and problem-solving skills. Students should have an increased ability to understand, visualize, categorize, model, and solve complicated calculus problems in both two- and three-dimensional spaces.

MAT 21C - Calculus: Partial Derivatives and Series **Davis, CA**
Calculus *March 2020 - June 2020*

- The goal of this course is to help students develop effective strategies for solving both mathematical and real-world problems. Although students often do not like “word problems” probing applications of their mathematical skills, it is very important that instructors emphasize these types of problems so that students become experts at them. In particular, students should be taught how to create mathematical models, develop effective strategies for solving problems in applied settings and non-routine situations.
  - Care should be taken in this course to teach students how to produce simple yet meaningful sketches of the three-dimensional geometric objects studied. Use appropriate visualization technology.

Jonathan Trans

https://www.linkedin.com/in/jonathantrans/
jbtrans@ucdavis.edu | 209-319-6752 | Davis, CA.

ECS 36A - Programming & Problem-Solving **Davis, CA**
Linux/Unix, C September 2020 - *December 2020*

- Computers and computer programming for students with some prior experience, algorithm design, and debugging. Good programming style. Use of basic UNIX tools.

1. **The UNIX Environment**
   - First-level understanding of the nature of UNIX processes and job control.
   - UNIX hierarchical file system.
   - UNIX utilities, e.g., find, grep, sort, uniq, head.
   - Shell script understanding.
2. **Algorithms: general concept, development of efficient algorithms**
3. **Programming in Python (or an alternative programming language)**
   - Review of the syntax of simple statements, arithmetic and boolean expressions, assignment statements, input and output statements.
   - Data types: lists, tuples, dictionaries, classes.
   - Functions: general concept, declaration, and calls, parameters, function call stacks.
   - Recursion.
   - Use of an Integrated Development Environment (IDE).
4. **Software engineering: running, debugging, testing programs, building quality program using software development tools**
   - Practice writing larger programs and finding good abstraction boundaries
   - Use of libraries and importing.
   - Debugging techniques, especially using debugging aids such as pdb/rpdb.

MAT 22A - Linear Algebra **Davis, CA**
Linear Algebra *September 2020 - December 2020*

- The purpose of MAT 22A is to introduce students to the fundamental objects and concepts in Linear Algebra, including scalars, vectors, matrices, diagonal matrices, symmetric matrices, inverse matrices, singular & nonsingular matrices, permutation matrices, linear combination, linear dependence & independence, vector spaces, subspaces, the dimension of a vector space, bases of vector spaces, rank, nullity, the four fundamental subspaces associated with a matrix, projections, determinants, permutations and cofactors, and eigenvectors & eigenvalues, and the fundamental operations on these objects including the dot product, matrix multiplication, matrix transpose, Gaussian elimination, Gauss-Jordon elimination, LU decomposition, matrix factorizations, reduction of a matrix to row reduced echelon form (RREF), the solution of systems of linear equations in an arbitrary number of unknowns, the least-squares procedure, and the Gram-Schmidt procedure. In MAT 22A a modern, innovative approach is taken in presenting this material. Each new idea is introduced by constructing concrete examples, in which, for example, the student learns to determine key parameters associated with a given matrix, (e.g., it's rank and nullity), by applying procedures, (e.g., row reduction) to find the relevant canonical form of the matrix (e.g., RREF) from which these parameters can be easily found. Furthermore, this approach provides the student with constructive proof of each of the key theorems presented in the course.
  - This allows students to learn the theory underlying Linear Algebra by associating facts (i.e., theorems) with procedures they have learned to perform on matrices and vectors, such as reduction to RREF. Thus, this approach gives students an introduction to logic and proof, which is one of the goals of this course.

MAT 22AL - Linear Algebra Computer Laboratory **Davis, CA**
Linear Algebra, Matlab September 2020 - December 2020

- Students will learn the basic operation of working with MATLAB software. The student will experience matrix operation using MATLAB and discover properties of Matrices and its operations by interacting with MATLAB. Students will learn about some applications of Linear Algebra and doing examples to represent concepts covered in their Linear Algebra course. Students will learn how to create m-files and use them for simple programming in MATLAB. Students will learn how to use MATLAB to simulate and repeat some of the algorithms used in their linear algebra course. Students will learn how to create two or three-dimensional graphs in MATLAB.
  - This course is an introduction to use MATLAB to enhance understanding of concepts and applications of Linear Algebra. Enhances the analytic and problem-solving skills of students.

ECS 36B - Software Development & Object-Oriented Programming **Davis, CA**
OOP, Linux/Unix, C++ *January 2021 - March 2021*

- Object-oriented programming in C++. Basic data structures and their use. Writing good programs of increased complexity and efficiency. Methods for debugging and verification.

Jonathan Trans https://www.linkedin.com/in/jonathantrans/
jbtrans@ucdavis.edu | 209-319-6752 | Davis, CA.

1. **Object-Oriented Programming and the C++ Language**
    - Object-oriented design and implementation, polymorphism, operator overloading, encapsulation, derivation, exceptions.
2. **Programming in C**
    - Basic syntax and difference of C and other languages, including Python, and semantic differences of C with C++
    - Data types: single and multidimensional arrays, character strings, structs.
    - Use of system files such as library and "include" files
    - Pointers and dynamic memory allocation.
    - Functions: declaration and calls, & and * operators, parameters, function call stacks.
    - Function pointers and inversion of control
3. **Software Engineering & Tools**
    - Integrated Development Environments
    - Debugging techniques, especially using UNIX debugging aids such as gdb/ddd.
    - Version Control
    - Unit Testing (e.g. GoogleTest)
    - Program development using third-party libraries, and use of the UNIX "make" program to organize them.
    - Function pointers and inversion of control
4. **Advanced Programming Concepts in C++**
    - Singly-and doubly-linked lists, recursion, and, if time permits, one or more topics chosen from: binary trees, stacks, queues
    - Templates and the Standard Template Library
    - Modern C++

ECS 32B - Introduction to Data Structures **Davis, CA**
Python *January 2021 - March 2021*

- Design and analysis of data structures using Python; trees, heaps, searching, sorting, and graphs.
    - Theory: Complexity, algorithm analysis
    - Abstract Data Types
    - Stacks, queues, lists, trees, balanced binary search trees, priority queues, graphs
    - Analysis of Sorting Algorithms
    - Insertion sort, merge sort, heapsort, bucket sort, and radix sort
    - Recursion

ECS 20 - Discrete Mathematics For Computer Science **Davis, CA**
Discrete Mathematics *January 2021 - March 2021*

- Discrete mathematics of particular utility to computer science. Proofs by induction. Propositional and first-order logic. Sets, functions, and relations. Big-O and related notations. Recursion and solutions of recurrence relations. Combinatorics. Probability on finite probability spaces. Graph theory.
    - The purpose of the course is to introduce fundamental techniques in discrete mathematics for applications in computer science. One of the central objectives is to learn methods of proof that transform intuition into mathematical proof and to stress the distinction between proof and opinion. Hence the course will be mathematical in two senses: first, it will contain specific techniques in discrete mathematics, and second, through examples and exercises, it will raise the student's general mathematical sophistication, i.e., the ability to deal with and create complex and convincing arguments.

ECS 36C - Data Structures in Python, Data Structures, Algorithms, & Programming **Davis, CA**
OOP, Linux/Unix, C++ *March 2021 - June 2021*

- Design and analysis of data structures for a variety of applications; trees, heaps, searching, sorting, hashing, and graphs. Extensive programming.
    - Students will be able to apply algorithm analysis to the operations of data structures and interpret the results.
    - Students will be able to understand and evaluate the operations and possible implementations of the following abstract data types: lists, stacks, queues, binary trees, AVL trees, splay trees, tries, B-trees, hash tables, and heaps.
    - Students will understand the operation and characteristics of the following sorting algorithms: insertion sort, shell sort, heapsort, mergesort, quicksort, bucket sort, radix sort, and external sorting.
    - Students will understand the operation and application of graph algorithms including depth-first search, breadth-first search, shortest path, minimum spanning tree, network flow, and topological sort.

Jonathan Trans

https://www.linkedin.com/in/jonathantrans/
jbtrans@ucdavis.edu | 209-319-6752 | Davis, CA.

- Students will be able to choose (and justify) appropriate data structures and algorithms for complex programming tasks.

ECS 50 - Computer Organization & Machine-Dependent Programming **Davis, CA**
x86 Assembly, C/C++ *March 2021 - June 2021*

- Comparative study of different hardware architectures via programming in the assembly languages of various machines. Role of the system software in producing an abstract machine. Introduction to I/O devices and programming.

1. **Basic Memory Architecture**
   - Review of ECS 40 C-language material on bits, bytes, and memory addresses. Linear versus segmented address forms.
2. **Introduction to Processor Architecture**
   - Introduction to instruction sets, addressing modes and register sets, and their variation from one machine to another. Comparison of at least one CISC architecture and at least one RISC architecture, via extensive assembly-language programming on each. Memory-mapped versus separate-address-space I/O. I/O devices. Hardware interrupts. I/O programming—wait-loop, interrupt-driven, and via calls to operating system services or keyboard, video, and file access.
3. **The Role of System Software in Producing an Abstract Machine**
   - The distinction between the roles of hardware and software. The use of compilers and operating systems in providing abstractions and machine independence to the programmer. Compiler implementation of C/Pascal data types, and storage allocation of variables in memory. Role of the operating system in helping the programmer to create, store and execute his/her programs, and in managing system resources.
4. **Machine Capability and Speed**
   - Transportability/nontransportability of programs on different machines, and under different operating systems and compilers. The efficiency of compiled vs. hand-coded programs. Writing mixed C-language/assembly language programs for extra efficiency or for special capabilities. Tradeoffs between the speed of hardware implementation of a function and the flexibility of software implementation.

STA 131A - Probability Theory **Davis, CA**
Upper-Divison Statistics *March 2021 - June 2021*

- Fundamental concepts of probability theory, discrete and continuous random variables, standard distributions, moments and moment-generating functions, laws of large numbers, and the central limit theorem.