**Computer Science and Engineering 150**
**Programming Languages for Artificial Intelligence**
**SECTION ID: 452735**

Instructor: **Gary Cottrell** Phone: 858-534-6640
Office: AP&M Prime 4872 (across the bridge, which is an immediate left from the elevator).
email: cs150f@ieng9 or gary@cs
Office Hours: Wednesday 12:00-1:00OM, Thursday 12:00-1:00PM, or by appointment.

TA's: **David Kauchak and possibly Kristin Branson**
Office: AP&M Prime 4859 (This may change)
Office Hours: Not known at this time.
email: cs150f@ieng9

**Class Lectures:** Tuesday/Thursday: 3:30-4:50, WLH 2204
**Weekly Section**: Friday 3:00-3:50 CSB 001 (NO SECTION FRIDAY, September 27th, 2002).

## Course Description

CSE 150 is an introductory course in artificial intelligence programming, in two parts: the use of LISP for symbolic manipulation, and the use of MatLab for statistical approaches. Familiarity with calculus, statistics, computer programming, including procedural abstraction, recursion, list and tree data structures, and elementary tree traversal algorithms is presupposed (see below).

## Prerequisites

You must have all the prerequisites listed below. This is the BARE MINIMUM. If that is all you have, you will have to work hard to keep up. You cannot make up for any missing prerequisite with extra work or a friend who can help you or even the grace of God. In determining the bare minimum, we already figured in all those friends and extra help. IF YOU DO NOT HAVE ALL THE PREREQUISITES DO NOT TAKE THIS CLASS.

(1) CSE 100. (This supposes that you have taken the prerequisites for 100!). *CSE 100 may be taken concurrently.* If you took a course someplace else it matters little what programming language you used but you need to have covered at least the following topics: procedures, local variables, arrays, recursion and pointers. If you are not very comfortable with recursion in some language, then you do not have the prerequisites and you should not take this class.

(2) Enough knowledge of UNIX to move files around and to use the vi editor. (You will receive absolutely no UNIX instruction in this class, because we will assume that you are already comfortable using UNIX.)

(3) Knowledge of very basic data structures: If you understand both depth first and breadth first searches of binary trees and can implement them in some programming language,

then you probably know enough data structures to get by. If you do not feel comfortable with the notions of depth first and breadth first searches, then you should not take this class.

(4) Knowledge of Calculus, in particular, partial derivatives, and statistics.

This course is a prerequisite for CSE 151.

**Computer Accounts**

See the TA as soon as possible to obtain your computer account if you do not have an OCE account. Note that you may access the class machine via the work stations in the UAPE Laboratory in the basement of AP&M (or from any terminal room on campus or by telephone).

**Required Texts**

*LISP, 3rd Edition*, by Winston and Horn.
*Artificial Intelligence: A Modern Approach* by Russell & Norvig.

We will place these texts on reserve at the S&E library.

**Required work**

You will be given a series of approximately 6 programming assignments. Usually, you will have 7-14 days to finish the assignment. Programming assignments will *not* be accepted late. There will also be two in-class midterms. The midterms will not cover the same material, although you are not expected to have had a brain wipe in between. *There will not be a final exam.* The various components are weighted as follows:
Programming Assignments: 50%Midterm 1: 25%Midterm 2: 25%

**Class schedule**

The following schedule is mildly optimistic:

10/01/02: A0 handed out (Read Ch1-2)
10/08/02: A0 due, A1 handed out Read Ch 3
10/15/02: A1 due, A2 handed out Read Ch 4
10/24/02: A2 due, A3 handed out
10/29/02: last lisp/search lecture.
10/31/02: First lecture on Ch. 19
11/04/02: Review session (evening)
11/05/02: MT1
11/12/02: A3 due/A4 handout out
11/21/02: A4 due, A5 handed out
12/03/02: last lecture
12/04/02: Review session (evening)
12/05/02: MT2 (last day of class)
12/07/02: A5 due (Note this is Sat. before finals).

Programming assignments are generally due at midnight on the specified date. (That is, 11:59PM plus 1 minute -- people ask!). The midterms are closed book and will be at the regular class times. *You must bring a picture ID to the midterms.*

**Programming Assignments**

*NOTE: Many messages regarding programming assignments will appear on the discussion board for the assignment. YOU ARE EXPECTED TO KNOW THAT INFORMATION! Check the discussion board frequently!*

When you have a question about the programming assignment, use the discussion board to clarify points and ask brief questions. *Do not post your code and ask what is wrong with it.* If you do, you will receive a zero for that assignment! Very small fragments of code may be appropriate to post. For big problems or particularly stubborn bugs see the TA or instructor in person. If it is about your program, bring a listing with you.

**Naming of Functions**

Part of the grading will be done automatically by grading programs. Be sure to spell function names exactly as the exercise states, otherwise the grading program will not find your function and will mark it incorrect. This applies to the choice of uppercase versus lowercase letters as well as the words and their spellings.

**Style rules for LISP programs**:

All programs must be easy to read and well documented. We will encourage the use of "functional programming" via the following rules: You may not use **setf**, **setq**, **set**, or any thing that has similar effects (unless the assignment explicitly says you may). You may not use **prog** or any loop construct (e.g. **do**) unless the assignment explicitly says that you may. More details guidelines will be given in class and you must follow them. If a program is not easy to read or if it contains a **set** or some other forbidden construct, then the grader will take off half of the style points for each one.

Generally, LISP programming encourages lots of small functions that do simple tasks. These are then composed to form the solution. We encourage you to follow this idea. "Small" is around 5 or 6 lines, but obviously, things get longer sometimes.

A large part of your grade will be determined by the following: We expect you to have function headers on your functions, and a page at the top of the file explaining your approach - data structures, your algorithm (if not specified in the assignment), explanation of your heuristics (if heuristics are asked for in the assignment) etc. The page at the top of your file is the "documentation". The style you use for your programming, and comments, are "style".

Here is an example. Your TA may provide others:

```
;;; Function:  make-open-init
;;; Arguments: initial_state
;;; initial_state should be a legal state of the puzzle.
;;; Returns: a legal OPEN list, with one path on it,
```

```
;;; consisting of the null move from the initial state.
;;; Side effects: none
(defun make-open-init (initial_state)
  ...
```

**Turning in Programs**:

Use the "turnin" facility to turn in your programs. To use it, first "prep cs150" if you have an OCE account. (Do nothing otherwise!). Then, simply type "turnin <filename>".

You MUST execute this command BEFORE MIDNIGHT ON THE DUE DATE (that is, 11:59PM plus one minute). Programming assignments will *not* be accepted late.

NOTE: HALF OF YOUR GRADE FOR PROGRAMS IS DOCUMENTATION AND "STYLE". DOCUMENTATION MEANS: A roughly page-long comment at the top of the file explaining what the program does, how it does it, and examples of its use. STYLE MEANS: A well-commented program written in an appropriate style for the language. See the example above. IT IS USUALLY BETTER TO TURN IN A NON-WORKING PROGRAM ON TIME THAT IS WELL-DOCUMENTED THAN A WORKING PROGRAM LATE, WHICH WILL RECEIVE A ZERO!

**Rules for collaboration on Programming Assignments:**

You may not work in groups unless the assignment explicitly says that you may. You must write your own programs. **The attached "Integrity of Scholarship Agreement" (ISA) is the final arbiter in cases of suspected cheating.** A good rule of thumb is the "Gilligan's Island Rule": You may discuss assignments with classmates *but you may not take away any written notes* and you must watch an hour's worth of Gilligan's Island (an old, pretty mindless TV show. A modern equivalent: "Who wants to be a millionaire") or equivalent before writing anything down. A second rule of thumb is, if your program logic ends up identical to your friend's, you have not followed the rule. If a question regarding cheating arises, we reserve the right to give you an oral exam on the homework to see if you understand your answer. If you do not understand the assignment, you will receive a zero for that assignment, even if the program you have is perfect in every way. If you are caught blatantly copying we will give you an F for the course.

**Cheating Policies**

Unfortunately, cheating has become a serious problem in CSE. A number of students have complained that they are being hurt by others cheating in classes. This is correct. Cheating lowers the value of honestly earned grades. Unfortunately, we have no alternative but to enforce strict rules against cheating. Everyone must sign and return to the TA the attached **Integrity of Scholarship Agreement**. *If you are having trouble with an assignment, see me or the TA during office hours or make an appointment!!! There is no reason to cheat!!!* UCSD was recently ranked the number 1 Public Science University in the US, based on things like grants awarded, papers published, etc. I expect you to learn to become independent thinkers and problem solvers while you are here.

*Exams*: Anybody talking during an exam will receive a zero for that exam. There will be no exceptions to this rule. If, as you leave an exam, you turn to a friend and say "meet me at the coffee cart when you finish," then you will receive a zero for that exam. "I forgot" or "I only said ..." will not change the zero. This may sound unreasonable. However, we have no good alternative. We can easily tell if you are talking, but we cannot listen to and evaluate everything that anybody in the room might say. Similarly, books or notes may not be with you at your seat during an exam. If you bring books into the exam room, then you must leave them in the front of the room and not pick them up until after the exam. If you are found to have any books or notes at your seat, you will receive a zero for the exam, even if you did not look at them. For the second midterm, calculators will NOT be allowed. You must remember how to multiply decimal numbers!!!

*Programming Assignments*: You must follow the Gilligan's Island rule above (but this, again, is just a rule of thumb - the fine print is in the Integrity statement).

Anybody caught clearly cheating will receive an F in the class and will be reported to their Provost's office. For example, if you are caught talking at all in an exam then you receive a zero for that exam, no matter how innocent the talk. If we are not sure you are cheating, that is the end of that. However, if we hear you giving or receiving answers, then you receive an F for the class. Programming assignments that are obvious copies are also an example of clearly cheating.

## Getting Started with LISP

To get the feel of LISP simply log on and then type

**cl**

This stands for "Common Lisp". You will then get a prompt that looks like the following:

**USER(1):**

You are now in the LISP interpreter. You can evaluate any LISP function by giving it along with its arguments in the correct format, e.g.,

**USER(1): (+ 2 3)**

produces the response:

**5**

Numbers are easy to work with at first because LISP does not care whether they have a quote in front or not. For other types of arguments and most functions, the arguments must normally be precede by a single quote when using the interpreter. The details are in the text and will be discussed in class, but the following simple example may help illustrate enough to let you start playing with the interpreter. The function **car** normally takes a list as its single argument. It returns the first thing on that list. For example,

**USER(2): (car ’(a b c))**

produces the response:

**A**

Common Lisp is case insensitive. **a** and **A** are the same to it. It will write in all caps. You can use either.

If you make a mistake, the prompt changes to what looks like an error count. For example, if you forget the quote and type

**USER(3): (car (a b c))**

the response will be:

**Error: bla bla bla**

**[1]USER(4):**

If you make a second error, the "1" will change to "2"s and so on. This can be treated just like any other prompt, but if you want to get back the old "clean" prompt, type **:reset**, that is, a colon followed by the word **reset**.. In other words to get rid of the error count type:

**[1]USER(4)::reset**

(The error count is there for reasons other than to remind you of how many mistakes you have made, but for now that need not concern us.)

You are not allowed to use **setf** in your homework assignments, but it is perfectly acceptable to use it in the interpreter to save values for testing. **setf** can be used much like an assignment operator. It binds values to atoms and then the atoms may be used as if they were the value. This works because of the way LISP evaluates functions and arguments. There will be more about that in class and there is more in the text book. To see a simple example before reading the details, you might type in something like the following to the LISP interpreter:

```
USER(5):(setf x '(a b c))
  (A B C)
USER(6):(car x)
  A
USER(7):x
  (A B C)
```

You type what is after the prompt **USER(1):**. The following line is the response of the interpreter. Note that **setf**, like all actions in LISP, is a function to be evaluated. It returns its second argument and has the side effect of binding a value to its first argument. Note that the second argument must be quoted. Its first argument is not quoted. That is a kludge to make your life easy. The first argument is quoted "automatically". Also, note that the **x** in the above example is not quoted. That is because we do not want **'x** we want the value of **x**. Also note that **x** by itself can be evaluated by LISP. That produces the value bound to **x**. If you make an error and the prompt changes, do not be concerned, just keep going or use **reset** as we described above.

To leave LISP, type

**USER(8): :exit**

Enjoy!

**CSE 150: Programming Languages for Artificial Intelligence**
**Integrity of Scholarship Agreement** (STUDENT COPY)

STUDENTS' RESPONSIBILITIES

Students are expected to complete the course in compliance with the instructor's standards. No student shall engage in any activity that involves attempting to receive a grade by means other than honest effort, for example:

1. No student shall knowingly procure, provide, or accept any materials that contain questions or answers to any examination or assignment to be given at a subsequent time.

2. No student shall complete, in part or in total, any examination or assignment for another person.

3. No student shall knowingly allow any examination or assignment to be completed, in part or in total, for himself or herself by another person.

4. No student shall plagiarize or copy the work of another person and submit it as his or her own work.

5. No student shall employ aids excluded by the instructor in undertaking course work.

6. No student shall alter graded class assignments or examinations and then resubmit them for regrading.

7. No student shall submit substantially the same material in more than one course without prior authorization. A student acting in the capacity of an instructional assistant (IA), including but not limited to teaching assistants, readers, and tutors, has a special responsibility to safeguard the integrity of scholarship. In these roles the student functions as an apprentice instructor, under the tutelage of the responsible instructor. An IA shall equitably grade student work in the manner agreed upon with the course instructor. An IA shall not make any unauthorized material related to tests, exams, homeworks, etc. available to any student.

The following are additional examples not listed in the General Catalog specific to programming classes:

8. No student shall provide their assignments, in part or in total, to any other student in current or future classes of this course. No student shall procure or accept assignments from any other student from current or prior classes of this course.

9. All programming code and documentation submitted for evaluation or existing inside the student's computer accounts must be the student's original work or material specifically authorized by the instructor. The course accounts are authorized for course work only.

10. Collaborating with other students to develop, complete or correct course work is limited to

activities explicitly authorized by the instructor. Use of other student's course work, in part or in total, to develop, complete or correct course work, including documentation, is unauthorized. Use of "Instant Messenger" or similar systems to share files in part or in total is unauthorized.

11. For all group assignments, each member of the group is responsible for the academic integrity of the entire submitted work.

Each student is responsible for knowing and abiding by UCSD's policies on Academic Dishonesty and on Student Conduct. Any student violating UCSD's Academic Dishonesty or UCSD's Student Conduct policies will earn an 'F' in the course and will be reported to their college Dean for administrative processing. Committing acts that violate Student Conduct policies that result in course disruption are cause for suspension or dismissal from UCSD.

Authorized course assistance is available in person and electronically from the course instructor, instructional assistants, and OASIS.

By signing this form, I, ____________________________(print LAST NAME, FIRST NAME), a student enrolled in CSE 150, understand and will abide by the above policies and by the spirit of the above policies. I will seek authorized assistance when I need help. I will keep a copy of this agreement for future reference. I have provided a signed copy of the agreement to the Instructor.

__________________________________________/________________________/______________________

Signature Student ID Date

**CSE 150: Programming Languages for Artificial Intelligence**
**Integrity of Scholarship Agreement** (INSTRUCTOR'S COPY)

STUDENTS' RESPONSIBILITIES

Students are expected to complete the course in compliance with the instructor's standards. No student shall engage in any activity that involves attempting to receive a grade by means other than honest effort, for example:

1. No student shall knowingly procure, provide, or accept any materials that contain questions or answers to any examination or assignment to be given at a subsequent time.

2. No student shall complete, in part or in total, any examination or assignment for another person.

3. No student shall knowingly allow any examination or assignment to be completed, in part or in total, for himself or herself by another person.

4. No student shall plagiarize or copy the work of another person and submit it as his or her own work.

5. No student shall employ aids excluded by the instructor in undertaking course work.

6. No student shall alter graded class assignments or examinations and then resubmit them for regrading.

7. No student shall submit substantially the same material in more than one course without prior authorization. A student acting in the capacity of an instructional assistant (IA), including but not limited to teaching assistants, readers, and tutors, has a special responsibility to safeguard the integrity of scholarship. In these roles the student functions as an apprentice instructor, under the tutelage of the responsible instructor. An IA shall equitably grade student work in the manner agreed upon with the course instructor. An IA shall not make any unauthorized material related to tests, exams, homeworks, etc. available to any student.

The following are additional examples not listed in the General Catalog specific to programming classes:

8. No student shall provide their assignments, in part or in total, to any other student in current or future classes of this course. No student shall procure or accept assignments from any other student from current or prior classes of this course.

9. All programming code and documentation submitted for evaluation or existing inside the student's computer accounts must be the student's original work or material specifically authorized by the instructor. The course accounts are authorized for course work only.

10. Collaborating with other students to develop, complete or correct course work is limited to

activities explicitly authorized by the instructor. Use of other student's course work, in part or in total, to develop, complete or correct course work, including documentation, is unauthorized. Use of "Instant Messenger" or similar systems to share files in part or in total is unauthorized.

11. For all group assignments, each member of the group is responsible for the academic integrity of the entire submitted work.

Each student is responsible for knowing and abiding by UCSD's policies on Academic Dishonesty and on Student Conduct. Any student violating UCSD's Academic Dishonesty or UCSD's Student Conduct policies will earn an 'F' in the course and will be reported to their college Dean for administrative processing. Committing acts that violate Student Conduct policies that result in course disruption are cause for suspension or dismissal from UCSD.

Authorized course assistance is available in person and electronically from the course instructor, instructional assistants, and OASIS.

By signing this form, I, ____________________________(print LAST NAME, FIRST NAME), a student enrolled in CSE 150, understand and will abide by the above policies and by the spirit of the above policies. I will seek authorized assistance when I need help. I will keep a copy of this agreement for future reference. I have provided a signed copy of the agreement to the Instructor.

__________________________________________/________________________/______________________

Signature Student ID Date