

**SCHOOL OF COMPUTER SCIENCE
COURSEWORK ASSESSMENT PROFORMA****MODULE & LECTURER:** CM0312 Artificial Intelligence II, F. C. Langbein**DATE SET:** 11th February 2010**SUBMISSION DATE:** 15th March 2010**SUBMISSION ARRANGEMENTS:**

Submit a single source code file containing the code for agent function and a brief description of your approach via the blackboard assignment module by 18:00. If there are any problems with submitting it via blackboard send the file by e-mail from your university e-mail address to F.C.Langbein@cs.cf.ac.uk with the subject "CM0312 AI CW I". All e-mails will be acknowledged. If you do not receive an acknowledgement within 24 hours or have any other problems contact the lecturer.

TITLE: Artificial Intelligence Coursework I

This coursework is worth 15% of the total marks available for this module. The penalty for late or non-submission is an award of zero marks. You are reminded of the need to comply with Cardiff University's Student Guide to Academic Integrity.

LEARNING OUTCOMES ADDRESSED:

- Explain how certain problems can be solved by searching and describe the issues involved
- Formulate an efficient problem space description and select and implement an appropriate search algorithm (brute-force, best-first, adversarial) or a simple constraint solving algorithm for finding a solution

INSTRUCTIONS

Complete the programming task as described in the attachment. Submit a single file called `MyPacManPlayer.java` containing the *source code* of your agent function with clear, concise *comments* which also provide a brief description of the approach taken by your agent.

Your agent function must be implemented in Java. It must run within the provided Pac-Man environment on at least one platform available at the Cardiff School of Computer Science. Your program may make use of any core Java API and the classes provided by the Pac-Man environment.

You may reproduce small code fragments from

- the CM0312 handouts, notes, web-site, tutorials and lab classes,
- any recommended and background textbooks for CM0312 as available from the printed text, CDs/DVDs, websites, etc. for the textbooks

provided that the sources contain clear reference to the origin of the code. Your comments in the code must indicate clearly that you understand the code fragment. Note that you will only get marks on your own code and the core algorithms and data structures must be implemented by yourself!

You may *not* reproduce code written by any other student or code obtained from any other source not mentioned above. If you are in doubt about whether you may include a code fragment that you have not written yourself, ask the lecturer.

CRITERIA FOR ASSESSMENT

Credit will be awarded against the following criteria.

	Excellent	Good	Adequate	Poor
Functionality: to what extent does the program realise the task described? 40%	efficient and complete implementation; all special cases are considered	feasible implementation, but not optimal; not all special cases are considered	progress towards a full implementation, but not fully working with major deficiencies	little or no progress towards implementation; approach not suitable
Design and Structure: how clear is the structure of the code and how well are data structures and algorithms used? 40%	well structured code with highly suitable data structures and elegant, clear algorithms	good structure with suitable data structures and algorithms; sometimes not optimal	attempt of using appropriate data structures and algorithms visible; sometimes structure is confusing	code is mostly obscure; little or no structure is visible in use of data structures and algorithm design
Code Documentation: how easy is it to understand the code with the comments provided? 20%	clear, concise comments describing ideas and high-level structure without unnecessary detail	clear comments about high-level structure and ideas; sometimes incomplete or too focused on details	some comments about the structure and ideas present, but hard to follow and too focused on details	hardly any comments or only very confusing or low-level comments about single instructions

The code documentation criterion is applied to the overall code documentation, whereas the functionality and design and structure criteria are applied to each individual feature described in the task and their combination will give the mark for each feature.

Feedback on your performance will address each of these criteria.

FURTHER DETAILS

Feedback on your coursework will address the above criteria and will be returned in approximately four weeks. This will be supplemented with oral feedback in lectures and tutorials. If you have any questions relating to your individual solutions talk to the lecturer or the tutor.

Task: Pac-Man Search Agent

The Pac-Man game is played in a 2D maze build on a regular grid that contains dots and the player controls Pac-Man, a yellow disc with a mouth. The maze also contains four ghosts in different colours which move around according to their own strategy. When Pac-Man runs into a dot he eats it. When he runs into a ghost, he dies. When Pac-Man dies, he loses one of his three lives. Once he has eaten all dots in a maze, he moves on to the next level with a new maze. The aim of Pac-Man is to eat as many dots as possible without wasting time, while trying to avoid dying. The task for this coursework is to design and implement one Pac-Man agent function which aims to get a score that is as high as possible.

The agent function must be based on a minimax search strategy. Minimax should be adjusted suitably to perform as good as possible on the problem. It has to be implemented as an agent function as part of the Java Pac-Man environment discussed in the lab classes. The source code for this environment is available on blackboard with this assignment in the `pacman.tar.gz` package. For details on this environment see the lab handout and the lab classes. You must write your own agent function in the `pacman/src/player/MyPacmanPlayer.java` file with appropriate comments and a description of your strategy in the comments and only submit one source code file. Do not edit any other part of the sources as doing so may mean that your agent is not behaving as expected or is not working at all for evaluation!

The particular features your agent function will be evaluated on are

- **Search algorithm (design):** implement a search algorithm based on a minimax search strategy with depth limit to search the Pac-Man game tree, i.e. a recursive search of the nodes in the game tree depending on their type to facilitate the game-specific functionalities as specified below. [6]
- **Problem-specific minimax strategy (functionality):** the minimax algorithm allows you to adjust the search to the specific problem by selecting the way the game tree is searched. To implement a good Pac-Man player, select specific options to
 - limit the branching factor at game tree nodes, e.g. by not considering that Pac-Man can reverse its direction in the search or ignoring the movement of distant ghosts; [2]
 - prune the search tree, e.g. by the usual alpha-beta pruning approach (this is sufficient, but if you are really ambitious you may instead consider some advanced pruning approach, e.g. zero window search, negascout); [2]
 - consider the ghosts as random opponents rather than optimal adversaries, e.g. by introducing chance nodes and considering the likelihood of all/some ghost moves occurring. [2]
- **Documentation:** your source code should have a clear structure and contain clear, concise comments. Also include in the source code at the beginning a longer comment describing briefly your overall search strategy focusing in particular on the specific adaption of the algorithm to the problem. [3]

Competition

The agent functions will be run multiple times (to account for the random effects) in the Pac-Man environment and their scores will be compared to generate a rank list of the submitted agents. The results will be announced in class, but do not count towards the marks. If your agent function is too slow, it will not be considered for the competition to ensure results can actually be announced by the end of the last lecture.