# Solent University

# Coursework Assessment Brief

# Assessment Details

|  |  |
| --- | --- |
| Unit Title: | Artificial Intelligence for Games |
| Unit Code: | DAC619 |
| Unit Leader: | Mark Bennett |
| Level: | 6 |
| Assessment Title: | Project AE1 |
| Assessment Number: | AE1 |
| Assessment Type: | Software Project and Report |
| Restrictions on Time/Word Count: | 1500 Words |
| Consequence of not meeting time/word count limit: | There is no penalty for submitting below the word/count limit, but students should be aware that there is a risk they may not maximise their potential mark. |
| Individual/Group: | Individual |
| Assessment Weighting: | 100% |
| Issue Date: | 01/10/2018 |
| Hand In Date: | 07/01/2019 |
| Planned Feedback Date: | February 2019 |
| Mode of Submission: | On-line |
| Number of copies to be submitted: | n/a |
| Anonymous Marking | This assessment is exempt from anonymous marking. |

# Assessment Task

Your task is to implement an AI algorithm to control two teams of agents acting as opponents in a ‘capture the flag’ combat game in a provided Unity project. You may use any suitable AI algorithm you have been shown in the lectures or any AI algorithm you have researched yourself. You may also use different algorithms for different agents if you wish for additional marks, however this is not essential. Your code should be tidy, well-structured and properly commented, you are encouraged to break the code up into multiple files and use folders to help keep it organised. You may *not* use any third party code for the AI itself (including code from the labs), you must implement the AI logic yourself. The AI algorithm should attempt to play the game to the best of its ability. Marks will be given for sophistication, that is, not just using simple ‘if’ statements and use of a specific AI algorithm.

You must also produce a report which includes a justification of your choice of algorithm and a discussion of the advantages and disadvantages of your chosen method. You must also compare your chosen algorithm with *at least* two other AI algorithms. Your report should describe your analysis and design for your proposed solution. Your design should include any appropriate diagrams related to the chosen algorithm e.g. a state diagram if using a state machine. You should also include pseudocode and flowcharts where appropriate i.e. only for AI logic and only if the logic is complex.

A test plan covering all the implemented behaviours should also be included in the report. The test plan should include a brief analysis of any problems encountered and a description of the solution. Finally include a brief critical evaluation discussing the strengths and weaknesses of your approach and any improvements you could make to the AI.

Ensure that your project will run on a machine other than the one used for development (test it on the University machines) and that any required assets are included with your project.

# The Unity Project

You will be provided with a simple Unity level, this level will contain two teams of AI agents, the flags, which start in the friendly base, some health kits and power ups which can be collected, they will respawn after 5 seconds, as well as a base for the AI agents to drop the flag to earn points. The project includes a code framework which implements various methods and properties that allow an AI agent to move to a location within the AI’s visual range, detect and collect objects, randomly wander the level and attack opponents. These properties and methods are detailed below and this information is duplicated in the provided Unity project at the top of the file you will be editing.

There are three members in each team, if a team member dies they will respawn after 5 seconds. The objective of the game is for each team to attempt take the opponents’ flag from their base and drop it in its own home base. If any team member, friendly or enemy, dies they will drop the flag. While the enemy flag is within the area of the friendly base (not while being carried, the flag must be dropped in the base) the score for that team will increment. The score for both teams will be indicated on the screen. There is no winning condition so the game will continue until the user stops it, this is to aid debugging.

The only script file you need to edit is AI.cs which contains a framework for your AI code. There are several other files providing supporting code which you are not encouraged to edit unless absolutely necessary. AI.cs has access to the AI agents’ actions, senses and data through three member variables called \_agentData which is of type AgentData, \_agentActions which is of type AgentActions, \_agentSenses which is of type Sensing and \_agentInventory which is of type InventoryController. All of these are script components and the scripts are viewable in the folder AI Support under the Scripts folder.

# The Code API

The script variables include the following methods, properties and public variables, you may look at the source code and Unity inspector for more details:

Predefined constants for Unity names and tags

Use these to access objects in your scripts

|  |  |
| --- | --- |
| **Unity Tags** | |
| public static class Tags | |
| public const string BlueTeam = "Blue Team"; | The tag assigned to blue team members. |
| public const string RedTeam = "Red Team"; | The tag assigned to red team members. |
| public const string Collectable = "Collectable"; | The tag assigned to collectable items (health kit and power up). |
| public const string Flag = "Flag"; | The tag assigned to the flags, blue or red. |

|  |  |
| --- | --- |
| **Unity GameObject names** | |
| public static class Names | |
| public const string PowerUp = "Power Up"; | Power up name |
| public const string HealthKit = "Health Kit"; | Health kit name. |
| public const string BlueFlag = "Blue Flag"; | The blue teams flag. |
| public const string RedFlag = "Red Flag"; | The red teams flag. |
| public const string BlueTeamMember1 = "Blue Team Member 1"; | Blue team member 1. |
| public const string BlueTeamMember2 = "Blue Team Member 2"; | Blue team member 2. |
| public const string BlueTeamMember3 = "Blue Team Member 3"; | Blue team member 3. |
| public const string RedTeamMember1 = "Red Team Member 1"; | Red team member 1. |
| public const string RedTeamMember2 = "Red Team Member 2"; | Red team member 2. |
| public const string RedTeamMember3 = "Red Team Member 3"; | Red team member 3. |

|  |  |
| --- | --- |
| **\_agentData properties and public variables** | |
| public bool IsAlive | Check if the agent is alive, returns true if agents alive, false otherwise. |
| public bool IsPoweredUp | Have we powered up, true if we’re powered up, false otherwise. |
| public int CurrentHitPoints | Our current hit points as an integer |
| public bool HasFriendlyFlag | True if we have collected our own flag |
| public bool HasEnemyFlag | True if we have collected the enemy flag |

|  |  |
| --- | --- |
| **\_agentActions methods** | |
| public bool MoveTo(GameObject target) | Move towards a target object. Takes a GameObject representing the target object as a parameter. Returns true if the location is on the navmesh, false otherwise. |
| public bool MoveTo(Vector3 target) | Move towards a target location. Takes a Vector3 representing the destination as a parameter. Returns true if the location is on the navmesh, false otherwise. |
| public bool MoveToRandomLocation() | Move to a random location on the level, returns true if the location is on the navmesh, false otherwise. |
| public void CollectItem(GameObject item) | Pick up an item from the level which is within reach of the agent and put it in the inventory. Takes a GameObject representing the item as a parameter. |
| public void DropItem(GameObject item) | Drop an inventory item or the flag at the agents’ location. Takes a GameObject representing the item as a parameter. |
| public void UseItem(GameObject item) | Use an item in the inventory (currently only health kit or power up). Takes a GameObject representing the item to use as a parameter. |
| public void AttackEnemy(GameObject enemy) | Attack the enemy if they are close enough. ). Takes a GameObject representing the enemy as a parameter. |
| public void Flee(GameObject enemy) | Move in the opposite direction to the enemy. Takes a GameObject representing the enemy as a parameter. |

|  |  |
| --- | --- |
| **\_agentSenses properties and methods** | |
| public List<GameObject> GetObjectsInViewByTag(string tag) | Return a list of objects with the same tag. Takes a string representing the Unity tag. Returns null if no objects with the specified tag are in view. |
| public GameObject GetObjectInViewByName(string name) | Returns a specific GameObject by name in view range. Takes a string representing the objects Unity name as a parameter. Returns null if named object is not in view. |
| public List<GameObject> GetObjectsInView() | Returns a list of objects within view range. Returns null if no objects are in view. |
| public List<GameObject> GetCollectablesInView() | Returns a list of objects with the tag Collectable within view range. Returns null if no collectable objects are in view. |
| public List<GameObject> GetFriendliesInView() | Returns a list of friendly team AI agents within view range. Returns null if no friendlies are in view. |
| public List<GameObject> GetEnemiesInView() | Returns a list of enemy team AI agents within view range. Returns null if no enemy are in view. |
| public bool IsItemInReach(GameObject item) | Checks if we are close enough to a specific collectible item to pick it up), returns true if the object is close enough, false otherwise. |
| public bool IsInAttackRange(GameObject target) | Check if we're with attacking range of the target), returns true if the target is in range, false otherwise. |

|  |  |
| --- | --- |
| **\_agentInventory properties and methods** | |
| public bool AddItem(GameObject item) | Adds an item to the inventory if there's enough room (max capacity is 'Constants.InventorySize'), returns true if the item has been successfully added to the inventory, false otherwise. |
| public GameObject GetItem(string itemName) | Retrieves an item from the inventory as a GameObject, returns null if the item is not in the inventory. |
| public bool HasItem(string itemName) | Checks if an item is stored in the inventory, returns true if the item is in the inventory, false otherwise. |

You can use the game objects name to access a GameObject from the sensing system. Thereafter all methods require the GameObject as a parameter.

# Hand-in Details

The Game Project and Project report must be submitted online before the deadline to receive full marks. Your submission should contain two files:

1. Your Project Report document, containing the design, testing, conclusions and any supporting diagrams, tables or charts.
   * The front page of this report should be a title page that contains at least the information  
     **"[Student Number]\_[LastName]\_[FirstName]\_DAC619\_AE1\_Report".**(square brackets indicate placeholders, they should not be in the final file name)
   * The document should preferably be in either MS Word or .pdf format.
   * Your source code may optionally be included as an appendix in the report.
2. A compressed/zipped file containing your Game Project
3. Only use .zip files, not .rar or other formats. If I can’t open your project I can’t mark it.
4. These two files should be submitted individually, not in one zip file, only the project file should be zipped.
   * Name the file  
     **"[Student Number]\_[LastName]\_[FirstName]\_DAC619\_AE1\_Project"**, (square brackets indicate placeholders, they should not be in the final file name).
   * Include all of solution and source code files:.cs files and any additional files you have created as resources for you project.
   * Make sure you include all source and resource files necessary to build and run your program, including any files supplied to you.
   * You do not need to include temporary or intermediate files created during the build process.

Due to the volume of network traffic especially near deadlines, the online submission system may be slower than you expect. Extensions for connection problems will not be granted unless university-wide. Give yourself plenty of time!

You **MUST** keep an exact copy of your project as backup in case of submission failure.

Ensure that the PC used to submit your work or create your submission package is free of viruses or malware. Submitting digital media containing any form of malware will result in significant penalty or failure.

# Assessment criteria

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***Component*** | ***Grade S, F3 – F1*** | ***Grade D3 – D1*** | ***Grade C3 – C1*** | ***Grade B3 – B1*** | ***Grade A4 – A1*** |
| **Implementation**  **50%** | Code not implemented or incorrect and non-functional  No behaviours implemented  Very crude approach e.g. just if statements  Code is unreadable and poorly structured | Code partially implemented but very basic, may have major errors  Very simple behaviours  Very simple AI implementation, small number of actions or decisions  Code does not follow best practice, poorly commented or structured | Code mostly implemented, may have a few major errors  AI successfully plays game  Moderately sophisticated approach taken, multiple decisions and actions  Attempt made at best practice, some comments, readable code, some structure | Code fully implemented, may have a few minor errors  AI uses tactics to win  Sophisticated approach taken, complex decision making, multiple actions  Best coding practice followed, naming, comments and structure are good | Code fully implemented and fully functional, very few minor errors  AI uses sophisticated tactics to win  Complex approach taken, multiple algorithms or complex decision making  Code very well structured, readable, good naming conventions and appropriate comments |
| **Report**  **50%** | No design included  No justification for algorithm choice  No discussion of  algorithm or comparison  No test plan  Little or no critical evaluation | Few behaviours designed  Little or no justification for algorithm choice  Superficial discussion of  algorithm and basic comparison  Basic test plan  Limited critical evaluation | Most behaviours designed, diagram or descriptions  Some justification for algorithm choice, pros and cons mentioned  Brief discussion of  algorithm and comparison  Test plan with results  Good critical evaluation | All behaviours designed with diagrams and descriptions  Detailed justification for algorithm choice. Pros and cons discussed in detail  Detailed discussion of  algorithm and detailed comparisons  Thorough test plan with analysis and results  Thorough critical evaluation | All behaviours correctly designed with clear diagrams and descriptions  Thorough justification for algorithm choice with all pros and cons discussed in detail.  Very detailed discussion of algorithm and detailed comparisons made  Detailed test plan with full coverage, analysis and results  Detailed critical evaluation |

# Learning Outcomes

This assessment will enable students to demonstrate in full or in part the learning outcomes identified in the unit descriptors.

# Late Submissions

Students are reminded that:

1. If this assessment is submitted late i.e. within 5 working days of the submission deadline, the mark will be capped at 40% if a pass mark is achieved;
2. If this assessment is submitted later than 5 working days after the submission deadline, the work will be regarded as a non-submission and will be awarded a zero;
3. If this assessment is being submitted as a referred piece of work (second or third attempt) then it must be submitted by the deadline date; any Refer assessment submitted late will be regarded as a non-submission and will be awarded a zero.

<http://portal.solent.ac.uk/documents/academic-services/academic-handbook/section-2/2o-assessment-principles-and-regulations.pdf?t=1534423842941>

# Extenuating Circumstances

The University’s Extenuating Circumstances procedure is in place if there are genuine circumstances that may prevent a student submitting an assessment. If students are not 'fit to study’, they can either request an extension to the submission deadline of 5 working days or they can request to submit the assessment at the next opportunity (Defer). In both instances students must submit an EC application with relevant evidence. If accepted by the EC Panel there will be no academic penalty for late submission or non-submission dependent on what is requested. Students are reminded that EC covers only short term issues (20 working days) and that if they experience longer term matters that impact on learning then they must contact the Student Hub for advice.

A summary of guidance notes for students is given below:

<http://portal.solent.ac.uk/documents/academic-services/academic-handbook/section-2/2p-extenuating-circumstances.pdf?t=1534423896787>

# Academic Misconduct

Any submission must be students’ own work and, where facts or ideas have been used from other sources, these sources must be appropriately referenced. The University’s Academic Handbook includes the definitions of all practices that will be deemed to constitute academic misconduct. Students should check this link before submitting their work.

Procedures relating to student academic misconduct are given below:

<http://portal.solent.ac.uk/support/official-documents/information-for-students/complaints-conduct/student-academic-misconduct.aspx>

**Ethics Policy**

The work being carried out by students must be in compliance with the Ethics Policy. Where there is an ethical issue, as specified within the Ethics Policy, then students will need an ethics release or an ethical approval prior to the start of the project.

The Ethics Policy is contained within Section 2S of the Academic Handbook:

<http://portal.solent.ac.uk/documents/academic-services/academic-handbook/section-2/2s-university-ethics-policy.pdf>

**Grade marking**

The University uses a letter grade scale for the marking of assessments. Unless students have been specifically informed otherwise their marked assignment will be awarded a letter grade. More detailed information on grade marking and the grade scale can be found on the portal and in the Student Handbook.

<http://portal.solent.ac.uk/documents/academic-services/academic-handbook/section-2/2o-annex-2-assessment-regulations-grade-marking-scale.pdf?t=1534424273208>

**Guidance for online submission through Solent Online Learning (SOL)**

<http://learn.solent.ac.uk/onlinesubmission>