University of Westminster
Faculty of Science and Technology

Module code and title:

5SENG001W / Algorithms: Theory, Design and Implementation

Module leader	**Dr. Epaminondas Kapetanios**
Unit	**Coursework**
Coursework mode	**Individual coursework**
Weight	**50%**
Qualifying mark	**30 marks**
Description	**Design and implementation of optimal path finding algorithmic approaches for navigation on a squared grid with randomly generated blocked cells (obstacles)**
Learning Outcomes	**LO2, LO3, LO4, LO5**
Handed Out:	**January 2018**
Due Date	**09/04/2018**
Expected deliverables	**1.** The source code in Java i.e., .java file, or a zipped file of your project including the source file(s), of your implemented solution. Your source code shall include header comments with your student ID and name. **You will be asked to demonstrate your solution and your source code to be inspected and explained during a viva in your tutorials commencing on the 9th of April, 2018.** The demonstrated source code must be identical with the one being submitted. A verification cross-check will be performed on a sampled set of submitted course works. Note: You will be zero-marked if you submit only the executable code, i.e., .class files, or source code without your student ID and name as header comments. **2.** A one A4 page document (Word, Plain text, or PDF) discussing the performance analysis of your algorithmic solution. The document should be structured as follows: - Student name and ID; - Input data sizes being used (size of squared grid and/or number of obstacles); - Times being spent on finding the optimal (shortest) path; - Conclusions on order-of-growth classification and justification.
Method of Submission:	**Electronically on BB**
Type of feedback and due date **Remark:**	**Formative feedback will be provided during tutorial sessions. *Verbal and formal feedback on the submitted CW will be provided during the CWK presentation/viva.* Feedback is due the week commencing on the 9th of April, 2018.** **Note: All marks will remain provisional until formally agreed by an Assessment Board.**
BCS Criteria meeting in this assignment:	**2.1.1 Knowledge and understanding of facts, concepts, principles & theories** **2.1.3 Problem solving strategies** **2.1.5 Deploy theory in design, implementation and evaluation of systems**

	2.2.2 Evaluate systems in terms of quality and trade-offs **2.3.2 Development of general transferable skills** **3.2.2 Defining problems, managing design process and evaluating outcomes** **4.1.1 Knowledge and understanding of scientific principles** **4.1.2 Knowledge and understanding of mathematical and statistical principles** **4.2.1 Use theoretical and practical methods in analysis and problem solving**

Coursework submission instructions

All coursework on this module is submitted via Blackboard only. It will automatically be scanned through a text matching system (designed to check for possible plagiarism). The system used will be either Turnitin or SafeAssign. The work you submit must be in Adobe Acrobat (.pdf) file format.

- You DO NOT need to attach a copy of the CA1 form;
- You DO need to include your name and student ID on the first page of your assignment.

To submit your assignment:

- Log on to Blackboard at http://learning.westminster.ac.uk;
- Go to the relevant module Blackboard site;
- Click on the *Assessment details* link on the left-hand side as advised by the module leader;
- Click on the link to the relevant assignment;
- Follow the 'upload' and 'submit' instructions.

If you are unable to submit your work due to a finance hold you must email your work to **fst-registry@westminster.ac.uk** by the same deadline, putting on the subject line the module code, assessment number, and your name. This shows that you have completed your work by the deadline. After the finance hold is lifted you must then submit the same work as normal on Blackboard, otherwise it will not be marked and you will get a fail for that assessment.

Assessment regulations

Refer to section 4 of the "How you study" guide for undergraduate students for a clarification of how you are assessed, penalties and late submissions, what constitutes plagiarism etc.

Penalty for Late Submission

If you submit your coursework late but within 24 hours or one working day of the specified deadline, 10 marks will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40 – 49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to

the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website:
http://www.westminster.ac.uk/study/current-students/resources/academic-regulations

Assessment Rationale:
The coursework will enable students to meet the following learning outcomes:

LO2: Be able to apply the theory for the effective design and implementation of appropriate data structures and algorithms in order to resolve the problem at hand.

LO3: Be able to analyse, predict, compare and contrast the performance of designed and implemented algorithms, particularly in the context of processing data.

LO4: Be able to use a range of typical data structures and collections as part of Application Programming Interfaces (APIs) offered by programming languages.

LO5: Be able to apply the theory for the definition and implementation of novel algorithms.

Problem description:

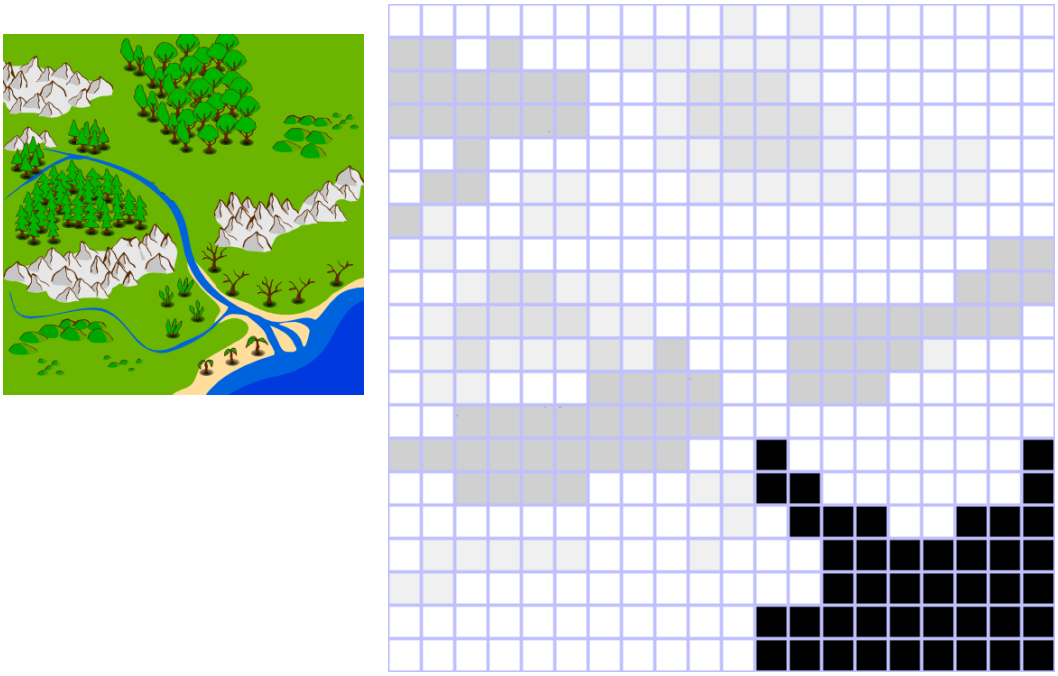
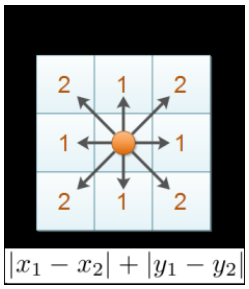


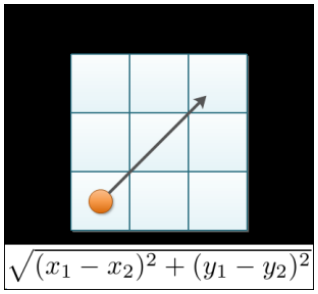
Figure 1: Grid based structure of the landscape

As depicted by the pictures above, you are asked to design, implement and analyse the performance, in Big-O notation, of heuristics, i.e., *Euclidean, Manhattan* and *Chebychev*, based shortert path finding algorithms between two arbitrarily chosen points A (start) and B (end). The picture to your left can be used as background image of a landscape, whereas the picture to your right depicts the landscape underpinning grid, a two-dimensional array, upon which your algorithm will operate. The following assumptions hold:

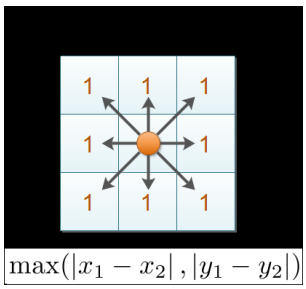
1. All white cells, apart from the black ones, are viable;
2. All adjacent cells can be visited by moving diagonally, vertically, or horizontally;
3. The cost for visiting a cell is dictated by the chosen heuristic as well as the four different shades of grey: *white shade of grey (weight = 1), lightest shade of grey (weight = 2), middle shade of grey (weight = 3), darkest shade of greay (weight = 4)*;
4. The definitions of the three distance metrics being used as heuristics, i.e., *Manhattan, Euclidean, Chebyshev,* are depicted by the figures below. The costs appearing on the figures below are based on the assumption of having cells with weights of one (1).



Manhattan distance



Euclidean distance



Chebyshev distance

Tasks to be performed:

Task 1. Set up a project (Java, Python, C++, etc.) by following the instructions and routine practised in all tutorial exercises**.**

Task 2. Implement the grid based data structure reflecting exactly the one depicted by figure 1, either as a file to be imported by the application, or a built-in memory based data structure.

Task 3. Implement a human-computer interaction mode, where the following will be possible:

a) Visualise the generated squared grid as depicted by figure 1;
b) The user should be in a position to mark interactively the start and end cells for the shortest path finding algorithm. This could be done by either entering the coordinates (i, j) as positive integers for two arbitrary *start* and *end* points, A and B, respectively, or by clicking these cells with the cursor and mouse;
c) Select from the three options of distance based metrics, i.e, Euclidean, Manhattan, Chebyshev, the one to apply for the shortest path finding algorithm.

Task 4. Design and implement your shortest path finding algorithm. Your implementation should switch over these three distance metrics and produce, for each one, as output: a) *visualisation of the shortest path on the squared grid*, b) *the total cost of the shortest path depending on the applied distance metric*.

Task 5. A brief report (no longer than half an A4-page) on the performance analysis of your algorithmic design and implementation. It suffices to discuss the performance analysis for one of these three distance metrics only, as no significant variations are expected. Your performance analysis should address the following:

a) Methodology based on an empirical study, e.g., *doubling hypothesis*, as already discussed in the lectures and practised with almost all tutorials, by using the Java internal watch (see also comments within the framework implementation);
b) Suggested *order-of-growth classification* (Big-O notation, i.e., average case) justified by and grounded on the results from (a) above.

Task 6. Demonstrate and **defend** the implemented algorithmic solution at viva by

a) Visualising the input grid as of figure 1. You may also make use of the landscape as a background image.

b) Marking the start and end points, A and B, respectively;

c) Demonstrate the shortest path in both terms, *drawn on the grid* or landscape, as well as a *number indicating the total cost*, for each one of the distance metrics;

d) Justifying the results.

To submit:

- The source code to be used during the viva; (Tasks 3, 4)
- The report about the algorithmic performance analysis (Task 5)

Coursework Marking Scheme:
CWK weighting 50% will be marked based on the following marking criteria:

Criteria	Mark per component	Mark provided	Comments
Task 4.1 (Manhattan distance metric)			
- **Drawn line**	10		
- **Calculated total cost**	10		
Task 4.2 (Euclidean distance metric)			
- **Drawn line**	10		
- **Calculated total cost**	10		
Task 4.3 (Chebyshev distance metric)			
- **Drawn line**	10		
- **Calculated total cost**	10		
Task 5.1			
- **Methodology (proper sizing of input data, i.e., size of grid, and times being spent)**	10		
- **Reasoning about changes in times**	10		
- **Order-of-growth-classification**	10		
- **Justification**	10		
-			
Total	**100**		

IMPORTANT NOTE: Failing to attend your viva will result into a grade of 30%, the highest possible, and in relation with the quality of your submissions. For instance, a 80% grade for the submitted CWK will result into 24%.