|  |  |  |
| --- | --- | --- |
|  | **Faculty of Computing, Engineering and Science** | Final mark awarded:\_\_\_\_\_ |

**Assessment Cover Sheet and Feedback Form 2017/18**

|  |  |  |  |
| --- | --- | --- | --- |
| Module Code:  **CS4S768** | Module Title:  **Advanced Data Structures and Algorithms** | | Module Lecturer:  Dr Janusz Kulon/ Dr Bertie Müller |
| Assessment Title and Tasks:  **Collision detection and response algorithm -** see attached tasks | | | Assessment No. **2 of 2** |
| No. of pages submitted in total including this page:  Completed by student | | | Word Count of submission  (if applicable): Completed by student |
| Date Set:  6/11/2017 | | Submission Date:  **Part 1: Demo of the Basic Development w/c 4th Dec 2017**  **Part 2: Final Submission of the assignment, including the report, code listings, executable version 12th Jan 2018** | Return Date:  5/02/2018 |

|  |  |
| --- | --- |
| ***Part A: Record of Submission (to be completed by Student)*** | |
| **Extenuating Circumstances**  If there are any exceptional circumstances that may have affected your ability to undertake or submit this assignment, make sure you contact the Advice Zone on your campus prior to your submission deadline. | |
| **Fit to sit policy**:  The University operates a fit to sit policy whereby you, in submitting or presenting yourself for an assessment, are declaring that you are fit to sit the assessment. You cannot subsequently claim that your performance in this assessment was affected by extenuating factors. | |
| **Plagiarism and Unfair Practice Declaration:**  By submitting this assessment, you declare that it is your own work and that the sources of information and material you have used (including the internet) have been fully identified and properly acknowledged as required[[1]](#footnote-1). Additionally, the work presented has not been submitted for any other assessment. You also understand that the Faculty reserves the right to investigate allegations of plagiarism or unfair practice which, if proven, could result in a fail in this assessment and may affect your progress. | |
| **Intellectual Property and Retention of Student Work:**  You understand that the University will retain a copy of any assessments submitted electronically for evidence and quality assurance purposes; requests for the removal of assessments will only be considered if the work contains information that is either politically and/or commercially sensitive (as determined by the University) and where requests are made by the relevant module leader or dissertation supervisor. | |
| **Details of Submission:**  Note that all work handed in after the submission date and within 5 working days will be capped at 40%[[2]](#footnote-2). No marks will be awarded if the assessment is submitted after the late submission date unless extenuating circumstances are applied for and accepted (Advice Zone to be consulted). | |
| You are required to acknowledge that you have read the above statements by writing your student number(s) in the box: | Student Number(s): |

**IT IS YOUR RESPONSIBILITY TO KEEP RECORDS OF ALL WORK SUBMITTED**

|  |
| --- |
| **Part B: Marking and Assessment**  **(to be completed by Module Lecturer)** |
| This assignment will be marked out of 100%  This assignment contributes to 50% of the total module marks.  This assignment is bonded / non- bonded. Details :  All coursework marks gained in this module are combined to form an overall element of assessment. In any coursework referral, the overall element will be capped at 40%. |
| **Assessment Task:**  See attached |
| **Learning Outcomes to be assessed** (as specified in the validated module descriptor <https://icis.southwales.ac.uk/> ):  LO1: Demonstrate a critical understanding of the concepts associated with applying and implementing a variety of data structures and algorithms  LO2: Create data structures and appraise their role of in producing software solutions to non-trivial programming problems |
| **Grading Criteria:** |

|  |  |  |
| --- | --- | --- |
| **Marking Scheme** | **Marks Available** | **Marks Awarded** |
| **Part 1 (All students to do this):** |  |  |
| An original and applicable introduction, background information (max 3 pages) | **10** |  |
| A complete program listing containing comprehensive comments which explain fully how your program works | **10** |  |
| Collision response algorithm – design and implementation | **15** |  |
| Collision detection algorithm - design and implementation | **15** |  |
| Testing of the application (develop a few tests to ensure the results are correct) Screen printouts for each test showing results and demonstrating the program in full working order. | **10** |  |
| Demonstration and critical evaluation of your application | **20** |  |
| **Part 2: Advanced Work:**  As part of the advanced development the collision detection and resolution algorithm should be extended to handle more geometrical shapes including rectangular blocks and convex polygons. | **20** |  |
|  |  |  |
| **Total** | **100** |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | **Assessment Criteria** | | | | | | | |  | | **Fail (0-29)** | **Narrow Fail (30-39)** | **3rd Class/Pass (40-49)** | **Lower 2nd Class / Pass (50-59)** | **Upper 2nd Class / Merit (60-69)** | **1st Class / Distinction (70 -100)** | | | An original and applicable introduction, background information (10) | | Very limited introduction, little or no references given | Limited introduction, < 3 references, little critical reflection on the literature sources | Satisfactory introduction, >3 relevant references cited and discussed, reasonable awareness of the subject demonstrated. | Good introduction, >5 relevant references cited and referred to demonstrating good understanding of the subject | Comprehensive introduction, >6 relevant references not only cited by critically evaluated, demonstrating a very good understanding of the topic of collision detection | Excellent introduction, >10 relevant references quoted and reflected upon demonstrating a critical understanding of the concepts associated with applying and implementing a  collision detection algorithms | | | A complete program listing containing comprehensive comments (10) | | No or very limited comments frequently unhelpful or misleading | Comments are incomplete and internal documentation is inadequate | Not all the comments are complete or internal documentation is inadequate. Some comments are unhelpful or occasionally misleading. | Almost all the comments are complete but internal documentation is in some small fashion inadequate. Comments usually clarify meaning. Unhelpful comments may exist. | Comments are nearly complete. Internal documentation is nearly complete and generally well suited to the program Comments generally clarify meaning where needed. | Comments are completed to a very good standard. Internal documentation is complete and well suited to the program. Comments clarify meaning where needed. | | | Collision detection algorithm (CDA) - design and implementation(15) | | CDA poorly implemented,  CPP code does not adhere to OOP principles,  Limited or no evidence of the OO design,  Poorly executed UML diagrams containing many mistakes etc | CDA implemented but contains significant omissions or mistakes,  CPP code does not always follow the OOP principles,  Limited evidence of the OO design, UML diagrams etc | CDA implemented without significant omissions or errors,  CPP code follows the OOP principles,  Some evidence of the OO design, UML diagrams etc | CDA implemented with only small errors or omissions  CPP code follows OOP principles,  Evidence of the sound and efficient OO design, UML diagrams comply with the framework from the lectures | CDA implemented to a good standard  CPP code follow s OOP principles,  Evidence of the sound and efficient OO design,  UML diagrams well executed showing correct relationships and realisation methods | CDA implemented to a good very standard  CPP code fully adheres to OOP principles,  Ample evidence of the sound and efficient OO design,  UML diagrams well executed showing correct relationships and realisation methods | | | Collision response algorithm (CRA) – design and implementation(15) | | CRA poorly implemented,  CPP code does not adhere to OOP principles,  Limited or no evidence of the OO design,  Poorly executed UML diagrams containing many mistakes etc | CRA implemented but contains significant omissions or mistakes,  CPP code does not always follow the OOP principles,  Limited evidence of the OO design, UML diagrams etc | CRA implemented without significant omissions or errors,  CPP code follows the OOP principles,  Some evidence of the OO design, UML diagrams etc | CRA implemented with only small errors or omissions  CPP code follows OOP principles,  Evidence of the sound and efficient OO design, UML diagrams comply with the framework from the lectures | CRA implemented to a good standard  CPP code follow s OOP principles,  Evidence of the sound and efficient OO design,  UML diagrams well executed showing correct relationships and realisation methods | CRA implemented to a good very standard  CPP code fully adheres to OOP principles,  Ample evidence of the sound and efficient OO design,  UML diagrams well executed showing correct relationships and realisation methods | | | Testing of the application(10) | | No or very little evidence of software testing. | Little evidence of software testing. | Some evidence of software testing. | Good evidence of software testing. | Significant testing of the program , well documented examples. | Comprehensive testing of the program , well documented examples. | | | Demonstration and critical evaluation of your application(20) | | Poor demonstration of limited program functionality or  Program does not run or crashes. | Very limited demonstration of some program functionality, Program sometimes does not run correctly or crashes | Basic demonstration the program functionality | Good demonstration of the program functionality, however limited to mostly simple examples of particle systems | V. Good demonstration the program functionality with some examples of challenging scenarios of particle systems | Excellent demonstration the program functionality with a number of examples of challenging scenarios of particle systems | | | Advanced Work (20) | | The advanced work has not been attempted or contains major flaws | The advanced work has been attempted but contains several minor flaws | The advanced work has been completed to a limited degree, and contains number of minor flaws or omissions | The significant part of the advanced work has been completed , however still contains a small number of minor flaws | The advanced work has been completed to a v. good standard and contains very few minor flaws | The advanced work has been completed to a very high standard with almost no flows or omissions | |   **Feedback/feed-forward** (linked to assessment criteria):   * Areas where you have done well: * Feedback from this assessment to help you to improve future assessments: * Other comments | | |
| **Mark:** | **Marker’s Signature:** | **Date:** |
| **Work on this module has been marked, double marked/moderated in**  **line with USW procedures.** | | |
| *Provisional mark only: subject to change and / or confirmation by*  *the Assessment Board* | | |

|  |  |
| --- | --- |
| **Part C: Reflections on Assessment**  **(to be completed by student – optional)** | |
| **Use of previous feedback:**  In this assessment, I have taken/took note of the following points in feedback on previous work: | |
| **Please indicate which of the following you feel/felt applies/applied to your submitted work**   * A reasonable attempt. I could have developed some of the   sections further.   * A good attempt, displaying my understanding and learning, with   analysis in some parts.   * A very good attempt. The work demonstrates my clear   understanding of the learning supported by relevant literature and scholarly work with good analysis and evaluation.   * An excellent attempt, with clear application of literature and   scholarly work, demonstrating significant analysis and evaluation. | |
| **What I found most difficult about this assessment:** |  |
| **The areas where I would value/would have valued feedback:** |  |

# ASSESSMENT - DETAILED REQUIREMENTS

**Part 1 Basic Development**

Collision detection is fundamental to many applications, including computer games, physical simulations, robotics and virtual prototyping. In computer games, collision detection ensures that the illusion of a solid world is maintained. Without collision detection, characters could go through walls and other obstacles. Collision detection involves algorithms checking for intersection of two given objects. Simulating what happens once a collision is detected is sometimes referred to as "collision response".

Your main task in this assignment is to design and implement a 2D rigid body collision detection and response algorithm. Collisions are to be handled sequentially and resolved using physical laws of motion, including gravity and drag force. It should be possible to set the properties of simulated objects such as: initial velocity, position, mass, acceleration and geometry. In addition to detecting and resolving collisions between objects in motion the program should be checking the bounds of the window in order to make sure that the objects stay within the confined space.

The project is to be implemented in object-oriented C++ using the principles of polymorphism and inheritance and making use of the various parts discussed in the lectures and tutorials. OpenGL should be used as a visualisation platform for the application. StarUML software should be used to produce diagrams illustrating program architecture and the logic of the algorithm.

The development of the program should be adequately documented to provide evidence that the application runs as expected and has been tested accordingly. This report must include code listing and testing (expected screen shots). Source code should be well commented.

In its basic development the collision detection and resolution algorithm should support circular shapes and handle an arbitrary number objects at the same time.

**Part 2 Advanced Work**

As part of the advanced development the collision detection and resolution algorithm should be extended to handle more geometrical shapes including rectangles and convex polygons.

**NOTE**

As with all statements of requirements, the above description of the problem contains errors, omissions and ambiguities. It is part of this coursework that you identify and resolve these, either through communication with your tutor, or through making assumptions which should be clearly described in the work that you hand in for this stage.

Be aware that the system you are building is your own, not all students will necessarily be given the same responses by the tutors.

**EXPECTED MODE OF WORKING**

This is an individual exercise. We need to ensure that the work you hand in **is developed by you.** This does not prevent you from asking for help from your tutors or from discussing elements of the work with colleagues.

1. University Academic Misconduct Regulations [↑](#footnote-ref-1)
2. Information on exclusions to this rule is available from the Advice Centre at each Campus [↑](#footnote-ref-2)