|  |
| --- |
| **The following questions assess the criteria**: “*detailed description and explanation of links between concepts, terminology, processes, and principles”* |

**Part A – Knowledge and Application – Answer in brief, succinct point form**

1. Briefly explain what the Big-θ notation is a measure of, and describe how most developers subjectively “consider” or “value judge” the following complexities shown in the graph:

|  |  |
| --- | --- |
|  | *(you can make annotations on the graph)* |

1. Briefly explain the main difference in algorithm between a **Bubble** and **Insertion** sort:
2. Label or demarcate examples of the following OOP elements in the algorithm provided:

|  |  |  |
| --- | --- | --- |
| **CLASS A** DEFINITION:  Data Members:   * Set Public Variable X to 1 * Set Protected Variable Y to 2 * Set Private Variable Z to 3   Class Methods:   * change(i): *to be implemented* | **CLASS B** DEFINITION:   * INHERITS **CLASS A** * REFERENCES **CLASS C**   Class Methods:  grow(): Call C.someMethod(X)  change(i): Set Y to i | *OOP elements to identify:*  ⮹ Class Name  ⮹ Encapsulation  ⮹ Polymorphism  ⮹ Inheritance  ⮹ Abstraction |

1. For each of the following algorithms or computational problems, fill in the following table with whichever is missing – either the **Name** or a **Short Description of intent**:

|  |  |  |
| --- | --- | --- |
| **Image** | **Name** | **Short Description of Intent** (1-2 sentences max) |
|  | **Perlin Noise Algorithm** |  |
|  |  | Find the shortest possible route that visits every city exactly once and returns to the starting point. This is a problem about optimization. |
| **patternMatching.png** | **Gale-Shapely Algorithm** |  |
|  |  | Efficient method for computing the greatest common divisor (GCD) of two numbers. |
| **fisherYates.pngfisherYates2.png** | **Fisher Yates Shuffle** |  |

|  |
| --- |
| **The following questions assess the criteria**: “*Detailed and effective application of set processes to solve simple and familiar problems”* |

|  |
| --- |
| **Breadth First Search:** |
|  |
| **Depth First Search:** |
|  |

5. Perform a **breadth first** and a **depth first** search on the following tree, recording *an* order in which the elements might be traversed:

A

/ \

B C

/ / \

D E F

/ \

G H

6. Desk check the following algorithm by recording the OUTPUT of X and Y:

|  |  |  |  |
| --- | --- | --- | --- |
| BEGIN MAIN  SET variable X to 40  SET variable Y to 70  OUTPUT X and Y  DO:  IF X == *findMax* (X,Y) THEN X = X – Y  ELSE Y = Y – X  END IF  OUTPUT X and Y  WHILE NOT (X == Y OR X==0 OR Y==0)  END MAIN | *findMax* method (requires numbers A and B):  BEGIN  IF (A > B) THEN RETURN A  ELSE RETURN B  END | X | Y |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

7. The following function F uses **recursion** to increase the variable X all the way up to the value of Y (you can assume X will always be an integer value lower than the integer Y). Fill in the three blank lines:

|  |  |
| --- | --- |
| *F* (X, Y):  IF \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_:  RETURN \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ELSE:  RETURN \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | BEGIN MAIN  SET variable Temp to *F*(1, 3)  OUTPUT(Temp)  END MAIN  **(\*should output 3)** |

8. Calculate the resulting set from the following set binary operations:

* Let A = {1, 2, 3}
* Let B = {3, 4, 5}
* Let C = {1, 4, 5}

|  |  |
| --- | --- |
| Operations | Resulting Set |
| A∪B∪C |  |
| (A∩B) ∪(B-C) |  |
| If {4} in (A∆B): ((C∆A)-B)∪(A∩C)  Else: ((C∩A)-B)∪(B∆C) |  |
| BEGIN  D = {}  E = {}  For i = 1 to 5:  Append i to set D  Append (D∩C) to set E  Next i  If ( {1} in E ): Set E to D-A  Else: Set E to B∪C  END | **Result of Set E only:** |

*9. Draw the most efficient logic circuits for the following expressions. W, X, Y and Z are variables of type Boolean:*

|  |
| --- |
| IF Z AND NOT(X) IS TRUE: |
|  |
| IF (X AND (Y OR NOT(Z))) OR NOT( NOT(W AND Z)) IS TRUE: |
|  |

*10. Design logic gates for the following truth tables*:

|  |  |  |
| --- | --- | --- |
| x | y | Output |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| X | y | Output |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

**End of Part A: Knowledge and Application.**

|  |
| --- |
| **Criteria assessed implicitly throughout all questions in Part B**: “*comprehensive construction of documentation and fluent presentation of information using suitable communication conventions to convey meaning appropriate to the context*” (in presentation of technical ideas, design concepts, solutions and evaluations). |

**Part B -** *2D procedurally generated platformer:* **Case Study**

A designer is interested in developing a 2D Platform game that uses *procedural generation* (i.e. levels can be “randomly” generated), mapping each created level to a grid so that the levels can be saved, recreated or shared. They have come to you to develop their algorithm. They have designed some legal, ideal and illegal concept sketches, but need your help to devise the rest of the logic for their game.

*You are welcome to suggest further level additions or modifications if it helps you demonstrate the IPT criteria of analysis, synthesis and evaluation.*

**Legal:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Player |  |  |  |  |  | Goal |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Player |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | Goal |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Player |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  | Goal |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

**Ideal:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Player |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | Goal |  |
| ground |  |  |  |  |  |  |  |  |  |
| filler |  |  | Spikes |  |  | Spikes | Spikes |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | AI 🡪 |  |  |  |  | Player |
|  |  | Goal |  |  |  | 🡨 |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Spikes |  |  | Spikes |  |  |  | Spikes |  |  |

*Where 🡪 and 🡨 are patrol limits*

**Illegal:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  | Goal |  |  |  |  | Player |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

*Fail reason: platform width too long*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  | Goal |  |  |  |  | Player |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

*Fail reason: gap width too far*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Player |  |  |  |  |  |  |  |  |  |
|  |  |  |  | Goal |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

*Fail reason: player goal proximity too close*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Player |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | Goal |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

*Fail reason: unreachable jump*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Player |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | Goal |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

*Fail reason: terrain height change too rapid*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Player |  |  |  |  |  |  |  |  |  |
|  |  |  |  | AI 🡪 | 🡨 |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | Goal |  |  |
|  |  |  | Spikes |  |  | Spikes | Spikes |  |  |

*Fail reason: player and goal must spawn on land*

**Part B -** *2D procedurally generated platformer:* **Analysis - Answer in brief, succinct point form**

Deconstruct the concept sketches, and interpret the feedback to **analyze the required specifications** for an algorithm that will procedurally generate legal platform game levels.

Your analysis could include:

* a list of suitable, salient **features** your algorithm will **seek** to deliver, using a “multiple perspectives approach” by addressing elements such as:
  + meeting the designer (client) needs;
  + meeting multiple player (customer) needs (such as detailing a valid player controller, or by attempting to scale levels or difficulty);
  + any further additions or modifications you wish to make (as the algorithm developer).
* an overarching set of rules that your algorithm must adhere to, such as:
  + physics (e.g. gravity) and other governing logic (AI, spawn, etc.)
  + valid / invalid level design rules or “checks”
  + other requirements or assumptions you are going to make
* a list of caveats, constraints or limits that will be outside the scope of your algorithm.

|  |
| --- |
| **Criteria assessed**: “*Detailed interpretation and analysis of problems and situations from multiple perspectives*” |

Space below for diagrams if required, or room to keep writing if necessary:

**Part B -** *2D procedurally generated platformer***: Synthesis**

Utilize appropriate algorithmic design methods and principles to implement your specified algorithmic solution for procedurally generating legal platform game levels.

Your synthesis to this unrehearsed problem should be:

* significant in scope
* complex in nature
* make use of consistent algorithmic conventions

|  |
| --- |
| **Criteria assessed**: “*designed and developed effective solutions to unrehearsed or complex problems*” |

|  |
| --- |
|  |

|  |
| --- |
|  |

*Write on the back of this page if more room is needed.*

**Part B -** *2D procedurally generated platformer***: Evaluation - Answer in brief, succinct point form**

**Evaluate** your algorithm by applying self-determined and prescribed criteria, reasoning or evidence to draw conclusions and make future recommendations. Your evaluation should encompass a use of logic and reason in a range of evaluation approaches to achieve success.

|  |
| --- |
| **Criteria assessed**: “*… application of self-determined and prescribed criteria, reasoning and evidence to draw conclusions and make supported recommendations.* ” |

|  |  |
| --- | --- |
| **Prescribed Criteria** | **Definition** |
| Scalability | The ability of the algorithm to continue to function well when it (or its context) is changed in size or volume. |
| Efficiency | The requirement for the algorithm to perform its required tasks at the highest speed, or with the minimum resource usage possible. |
| Modularity | Separating the functionality of an algorithm into independent, interchangeable modules, to assist management, reusability and decrease clutter. |
| **Self-determined Criteria** | **Definition** |
|  |  |
|  |  |

Space below for diagrams if required, or room to keep writing if necessary:

*Standards Matrix for this task:*

| Dimension | A | B | C | D | E |
| --- | --- | --- | --- | --- | --- |
| Knowledge and application | The student work has the following characteristics:   * detailed description and explanation of links between **algorithm** concepts, terminology, processes, and principles | The student work has the following characteristics:   * description and explanation of **algorithm** concepts, terminology, processes, and principles | The student work has the following characteristics:   * description of **algorithm** concepts, terminology, processes, and principles | The student work has the following characteristics:   * statements of **algorithm** facts | The student work has the following characteristics:   * reproduction of isolated **algorithm** facts |
| * detailed and effective application of set processes to solve simple and familiar **algorithmic** problems. | * effective application of set processes to solve simple and familiar **algorithmic** problems. | * application of set processes to solve simple or familiar **algorithmic** problems. | * elements of set processes to partially solve simple or familiar **algorithmic** problems. | * elements of set processes used. |
| Analysis and synthesis | The student work has the following characteristics:   * detailed interpretation and analysis of **algorithm** problems and situations from multiple perspectives | The student work has the following characteristics:   * interpretation and analysis of **algorithm** problems and situations | The student work has the following characteristics:   * analysis of **algorithm** problems and situations | The student work has the following characteristics:   * identification and classification of **algorithm** problems or situations | The student work has the following characteristics:   * restated **algorithm** problems or situations |
| * designed and developed effective solutions to unrehearsed or complex **algorithmic** problems. | * designed and developed solutions for unrehearsed or complex **algorithmic** problems. | * designed and developed partial solutions for unrehearsed or complex **algorithmic** problems. | * designed or developed elements of solutions for unrehearsed or complex **algorithmic** problems. | * superficial elements of unrehearsed or complex **algorithmic** problems. |
| Evaluation and communication | The student work has the following characteristics:   * … application of self-determined and prescribed criteria, reasoning and evidence to draw conclusions and make supported recommendations. | The student work has the following characteristics:   * … application of prescribed criteria, reasoning and evidence to draw conclusions and make supported recommendations | The student work has the following characteristics:   * … application of prescribed criteria, reasoning or evidence to draw conclusions and make recommendations | The student work has the following characteristics:   * … draws inferences | The student work has the following characteristics:   * elements of testing |
| * comprehensive construction of documentation and fluent presentation of information using suitable communication conventions to convey meaning appropriate to the context. | * effective construction of documentation and effective presentation of information using suitable communication conventions to convey meaning appropriate to the context. | * construction of documentation and presentation of information using communication conventions to convey meaning. | * presentation of information using elements of communication conventions. | * presentation of information. |

**Comments:**