# Food Magnate Simulation

## Programming Theory Questions – Mark Scheme

| **Q** | **Answer/Guidance** | **Marks** |
| --- | --- | --- |
| 1a | 1 mark:   * NextID | 1 |
| 1b | 1 mark for either of:   * GetRandomLocation * FindOutIfHouseholdEatsOut | 1 |
| 1c | 1 mark:   * LargeSettlement | 1 |
| 1d | 1 mark:   * CloseCompany | 1 |
| 1e | 1 mark for each of:   * GetListOfOutlets * GetDistanceBetweenTwoOutlets | 2 |
| 1f | 1 mark:   * ToLower | 1 |
| 1g | 1 mark:   * Outlets | 1 |
| 1h | 1 mark:   * SimulationSettlement | 1 |
| 2 | 3 marks:   * 1 mark for showing calculation: 0.25 \* 0.5 * 1 mark for explaining: 0.5 probability nested in selection statement for 0.25 probability // any explanation that makes clear that the 0.5 probability is only evaluated 25% of the time * 1 mark for correct probability: 0.125 / 12.5% / 1/8 / 1 in 8 | 3 |
| 3 | 3 marks:   * (Dimensions of) Settlement must be passed (as a parameter) * Selection/if statement to compare x and y to dimensions of settlement * Creation/construction of outlet inside the selection/if structure   (credit can be given for code that is described, but not code in isolation) | 3 |
| 4 | 3 marks:   * *Dictionary* uses key and value pairs * An index would be the key; the Household object would be the value * Households would be unordered // no guarantee of order | 3 |
| 5 | 4 marks:   * One X (or Y) coordinate is subtracted from the other * Result of this is squared * Square root of this is found (award this mark and the previous mark if the answer makes it clear that the distance is always positive) * Result is added to the result of the same process performed on Y (or X) coordinate | 4 |
| 6 | 2 marks:   * Generates a random float/fraction/single * To set ChanceEatOutPerDay // to set the probability of eating out | 2 |

| **Q** | **Answer/Guidance** | **Marks** |
| --- | --- | --- |
| 7 | 3 marks:   * Randomly set to either 0 or 1 * Determines whether fuel cost goes up or down // if it is set to zero, fuel cost goes up * If it is set to 1, fuel cost goes down   (last two marks can be given if reference is instead made to variable FuelCostChange) | 3 |
| 8 | 3 marks:   * 100,000: company's balance * 200: x position of company / first outlet * 203: y position of company / first outlet   (last 2 marks cannot be awarded if they simply refer to 'x' and 'y' – these must be in the correct context) | 3 |
| 9 | 5 marks:   * Loop through all companies (in companies list) * Convert search term / parameter / company name to lower case * Compare this with (lower-case version of) each company's name / call to GetName * If there is a match, the index / value of current / company's location within list is returned * If loop ends without a match, -1 is returned | 5 |
| 10 | 3 marks:   * User enters 1 / selects 'open new outlet' at the menu * X coordinate is outside the range of the settlement * **or** Y coordinate is within the range of the settlement   (for last 2 marks, there must be a clear expression that either x or y being beyond the bounds of the settlement would trigger the message) | 3 |
| 11 | 3 marks:   * LargeSettlement constructor calls Settlement/superclass constructor * Settlement constructor uses (x and y) values of 1,000 when placing houses * Reduction of XSize/YSize would take place after this // houses are already positioned according to value of 1,000, even if those values are subsequently reduced | 3 |
| 12 | 4 marks:   * Constructor overloading is multiple constructors in a single class * Each constructor requires a different signature / order of parameter types * Settlement class could have used constructor overloading * LargeSettlement would need (in addition to Settlement constructor parameters) parameters for (additional) size and number of households | 4 |
| 13 | 3 marks:  **ProcessCostChangeEvent**  **GetName**  AlterAvgCostPerMeal  **AlterDailyCosts**  1 mark per underlined term; order of GetName and AlterDailyCosts is unimportant, but each must be on the bottom row for the mark. | 3 |
| 14 | 2 marks:   * 'else' clause would be executed * Values for a named chef restaurant would be used // variables would be assigned values 20, 40 and a random float multiplied by 50 | 2 |
| **TOTAL MARKS** | | **50** |