| **(a)** | moveCandyToFirstRow |  |  |
| --- | --- | --- | --- |
|  | **Scoring Criteria** | **Decision Rules** |  |
| **1** | Accesses all necessary elements of column col of box (*no bounds errors*) | Responses **can** still earn the point even if they  • return early, as long as the loop bounds are appropriate    Responses **will not** earn the point if they  • fail to access an element of box in the loop  • access the elements of box incorrectly | **1 point** |
| **2** | Compares candy box element at row 0  and column col to null | Responses **can** still earn the point even if they  • make the comparison inside the loop or at some incorrect point in the code    Responses **will not** earn the point if they  • fail to use != or equivalent | **1 point** |
| **3** | Identifies and moves appropriate Candy object to first row if necessary (*algorithm*) | Responses **can** still earn the point even if they  • return early, as long as the identify-and- move are inside a loop and would work if the loop got that far    Responses **will not** earn the point if they  • fail to replace the moved Candy object with null  • move or swap objects when the first row is already occupied | **1 point** |
| **4** | Returns true when non-empty square is identified and false if non-empty square is not identified in the context of a loop (*algorithm*) | Responses **can** still earn the point even if they  • fail to replace the moved Candy object with null  • incorrectly identify a non-empty square    Responses **will not** earn the point if they  • return early | **1 point** |
|  |  | **Total for part (a)** | **4 points** |

| **(b)** | removeNextByFlavor |  |  |
| --- | --- | --- | --- |
|  | **Scoring Criteria** | **Decision Rules** |  |
| **5** | Traverses box in specified order (bottom to top, left to right) (*no bounds errors*) | Responses **will not** earn the point if they  • fail to access an element of box in the loop  • access the elements of box incorrectly | **1 point** |
| **6** | Guards against a method call on a  null element of the candy box (*in the context of an* if *statement*) | Responses **will not** earn the point if they  • fail to use != or equivalent | **1 point** |
| **7** | Calls getFlavor on a Candy  object | Responses **can** still earn the point even if they  • access the element of the candy box incorrectly  • call getFlavor on the incorrect Candy  object    Responses **will not** earn the point if they  • call getFlavor on an object of a different type or on the Candy class  • attempt to create a Candy object  • include parameters | **1 point** |
| **8** | Compares a Candy object’s flavor with flavor parameter | Responses **will not** earn the point if they  • fail to use equals | **1 point** |
| **9** | Replaces first matching Candy object with null and returns replaced object (*algorithm* | Responses **can** still earn the point even if they  • access elements of the candy box incorrectly  • call getFlavor incorrectly or on a  null Candy object  • compare a Candy object’s flavor to the parameter using ==  Responses **will not** earn the point if they  • fail to replace the identified object within the 2D array with null before returning  • fail to return null when no matching  Candy object is found  • set multiple elements to null | **1 point** |
|  |  | **Total for part (b)** | **5 points** |
|  | **Question-specific penalties** |  |  |
|  | None |  |  |
|  |  | **Total for question 4** | **9 points** |